



REFERENCE ONLY

UNIVERSITY OF LONDON THESIS

Degree *Pho*

Year *2005*

Name of Author *ALVES C.F*

**COPYRIGHT**

This is a thesis accepted for a Higher Degree of the University of London. It is an unpublished typescript and the copyright is held by the author. All persons consulting the thesis must read and abide by the Copyright Declaration below.

**COPYRIGHT DECLARATION**

I recognise that the copyright of the above-described thesis rests with the author and that no quotation from it or information derived from it may be published without the prior written consent of the author.

**LOANS**

Theses may not be lent to individuals, but the Senate House Library may lend a copy to approved libraries within the United Kingdom, for consultation solely on the premises of those libraries. Application should be made to: Inter-Library Loans, Senate House Library, Senate House, Malet Street, London WC1E 7HU.

**REPRODUCTION**

University of London theses may not be reproduced without explicit written permission from the Senate House Library. Enquiries should be addressed to the Theses Section of the Library. Regulations concerning reproduction vary according to the date of acceptance of the thesis and are listed below as guidelines.

- A. Before 1962. Permission granted only upon the prior written consent of the author. (The Senate House Library will provide addresses where possible).
- B. 1962 - 1974. In many cases the author has agreed to permit copying upon completion of a Copyright Declaration.
- C. 1975 - 1988. Most theses may be copied upon completion of a Copyright Declaration.
- D. 1989 onwards. Most theses may be copied.

*This thesis comes within category D.*



This copy has been deposited in the Library of *UCL*



This copy has been deposited in the Senate House Library, Senate House, Malet Street, London WC1E 7HU.



Department of Computer Science  
University College London  
University of London

**Managing Mismatches  
in COTS-Based Development**

Carina Frota Alves  
`c.alves@cs.ucl.ac.uk`



Submitted in partial fulfilment of the requirements  
for the degree of Doctor of Philosophy  
at the University of London

September 2005

UMI Number: U591797

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI U591797

Published by ProQuest LLC 2013. Copyright in the Dissertation held by the Author.  
Microform Edition © ProQuest LLC.

All rights reserved. This work is protected against  
unauthorized copying under Title 17, United States Code.



ProQuest LLC  
789 East Eisenhower Parkway  
P.O. Box 1346  
Ann Arbor, MI 48106-1346

## Abstract

The prospect of reducing the time, cost and risk of system development has increased the interest in developing systems from COTS (Commercial-Off-The-Shelf) products. The development of complex COTS-based systems is known to be an intricate and risk prone process. There are three main reasons for this. Firstly, suppliers develop COTS products with the objective of satisfying the needs of the marketplace rather than the specific needs of the acquirer organization. Secondly, COTS products are often delivered as black-boxes, which means that the understanding of COTS features is frequently partial and uncertain on the part of the acquirer. Thirdly, in order to sustain competitive advantage, suppliers regularly modify their products, hence forcing customers to update their systems. These challenges do not occur in traditional software development, they are particular attributes of COTS-based development (CBD). As a result, new processes, methods and models are needed to support the development of COTS-based systems.

The selection of COTS products is one of the most important activities taking place in the context of CBD. It involves the assessment of how well COTS products satisfy customer requirements. Due to the nature of COTS, mismatches may occur between what is wanted from the system (i.e. customer requirements) and what the system is able to provide (i.e. its features). In addition, a number of risks may arise from these mismatches, such as insufficient COTS adherence to requirements, low confidence in COTS quality, and unwanted COTS features. We argue that the successful selection of a suitable COTS product depends on the effective analysis of mismatches and management of risks.

This thesis proposes a novel method, called TAOS (Tradeoff Analysis for COTS-based Systems), to guide the selection of COTS products. TAOS offers a systematic approach to assess the suitability of COTS products by exploring mismatches, handling risks and suggesting possible tradeoffs to be made. The method uses a goal-oriented approach to specify the requirements of the acquirer organization.

We demonstrate how utility theory can be used to compare COTS alternatives by examining the degree to which COTS products satisfy requirements, and therefore inform the decision making process. As a way to complement the quantitative assessment obtained from the use of utility theory, we present a set of templates to build exploratory scenarios and define heuristics to facilitate the tradeoff analysis. We also present a strategy to identify and manage risks. To establish the effectiveness of TAOS to improve the quality of decisions made during the selection process, we have conducted a number of case studies in different domains.

*Para Mario,  
por todos os sonhos e poesias que viajamos*

## Acknowledgements

I am deeply grateful to my supervisor Anthony Finkelstein, for his guidance, experience and encouragement. Thanks Anthony, for not only being my supervisor, but more importantly, for being such a good friend, for having been so supportive whenever I needed. I am also deeply indebted to Rami Bahsoon for all the stimulating discussions and endless support during the whole period of my PhD. I would like to thank Andrew Dingwall-Smith for the precious help correcting my English. Xavier Franch for the fun and constructive discussions we had during our meetings in London and Barcelona. Also thanks to Marc Burgess for his support in the electronic document and records management case study.

My years in London were full of joy and laugh thanks to the guys from the WDP (Weekly Drinking Procedure). The weekly exploration of the old pubs of London, tasting the best real ales are memorable memories I will take with me from the cold island. Also special thanks to Leticia, Genaina, Licia, Stefanos, Clovis, Anis, Marike, Maria, Florian and Marc for their friendship and enjoyable time I spent with all of them. Thanks to Gizela and her family for adopting me during the last months of my PhD.

The support of my family, my parents Edna and Ferreira, my sister Clarissa and my brother Leonardo, has been invaluable. My parents have always been so enthusiastic with all achievements in my academic life. There is one person who has followed every single step of this research, my partner Mario. His inspiration, love and wittiness have helped me to overcome the (sometimes) arduous moments and enjoy even more the cheerful periods. Thank you, Ma!

Finally, thanks to CAPES, the Brazilian funding agency that has sponsored my research. I will do my best to return to the Brazilian people all the investment I received towards my education.

# Contents

<b>1</b>	<b>Introduction</b>	<b>14</b>
1.1	Background . . . . .	14
1.2	Research Scope . . . . .	16
1.3	Thesis Contributions . . . . .	19
1.4	Thesis outline . . . . .	20
<b>2</b>	<b>COTS-Based Development</b>	<b>22</b>
2.1	Defining COTS Software . . . . .	22
2.2	Impact of COTS Software . . . . .	24
2.2.1	Requirements Flexibility . . . . .	25
2.2.2	Dilution of Control and Uncertainty . . . . .	26
2.3	COTS-Based Development Lifecycle . . . . .	27
2.4	Accommodating Changes . . . . .	29
2.5	Summary . . . . .	30
<b>3</b>	<b>COTS Selection</b>	<b>32</b>
3.1	COTS Selection Approaches . . . . .	32
3.2	Decision Making Techniques . . . . .	41
3.3	Risk Management . . . . .	45



3.4	Summary . . . . .	46
<b>4</b>	<b>Requirements Engineering for COTS-based Systems</b>	<b>48</b>
4.1	Traditional Requirements Engineering . . . . .	48
4.1.1	Traditional RE Lifecycle Process . . . . .	49
4.1.2	Goal-oriented Requirements Engineering . . . . .	52
4.2	Conflict Management and Negotiation Strategies . . . . .	53
4.3	COTS-based Requirements Engineering . . . . .	56
4.4	Summary . . . . .	59
<b>5</b>	<b>Motivation</b>	<b>61</b>
5.1	Understanding the COTS Selection Problem . . . . .	61
5.1.1	AFTN Message Switch Case Study . . . . .	62
5.2	Requirements for a COTS Selection Method . . . . .	65
5.3	Research Objectives . . . . .	67
5.4	Introducing a Running Example . . . . .	68
5.5	Summary . . . . .	69
<b>6</b>	<b>The TAOS Method</b>	<b>70</b>
6.1	Main Principles . . . . .	71
6.2	Specify Goals . . . . .	72
6.2.1	Acquire high level goals . . . . .	73
6.2.2	Refine Goals . . . . .	74
6.2.3	Define Satisfaction Functions . . . . .	77
6.2.4	Identify interactions between goals . . . . .	81
6.2.5	Prioritise Goals . . . . .	85
6.2.6	Manage Goals . . . . .	96

6.3	Assess COTS Candidates . . . . .	98
6.3.1	Identification . . . . .	98
6.3.2	Pre-Qualification . . . . .	99
6.3.3	Exhaustive Screening . . . . .	100
6.3.4	Trial . . . . .	101
6.4	Perform Matching . . . . .	104
6.5	Analyse Mismatches and Manage Risks . . . . .	112
6.5.1	Risk Identification . . . . .	112
6.5.2	Risk Analysis and Prioritization . . . . .	114
6.5.3	Risk Resolution . . . . .	116
6.6	Select COTS Solution . . . . .	121
6.7	Summary . . . . .	127
<b>7</b>	<b>Evaluation</b>	<b>128</b>
7.1	Mail Server Case Study . . . . .	130
7.1.1	Background . . . . .	130
7.1.2	Methodology and Case Study Artifacts . . . . .	132
7.1.3	Critical Evaluation . . . . .	145
7.1.4	Lessons Learned . . . . .	146
7.2	Electronic Document and Records Management System Case Study . . . . .	147
7.2.1	Background . . . . .	147
7.2.2	Methodology and Case Study Artifacts . . . . .	149
7.2.3	Critical Evaluation . . . . .	156
7.2.4	Lessons Learned . . . . .	161
7.3	Comparative Study . . . . .	162
7.3.1	Comparison with other COTS selection methods . . . . .	163

7.3.2	Critical Evaluation . . . . .	169
7.4	Summary . . . . .	169
<b>8</b>	<b>Conclusions and Future Work</b>	<b>171</b>
8.1	Summary of Contributions . . . . .	171
8.2	Future Work . . . . .	172
	<b>References</b>	<b>175</b>

# List of Figures

2.1	Comparison between traditional and COTS-based development process model (adapted from [Brownsword et al., 2000]) . . . . .	27
2.2	CISD COTS-based Process Model . . . . .	29
3.1	Overview of PORE Method (adapted from [Ncube, 2000]) . . . . .	34
3.2	Overview of decision-making process . . . . .	42
4.1	Spiral model of the requirements engineering process (adapted from [Sommerville and Kontonya, 1998]) . . . . .	50
4.2	Categories of conflict management . . . . .	55
6.1	Overview of TAOS Method . . . . .	71
6.2	Overview of Goal Specification Phase . . . . .	73
6.3	Goal meta-model . . . . .	74
6.4	Goal decomposition for the online bookshop system . . . . .	75
6.5	Refinement alternatives to measure system learnability . . . . .	76
6.6	Guidelines to refine goals . . . . .	77
6.7	Acceptable interval for different operational goals . . . . .	79
6.8	Satisfaction degrees for operational goals fast sales authorization and safe sales authorization . . . . .	81
6.9	Positive and negative interactions between goals . . . . .	84

6.10	Goal Weights . . . . .	85
6.11	Goal weights and composed weights . . . . .	88
6.12	Satisfaction degrees of goals $g_1$ and $g_2$ . . . . .	89
6.13	Substitution rates for goal $g_1$ . . . . .	90
6.14	Substitution rates for goal $g_2$ . . . . .	91
6.15	Substitution rates of sales authorization time . . . . .	93
6.16	Corresponding Tradeoff between $g_1$ and $g_2$ . . . . .	94
6.17	Corresponding Tradeoff between safe sales authorization and fast sales au- thorization . . . . .	95
6.18	Guidelines to Manage Changes in Goals . . . . .	96
6.19	COTS Assessment Phase . . . . .	97
6.20	Guidelines for initial rejection of COTS Candidates . . . . .	100
6.21	Guidelines for detailed rejection of COTS Candidates . . . . .	102
6.22	Guidelines for designing test cases . . . . .	103
6.23	Matching Meta Model . . . . .	106
6.24	Matching between goals and COTS Products . . . . .	111
6.25	Risk factors and resulting risk exposure zones . . . . .	115
6.26	Risk events posed by COTS B with resulting risk exposure . . . . .	116
6.27	Risk Management Taxonomy . . . . .	118
6.28	Factors to assess the worthiness of each COTS solution . . . . .	122
7.1	Basic electronic mail system . . . . .	131
7.2	Acceptable interval and satisfaction function for operational goals $g_{3.1}$ , $g_{5.2}$ , $g_{7.1.1}$ . . . . .	136
7.3	Interaction between goals for the Mail Server system . . . . .	138
7.4	Possible configurations of integrated system composed by mail server, anti- virus and anti-spam tools . . . . .	142

# List of Tables

3.1	Fundamental scale for pairwise comparisons . . . . .	43
5.1	Set of requirements a COTS selection method would need to satisfy . . . . .	65
5.2	Set of high level goals for the online system of Black Books bookshop . . . . .	68
6.1	Scales of measurement and possible metric examples . . . . .	77
6.2	Satisfaction degree associated with matching patterns . . . . .	110
6.3	Risk Identification Template . . . . .	114
6.4	Risk exposure factors for COTS B . . . . .	116
6.5	Scenario 1 - Explore each COTS intensive system . . . . .	123
6.6	Scenario 2 - Explore Mismatch Situations . . . . .	125
6.7	Scenario 3 - Explore Conflict between Goals . . . . .	126
7.1	Goal Specification for Mail Server . . . . .	135
7.2	Identifying risk events . . . . .	142
7.3	Exploratory Scenario - Mismatch between $g_{4.1}$ and Exchange 2000 . . . . .	144
7.4	High Level Goals that the EDRM product to be acquired should satisfy . . . . .	151
7.5	Comparison of selection methods - Part 1 . . . . .	164
7.6	Comparison of selection methods - Part 2 . . . . .	165
7.7	Comparison of selection methods - Part 3 . . . . .	166

7.8 Comparison of selection methods - Part 4 . . . . . 167

# Chapter 1

## Introduction

### 1.1 Background

The use of Commercial Off-The-Shelf (COTS) software is becoming an economic and strategic need for many organisations. The prospect of reducing the time and cost associated with software development has led organizations to an increasing interest in acquiring and integrating commercial products instead of developing systems from scratch. The idea of developing systems from software components is not a new one, the principle of mass produced software components has been suggested back in 1968 by [McIlroy, 1968]. Since then the software industry has moved progressively towards a coarse-grain component-based paradigm. This trend can be verified by the large availability of off-the-shelf components such as software development environments, operating systems, database management systems, and business specific applications.

COTS-based systems comprise a spectrum ranging from COTS-solution systems at one extreme to COTS-intensive systems at the other extreme [Wallnau et al., 1998]. COTS-solution refers to systems in which a single large product is acquired to provide specific and substantial system functionality. An example of this type of system is Enterprise Resource Planning (ERP) packages. ERP attempts to integrate all departments and functions across a company into a single software system that can provide particular needs to different departments [Koch, 2002]. By having an integrated software solution across the organization, ERP has the potential for staff to easily share information and communicate with each other, and hence contributing to increase organizational value. On the other hand, the successful implementation of ERP requires organizations to carry out an extensive business change process in order to align organizational processes with package capabilities. In COTS-solution systems, the product is tightly coupled to the user business process, which means that the organization needs to engage in an extensive adaptation process at the same time that the product has to be tailored. In contrast to



COTS-solution, COTS-intensive systems involve the integration of several products from different suppliers. In COTS-intensive system development, multiple, heterogeneous and interdependent products have to be acquired, modified and integrated in order to provide the desired system functionality. As the size and complexity of systems continues to grow, this type of software system is becoming increasingly ubiquitous. COTS-intensive systems are generally more flexible than COTS-solution systems because alternative configurations of integrated products enable customers to explore different ways to achieve the desired functionality. By contrast, in COTS-solution systems the entire product functionality is determined by a single supplier.

Organizations expect to gain a number of benefits including faster system development time, lower development costs, and continual product improvement by using COTS products. Given that the number of customers using off-the-shelf products is likely to be wide and diverse, the opportunities to surface problems increases and ultimately leads to a more stable and mature product. Moreover, the costs to acquire COTS products is expected to be lower than to develop bespoke systems because the product development costs can be shared among many users. It is important to note that there are a number of difficulties involved as well, mostly due to the fact that suppliers have full control over product functionality and evolution [Brereton, 2004]. Customers have very limited control and access to product internals since COTS are typically delivered as black-boxes. To complicate the situation, COTS products evolves frequently because of pressure from market competition and other economic factors.

Due to all these issues, the development of complex COTS-based systems is known to be an intricate and risk prone process [Wallnau et al., 2002, Finkelstein and Spanoudakis, 1996]. The failure of complex COTS-based systems is hardly related to a single root, instead the origins are the interaction of different factors ranging from inadequate product alignment to customer requirements and poor product quality to social and organizational impact caused by the new product [Finkelstein, 2001]. In order to deal with these difficulties, the evaluation and selection of COTS products have to be carefully performed [Alves and Castro, 2001, Ncube and Maiden, 1999, Meyers and Oberndorf, 2001]. During the selection activity, the functionality and quality of COTS candidates have to be assessed against the requirements of the acquirer organization. A good understanding of the requirements that the organization wishes to achieve by procuring the new software product is fundamental to enable an effective evaluation of COTS alternatives. As a result, the requirements engineering process plays a main role during the COTS selection process.

Given that COTS products are developed to satisfy the requirements of an entire market instead of the specific requirements of the buyer organization, it is possible that mismatches may occur between what is desired by the stakeholders of the organization and what it is possible to achieve with the COTS product [Alves and Finkelstein, 2003]. In order to control mismatches, it is necessary to define requirements in a flexible way, so that

it is possible to distinguish between critical requirements and those that are negotiable. An additional difficulty is that the degree of confidence in the way COTS candidates satisfy a particular requirement may be low because of the lack of trusted and complete information about COTS capabilities. These situations give rise to a number of risks that may prevent the successful selection of COTS. Examples of risk include non-adherence to stakeholder requirements, low confidence in COTS quality, adaptability and integration problems. When selecting COTS products, organizations should be prepared to make simultaneous tradeoffs among requirements the organization wishes to satisfy, availability of COTS product capabilities, risks imposed by COTS alternatives and costs to develop the system (i.e. acquisition, adaptation and integration costs).

The development of systems using COTS products brings several new opportunities and benefits over the traditional system development. However, together with the positive aspects, there are also a number of challenges and risks that do not occur in traditional system development. New processes, methods and tools have, therefore, to be developed to suit the needs of COTS-based development paradigm.

## 1.2 Research Scope

COTS-Based Development (CBD) refers to the development of software systems from commercial components. Much effort has been devoted trying to agree on a definition of what constitutes a COTS product [Clark and Torchiano, 2004]. For the purposes of this thesis we follow the definition given by [Meyers and Oberndorf, 2001] of what a COTS product is: *COTS is a product that is: sold, leased, or licensed to the general public; offered by a vendor trying to profit from it; supported and evolved by the vendor, which retains the intellectual property rights, available in multiple, identical copies; used without internal modification by a consumer.* This thesis focuses on a particular class of COTS: large grained software components [Wallnau et al., 2002]. This class of components refers to large, independent products providing substantial system functionality. Examples of COTS software products under this class include Enterprise Resource Planning, content management systems, mail servers, database management systems, etc. It is important to note that large grained components can compose both COTS-solution and COTS-intensive systems. On the other hand, fine grained components can only be employed in COTS-intensive systems since this class of components refers to implementable units of software that need to be integrated in order to form a functional system [Szyperski, 1998].

The COTS-based paradigm involves a new way of developing systems where different skills, knowledge and abilities as well as changed roles and responsibilities are necessary. The COTS-based development process involves the participation of the following groups of stakeholders:

*The acquirer organization* - who evaluates and selects COTS products in the market,

*Suppliers of COTS products* - companies responsible to develop and market COTS products,

*Application developer or system integrator* - who acts as intermediary between suppliers and acquirer organizations, developing systems that incorporates COTS products and custom-built code.

*End users* - people who will use the final integrated COTS-based system.

The typical relationship between stakeholders is characterised by two main players: large, powerful COTS suppliers and acquirer organizations. A small group of employees from the organization is usually appointed to be part of the evaluation team. The objective of the evaluation team is to evaluate COTS candidates, negotiate contract with supplier, make selection decisions and recommend the “best” COTS product to be acquired. The primary motivation of this thesis is to investigate the processes of COTS evaluation and selection. Another important topic we aim to explore is the requirements process for COTS-based development. More specifically, this thesis addresses the following issues:

**Requirements prioritization and negotiation** - Given the market-oriented nature of COTS products, it is likely that a number of requirements that organizations aim to achieve may not be satisfied by any available product. Organizations have to prioritise and negotiate their requirements against the features provided by products. We argue that requirements prioritization and negotiation are core activities to be performed during the selection of COTS products.

**Decision making** - The selection of COTS products is a typical multi-criteria decision making problem [Ncube and Dean, 2002]. In order to select a suitable COTS product, alternatives have to be assessed against the requirements of the acquirer organization and ranked according to their satisfaction to requirements. We present a well-defined approach to measure the satisfaction degree of requirements as a way to objectively inform the decision process.

**COTS tradeoff analysis** - When developing COTS-based systems, acquirer organizations have to make simultaneous tradeoffs among requirements, COTS, risks and costs. We aim to analyse and manage these tradeoffs as well as to explore how decisions made in one aspect may affect others.

**Risk analysis** - The development of COTS-based systems involves a number of risks. In order to make rationale decisions, acquirer organizations have to assess the risks involved with COTS alternatives and balance them against the benefits of each alternative.

**Selection of multiple COTS** - This topic is partially addressed in this thesis. The evaluation

and selection of several COTS products is a highly complex task. This is due to the fact that products may conflict with each other in terms of dependencies and interoperability problems, for instance. To simplify the scope of situations we aim to address in this thesis, we assume that multiple COTS-based systems involve the selection of one substantial product controlling the selection decisions and other complimentary, smaller products surrounding it. In this thesis we solely consider the phases of evaluation and selection of multiple COTS-based systems. Substantial effort is necessary during the adaptation and integration phases, when the products have to be composed within a single system architecture. However, these topics are beyond the scope of this research.

The following issues are outside the scope of this thesis:

**Buy vs. build decisions** - Given the size and complexity of the class of COTS applications we are interested to study, it is not economically feasible to develop such systems from scratch. Instead, we assume that buying commercial-off-the-shelf products is always considered to be the best strategic decision. Said that, we do not deal with the buy versus build decisions.

**COTS adaptation and integration** - We do not address the issues involved during the adaptation and integration of COTS products. These phases are subsequent to the selection activity which is the key focus of this thesis.

**Open source systems** - We consider that the selection of open source systems is fundamentally different from the selection of COTS products. This is due to the fact that the source code is available in these components, therefore, the system functionality is fully known whereas the internals of COTS products are usually unknown. Another key difference is the nature of system ownership, in open source the system owner is the community who has also permission to modify and extend the code. While in the context of CBD, the supplier has full control over the system source code. This thesis addresses the specific problems involved in selecting COTS products which are not suitable to support the selection of open source systems.

**Web services** - The development of systems using web services is outside the scope of this thesis. Similarly to what we said with respect to open source systems, we consider that web services require a different acquisition framework from the one we propose in this thesis to select COTS products. In particular, the relationship between supplier and customer in the software service engineering paradigm can be automatically established [Brereton, 2004]. Such a level of automated relationship is very different from what occurs in the relationship between supplier and customer of COTS products, which usually requires extensive interpersonal negotiation between both parties.

**COTS certification** - Certification of COTS products is an important way of increasing

trust in COTS products [Voas, 1998, Cechich and Piattini, 2004]. COTS certification consists of testing if a COTS product complies with quality standards as well as measuring the product maturity and functional suitability. Frequently, the certification testing is performed by third party bodies rather than by the acquirer organization. We assume that this is the chosen mechanism employed by organizations acquiring COTS products. COTS certification is therefore beyond the scope of this thesis.

### 1.3 Thesis Contributions

The overall objective of this thesis is to investigate new processes, strategies and mechanisms to support the requirement engineering process for COTS-based systems. We argue that by better supporting the COTS-based requirements process, it is possible to improve the quality of decisions made during the selection of COTS products. More specifically, the benefits of improving the quality of decisions are: (1) increasing the overall satisfaction of stakeholders; (2) reducing the risks of an inadequate product to be selected; (3) controlling the costs involved in the selection process. In order to address these issues, we have developed a novel method called TAOS (Tradeoff Analysis for cOts-based Systems). The main contributions present in TAOS include the following:

1. **Goal-oriented requirements process for COTS-based development** - We present a goal-oriented approach to elicit and model the requirements for the software system to be acquired. TAOS provides guidelines and heuristics to conduct the refinement of goals and identification of interactions (i.e. synergies and conflicts) among them. A technique based on utility theory is presented to facilitate the prioritization of goals and further negotiation of conflicting goals.
2. **Guidance to conduct COTS identification and assessment** - TAOS describes systematic guidance to identify potential COTS candidates in the market, to pre-qualify the best alternatives and to perform detailed testing in order to better understand COTS functionality and quality.
3. **Support the matching analysis between COTS and requirements** - The matching process involves the measurement of the degree in which COTS candidates satisfy the requirements of the acquirer organization. To facilitate this assessment, we developed a measurement strategy using utility theory techniques to obtain the satisfaction degree of requirements. TAOS also provides a set of matching patterns to formally categorise the degree in which requirements are satisfied by products, and consequently help the identification of mismatches between the requirements of the organization and what is possible to achieve with available COTS products.

4. **Heuristics to facilitate mismatch analysis and resolution** - We provide a number of heuristics to analyse the nature and impact of mismatches. These heuristics can guide the evaluation team in deciding if the mismatches may pose real risk to the selection project. In case risk events are detected, specific heuristics are proposed to guide the selection of appropriate risk resolution strategies to handle mismatches. These heuristics exploit trade-off analysis strategies as a way to handle mismatches between requirements and COTS products.

5. **Risk management strategy** - We present a comprehensive strategy to examine and manage risks. The risk management effort starts with the identification of risks events, then risks are prioritised based on their probability of occurrence and utility loss, finally risk management strategies are proposed.

6. **Exploratory scenarios to guide tradeoff decisions** - TAOS provides a set of templates to build exploratory scenarios. The objective of these scenarios is to capture and reason about all relevant issues involved in the decision of selecting a particular product. Exploratory scenarios allow the evaluation team to assess the overall worth of each COTS candidate.

7. **Evaluation of Results** - To establish the applicability and usefulness of the TAOS method, we present two case studies. The first is a real case study that investigates the selection of document management system for a Higher Education Institution. The second case study explores the multiple selection of products in the domain of electronic messaging systems. Besides these case studies, we also provide a comprehensive comparison of TAOS with existing COTS selection methods.

## 1.4 Thesis outline

The remainder of this thesis is divided in the following chapters:

Chapter 2 - discusses the COTS-based development process. This chapter explores the new challenges and difficulties organizations face when developing software systems using COTS products. It also describes core activities involved in CBD and discusses the changes in roles and responsibilities needed in the COTS-based development paradigm.

Chapter 3 - introduces the COTS selection process in detail. This chapter presents the core processes involved during the evaluation and selection of COTS products. It discusses existing work done in the area as well as investigates the importance of multiple-criteria decision making techniques for the selection process. Finally, research done in the areas of conflict resolution and risk management is discussed.

Chapter 4 - discusses the differences between the traditional requirements engineering process and the COTS-based one. This chapter explores the new concerns and difficulties engineers need to address when elaborating requirements for COTS-based systems. The chapter also discusses existing work done in the area.

Chapter 5 - defines the motivation for this research. This chapter presents a preliminary case study in order to better understand the COTS selection problem. Based on the results of the case study, a set of requirements that a COTS selection method would need to fulfill is defined. Then, the specific objectives of this research are presented. Finally, a running example that will be used to illustrate the TAOS method is described.

Chapter 6 - presents the TAOS method. The method has been developed with the objective of addressing the set of requirements for a COTS selection method defined in Chapter 5.

Chapter 7 - describes the evaluation of TAOS by means of three evaluation efforts: an industrial case study, a simulated case study and a critical comparison with existing selection methods.

Chapter 8 - summarises the contributions presented in this thesis and explores directions for future work.

## Chapter 2

# COTS-Based Development

In this chapter we give an account of COTS-based development. Firstly, we discuss the several definitions existing in the literature for the term COTS (Commercial-Off-The-Shelf) and contrast these with definitions of component. Secondly, we discuss the impact that using COTS products will bring to the development process in general and focus in more detail on issues that are related to requirements engineering process as this is a key aspect to be investigated in this thesis. Then, we describe the main activities of the COTS-based development process. Finally, we discuss how to accommodate the changes caused by the use of COTS products.

### 2.1 Defining COTS Software

Within the literature, many different definitions of components can be found [Carney and Long, 2000]. There are significant differences among the definitions in terms of granularity, scope and context in which components are used. In particular, the meaning of terms COTS and components are frequently used in a vague and confusing way. As a result, researchers and practitioners use each term with very different meanings. We agree with other researchers that it is better to define these terms in a broad perspective rather than limiting its coverage [Brereton and Budgen, 2000, Clark and Torchiano, 2004]. For the purposes of this thesis, we still need to define these terms in a clear-cut way to avoid the risk of ambiguity. Let us examine some definitions of COTS existing in the literature. [Torchiano and Morisio, 2004], for instance, provides an empirically based definition for COTS component: *“COTS is a commercially available or open source piece of software that other software projects can reuse and integrate into their own products.* For [Vidger and Dean, 1997], COTS is *“pre-existing software product; sold in many copies with minimal changes; whose customers have no control over specification, schedule and evolution; access to source code as well as internal documentation is*



*usually unavailable*". While [Basili and Boehm, 2001] argues that COTS software has the following characteristics: *"the buyer has no access to the source code, the vendor controls its development, and it has a nontrivial installed base (that is, more than one customer; more than a few copies"*. We can observe that the commercial nature of COTS appears in all definitions. In [Torchiano and Morisio, 2004] the notion of COTS is wider and also includes open source software, which generally has the source code available to the public. In a different way, Vidger and Basili seem to agree that customers have very limited access to the software internals. In this thesis, we prefer to adopt the definition of COTS given by [Meyers and Oberndorf, 2001] because it offers broad coverage and precision. According to them *"COTS is a product that is: sold, leased, or licensed to the general public; offered by a vendor trying to profit from it; supported and evolved by the vendor, which retains the intellectual property rights, available in multiple, identical copies; used without internal modification by a consumer"*. Besides COTS, other acronyms have been proposed in the literature, such as Modifiable Off-The-Shelf (MOTS), Government Off-The-Shelf (GOTS), Nondevelopmental item (NDI). The distinction among these terms is not always clear. In particular, there may exist different degrees of modification permitted as well as different notions of commercial software [Carney and Long, 2000]. In this thesis, we prefer to use the term COTS without its variations.

The definitions of COTS described above can be contrasted with other different classifications of software components. According to [Szyperski, 1998], software components are *"units of independent production, acquisition, and deployment that interact to form a functional system"*. [Ning, 1999] defines a component as *"an encapsulated, distributable and executable piece of software that provides and receives services through well-defined interfaces"*. In Szyperski and Ning's definitions a key feature to characterise software components is the ability to deploy components independently. According to Ning, components communicate with each other through well-defined interfaces. This does not necessarily mean that the internal code of the component is known, it can still be delivered as black-box. In terms of the granularity of components, both definitions are quite vague. Current research in component-based software engineering suggests that software components encompass middleware and component infrastructure such as Common Object Request Broker Architecture (CORBA), Common Object Model (COM) and Distributed Common Object Model (DCOM), Java Beans and Enterprise Java Beans. In a different vein, [Brown and Wallnau, 1998] defines business components as *"the software implementation of an autonomous business concept or business process. It consists of the software artifacts necessary to express, implement, and deploy the concept as a reusable element of a larger business system"*. Enterprise Resource Planning (ERP) systems are considered large scale business components [Koch, 2002]. ERP are software systems developed with the purpose of integrating different departments of organizations including product planning, purchasing, maintaining inventories, interacting with suppliers, providing customer service, and tracking orders.

Several works have focused on classifying COTS. These go from proposing taxonomies of application domain [Glass and Vessey, 1995] and quality models [Franch and Carvallo, 2002], to defining a set of common quality attributes [Cechich and Piattini, 2004, Bertoa and Vallecillo, 2002] for COTS products, and defining a set of attribute-value pairs in a cartesian space to categorise COTS products [Torchiano et al., 2002, Carney and Long, 2000]. The problem with approaches like [Glass and Vessey, 1995, Franch and Carvallo, 2002] is that they require significant domain knowledge to model the quality attributes of entire domains that have to be organized in a hierarchical fashion [Brereton et al., 2002]. The ISO/IEC 9126 standard is a widely adopted framework to model COTS domains [Franch and Carvallo, 2002, Cechich and Piattini, 2004, Bertoa and Vallecillo, 2002]. This standard defines a set of generic attributes to measure the quality of software products. [Torchiano et al., 2002], on the other hand, argues that ISO/IEC 9126 is insufficient to characterise COTS attributes. According to them, the evaluation of some quality attributes defined by the standard may require access to internal aspects of the software system, that may not be available in COTS products. Furthermore, he proposes a framework that solely evaluates external attributes of COTS products without the need of having the source code or other design information.

## 2.2 Impact of COTS Software

As software systems become increasingly complex, it is no longer feasible to develop systems totally from scratch. COTS-Based Development (CBD) has become a widespread strategy to develop large, complex software systems [Wallnau et al., 2002, Finkelstein and Spanoudakis, 1996]. There is a growing market offering COTS software components ranging from software development environments to operating systems, database management systems and business applications [Albert and Brownsword, 2002]. The potential benefits of using COTS products are increased software quality, reduced development costs and time [Meyers and Oberndorf, 2001, Brereton et al., 2002]. COTS products are used by a wide number of users, hence it increases the chances of detecting problems and ultimately leads to a more mature and stable product. Also due to the large customer base, the costs to develop a COTS product can be shared among several customers, which is generally a more cost-effective approach than building custom software systems. A reduced development time is due to the large availability of ready to use COTS products. Other secondary benefits of adopting CBS include timely maintenance, continual product improvement, and ease of modernization.

Enterprise Resource Planning (ERP) are well known examples of COTS products. Nowadays, ERP are key software systems to integrate the IT infrastructure of large and medium size organizations. According to a recent survey performed by PMP Research [ERP, 2005] among different sizes of companies in the UK (annual turnover ranging from 10 million to

5 billion), a market trend is that companies are opting to supplement ERP solutions with either best of breed packages or bespoke developments. (26%) integrate ERP applications with standalone choices, while (20%) complement their ERP solution with bespoke applications. By comparison, just (11%) have an enterprise IT system built entirely on either best of breed options or custom developments, and a further (16%) use a combination of these two. These results suggest that while ERP solutions continue to be the core infrastructure for enterprise systems, companies prefer to combine generic ERP packages with more specialised applications, either developed in-house or acquired from niche suppliers. With respect to project risks, cost constraint is considered one of the biggest threats to the success of ERP, followed by poor defined project objectives, difficulty of mapping business processes onto applications and problems in managing outcomes and expectations. From these results we can observe the importance of understanding project objectives and requirements of stakeholders in order to successfully implement large COTS packages like ERP. Other empirical works confirm the importance that requirements play in COTS-based development [Morisio et al., 2002, Boehm and Abts, 1999, Carney, 1998].

Besides the potential benefits promised by the use of COTS, the COTS-based paradigm involves a number of challenges and risks. In particular, dependence on supplier and flexibility on requirements are major issues that organizations willing to adopt COTS have to accept and properly handle. In addition, the use of COTS is likely to affect the whole software lifecycle process [Albert and Brownsword, 2002, Brown and Wallnau, 1998], hence forcing a continuous process of requirements definition and negotiation in order to successfully integrate the COTS solution onto the organization business processes [Rolland and Prakash, 2001]. These issues can be seen as tradeoffs that organizations have to tolerate in order to obtain the expected gains in schedule, effort and cost [Morisio et al., 2000]. In the following sections we discuss these issues in further detail.

### 2.2.1 Requirements Flexibility

COTS products are developed for the mass market [Vitharana, 2003]. This means that when developing COTS suppliers aim to satisfy the needs of the whole marketplace rather than implementing the specific requirements of a particular customer [Alves and Finkelstein, 2002]. As a result, organizations willing to adopt COTS solutions should be prepared to accept flexibility in their requirements where the number of “hard” requirements should be minimised while more “elastic” requirements should be preferred. Moreover, requirements for the new system should not be so strict that either exclude the use of COTS entirely nor require large product modification in order to satisfy very specific requirements [Carney, 1998]. An interesting approach is to let, to a certain extent, the available COTS features drive the elaboration of requirements [Finkelstein and Spanoudakis, 1996, Boehm, 2000]. This strategy suggests that

the specification of requirements should occur in parallel with the evaluation of COTS products. Other researchers in the area seem to agree that the processes of requirements engineering and COTS selection are highly intertwined [Chung and Cooper, 2003, Lewis and Morris, 2004, Ncube, 2000].

The evaluation of COTS demands some inexact matching between customer requirements and product features [Alves, 2003]. There may be, for instance, requirements not satisfied by any available package, requirements satisfied by some joint packages, requirements partially satisfied, features not requested initially but that could be nice to have, irrelevant or even unwanted features. Moreover, there are situations where critical requirements cannot be entirely satisfied without considerable product adaptation and cases where requirements must be compromised to accept product limitations. Leaving requirements negotiable until the end of the selection process allow stakeholders to have a clear picture of which requirements are possible to be implemented, and hence prevents them of having high expectations that cannot be satisfied. In addition, this practice ensures that promising suppliers are not rejected in the beginning of the evaluation process just because their products do not satisfy some requirements.

An important issue to be considered when specifying requirements for COTS-based systems is the prioritisation of requirements. One of the main objectives of the prioritisation process is to distinguish “nice to have” requirements that could be more easily traded off in case none of the available products meet such desired features from “core” requirements that should not be compromised. A potential danger is to ask the chosen supplier to change or add features to fit every requirement specified by the stakeholders. Suppliers do modify features in response to market trends, but they are unlikely to change features to satisfy individual users [Boehm and Abts, 1999]. It may happen that suppliers try their best to satisfy the needs of high calibre customers. Even in such cases it is unwise to make large modifications on COTS products as they will be more like a bespoke system and future maintenance will be the customer’s own responsibility.

### 2.2.2 Dilution of Control and Uncertainty

COTS products are developed based on a set of requirements that vendors believe will meet the widest number of potential customers [Deifel, 1998]. Therefore, COTS products are designed to satisfy very generic requirements. For customers to ensure that candidate products satisfy at least their critical requirements, they must have an accurate understanding of product features to decide which parts must be adapted to conform with their particular needs. Given that COTS are commonly delivered as black-boxes, customers have limited visibility into the product internals and no control over its operation. Moreover, the information available mainly consists of commercial and marketing-oriented documentation. As a result, customers have no guarantees that the product will perform

as advertised by the vendor.

A key challenge customer organizations need to overcome when evaluating COTS is the fact that it is generally difficult to obtain a comprehensive description of products since competitive pressures in the market force vendors to innovate and differentiate product features rather than standardize them [Wallnau et al., 2002]. This lack of widely agreed vocabulary among suppliers results in inconsistent and biased information about product features and quality.

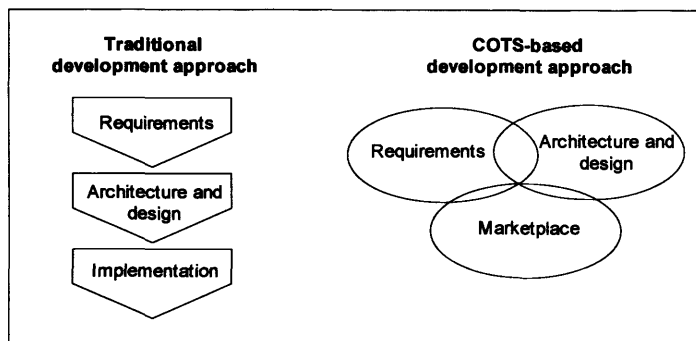


Figure 2.1: Comparison between traditional and COTS-based development process model (adapted from [Brownsword et al., 2000])

An additional complication is that suppliers have full control over product releases and upgrades [Albert and Brownsword, 2002]. Very often suppliers stop supporting old versions, forcing customers to continuously upgrade their system [Boehm, 2000]. New versions can destabilise the system and generate conflicts with other parts of the system. Therefore, customers are put into unexpected situations over which they have no control. Even though suppliers greatly rely on users to test and improve their products, users have little choice when new versions are released but to update the system or to shift to a competitor supplier. Changing the current product for a new one can bring a number of undesirable situations such as system instability and extra costs. Consequently, it is quite convenient to accept supplier decisions and continue upgrading their product. In the case of large products such as ERP, this decision is even more appropriate. According to a survey with ERP customers in the UK [ERP, 2005], only (2%) of organizations consider switching vendors after their packages have been fully integrated.

## 2.3 COTS-Based Development Lifecycle

We have discussed in the previous section that the paradigm shift to COTS-based development brings several new challenges and risks to organizations. These challenges and

risks will, in turn, profoundly affect the system development process [Vitharana, 2003]. The market-driven nature of COTS-based development brings circumstances that did not exist in the traditional system development. As a result, COTS-based development requires new practices and approaches to be developed and adopted. A fundamental difference is that in the traditional software development, the boundary between development phases is clearly defined. It starts with the specification of requirements, then the system design and architecture are defined and the software system is implemented. Even if more dynamic, spiral-type process models are adopted, it is quite easy to distinguish the limits between each phase. On the other hand, the COTS-based development process consists of highly interactive processes, where the distinctions between the activities of requirements specification, architecture design and COTS evaluation/integration become blurred [Wallnau et al., 2002].

According to [Brownsword et al., 2000], developers have to consider requirements, architecture and marketplace simultaneously as these three aspects are interdependent. More specifically, it is necessary to observe how decisions made in one aspect will affect the others. This means that the development process involves simultaneous tradeoffs among these three sources of influence [Meyers and Oberndorf, 2001]. Figure 2.1 provides a comparison between the traditional and COTS-based development processes. This figure suggests that the traditional development process uses the waterfall model [Sommerville, 2004]. However, we consider that even in traditional software development, spiral models [Boehm, 1988] are more widely used and appropriate than the obsolete waterfall model. Given the interactive character of COTS-based development, the process model for such systems should be inspired on spiral, risk-driven models [Boehm and Abts, 1999].

Despite the differences between the traditional and COTS-based paradigms, the development process of COTS-based systems involves a number of activities that are also present in the traditional development. However, these activities may suffer changes in the context of CBD. For instance, the requirements engineering for COTS-based systems is considerably different from the traditional one [Ncube, 2000]. As we discussed in Section 2.2.1, the flexibility in requirements imposed by the use of COTS brings completely new situations to the requirements process. Although some traditional requirements techniques are still effective, new approaches are needed to suit specific circumstances of CBD. The requirements process for COTS-based systems is an important issue to be treated in this thesis and it will be explored in further detail in Chapter 4. The frequent modifications in COTS products affect the system architecture and design in such a way that the architecture has to be sufficiently flexible to accommodate evolving requirements [Bahsoon and Emmerich, 2004] and COTS products. The implementation activity substantially loses its importance in CBD, in its place appears the integration activity. Finally, during the system maintenance activity for COTS-based systems, which includes product upgrades and supplier switch, it is also necessary to perform the phases

of requirements analysis, integration and testing.

Other process activities are particular of COTS-based development, such as product evaluation, license negotiation, supplier relationship management, adaptation and integration. [Tran and Liu, 1997] proposes a model called (COTS-based Integrated Systems Development) CISD to cover the main technical activities of CBD. Figure 2.2 gives an overview of the model, which consists of three phases called product identification, evaluation and integration as well as various sub-phases. Several variations of CBD process models have been proposed [Albert and Brownsword, 2002, Cooper and Chung, 2002]. The similarity among all process models is the highly interactive nature of the CBD process model. Many of the activities exclusive of CBD have an organizational and social nature, we discuss these aspects in more detail in the next section.

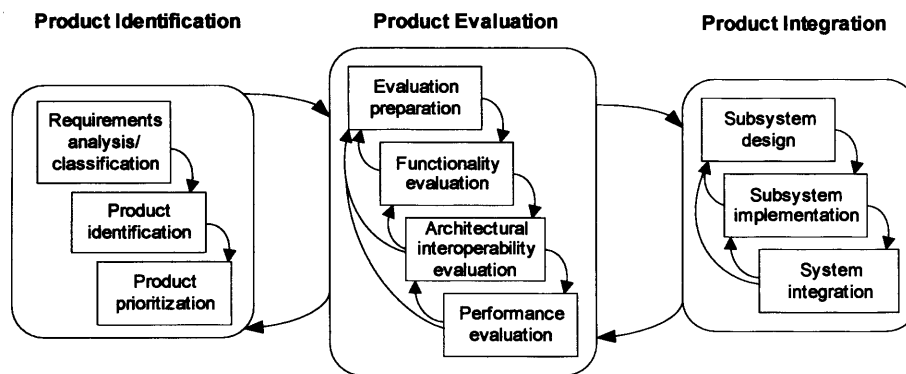


Figure 2.2: CISD COTS-based Process Model

## 2.4 Accommodating Changes

The shift from development-centric towards a procurement-centric approach involves significant changes in the software process model for systems using COTS products [Basili and Boehm, 2001]. The successful development of COTS-based systems demands a new way of doing business [Albert and Brownsword, 2002] where new skills, knowledge, and abilities as well as changed roles and responsibilities are required. [Basili and Boehm, 2001] suggests that personnel capability and experience are key factors affecting the productivity of COTS-based development.

In particular, developers must be able to deal with a new set of problems and situations. In the traditional development paradigm, developers have a fairly controlling role over the development of the new system. Their principal tasks consisted of producing requirements specification, defining architecture and design models, implementing the software system reflecting the requirements and maintaining the system to accommo-

date future changes. In the COTS-based paradigm, a fundamental difference in the role of developer is that they no longer focus on the implementation of the system, instead developers have to concentrate on the activities of product evaluation and integration. When performing these new roles, developers are commonly referred as evaluators and integrators [Torchiano and Morisio, 2004], respectively. Similarly, the role of users suffers significant changes in the development of COTS-based systems. They must embrace the idea that some of their requirements may not be attainable, hence they have to be prepared to undergo an extensive prioritisation and negotiation of their requirements. In addition, users must actively participate in the system development and contribute to the decision-making process.

The COTS-based paradigm involves changes in several spheres of software development, affecting technical aspects as well as business and organizational issues. For instance, during the evaluation of COTS products evaluators have to assess not only the functionality and quality of COTS candidates, other soft issues are also relevant to the decision process. These can range from the cultural transition imposed by the new system to assessing the level of supplier support and negotiating license agreements. In organizational contexts where long-lived, complex COTS systems are used, obtaining a high quality product is as important as having a successful relationship with the chosen supplier. As a result, the decisions made during the evaluation of COTS have to take into account both dimensions of COTS suitability. A direct result of this situation is that evaluators must have the skills to measure the technical competence of COTS products as well as have the competence to assess more subjective issues like the adequacy of supplier relationship. Moreover, evaluators have to deal with issues such as making tradeoffs between available products and requirements, anticipating changes in technology and predicting how products will integrate with the rest of the system [Brereton et al., 2002, Wallnau et al., 2002].

## 2.5 Summary

In this chapter, we have provided an overview of COTS-based development. We have presented a number of definitions available in the literature for the term COTS and contrasted the various meanings of what constitutes a COTS software with existing definitions for the term component. For the purposes of this thesis, the most appropriate definition of COTS is given by [Meyers and Oberndorf, 2001]. COTS cover several classes of software products, largely varying in terms of granularity and scope. Considerable research effort has been spent trying to classify these diverse COTS products.

The development of systems using COTS is considered to be a promising approach to increase system quality as well as to reduce the time and cost of software development. However, the use of COTS brings several challenges and risks that organizations have to manage that were not present in the traditional software development. Two fundamen-



tal issues that organizations have to be able to handle are dependence on supplier and requirements flexibility. The market-driven nature of COTS products profoundly affects the development process. As a result, new approaches and strategies have to be developed to address the specific needs of COTS-based systems. In addition to the impact over the engineering process, the use of COTS imposes changes in the social and organizational spheres as well. As a result, it is necessary to modify standard development practices as well as stakeholders attitude to accommodate the changes imposed by the COTS-based development paradigm.

We have discussed in this chapter that the development process model for COTS-based systems differs from the traditional one. Some activities occur in both development paradigms suffering different degrees of change, while others are particular of COTS-based development. One of the activities exclusive of CBD is the selection of COTS. This is considered to be a core activity of the COTS-based development process model. In the next chapter we discuss in detail the processes and activities involved in the selection of COTS.

## Chapter 3

# COTS Selection

In this chapter we discuss the selection of COTS products and explain the main processes and steps involved in the selection activity. As we have outlined in Chapter 1, one of the objectives of this research has been to develop techniques and processes that can help organizations to select appropriate COTS. To achieve this objective, firstly it is necessary to perform a critical review of existing COTS selection approaches. In the following sections we provide a discussion on prominent approaches that support the selection process. Given the importance of decision-making and risk management processes during the COTS selection, we also give an account of existing techniques in these research areas and describe how they have been applied in the context of COTS-based development.

### 3.1 COTS Selection Approaches

The typical steps involved in the selection of COTS include the identification of COTS candidates available in the marketplace, evaluation of products, decision to select/reject products, and finally acquisition of the best COTS product. Similarly to the traditional software development, where the requirements of stakeholders will drive the implementation of the system; in the COTS-based paradigm, the requirements will drive the selection of COTS software. In particular, competing products are evaluated against the requirements in which the acquirer organization aims to achieve with the new software system [Ncube, 2000]. The successful product is the candidate that sufficiently meets the requirements of the organization. It is important to note that the final COTS-based system is generally the result of several tradeoffs between what the organization aims to achieve (i.e. its requirements) and what is offered by the selected product (i.e. the COTS features) [Alves and Finkelstein, 2003]. It has been widely agreed in the COTS community that requirements engineering is a core activity to ensure the success of the selection process [Maiden and Ncube, 1998a, Chung and Cooper, 2003, Rolland, 1999,

Carney, 1998]. We now describe various relevant approaches that have been proposed in the literature with the purpose of supporting the evaluation process. Whenever possible, we discuss how the requirements process is addressed by the approaches.

The Off-The-Shelf Option (OTSO) method [Kontio, 1996, Kontio, 1995] was one of the first COTS selection methods proposed in the literature. OTSO is a well-defined method that covers the whole selection process. The definition of hierarchical evaluation criteria is the core task of this method. The criteria consists of a set of functionalities, architectural constraints, and organizational needs. The selection activity identifies four different subprocesses: search criteria, definition of the baseline, definition of evaluation criteria, weighting of criteria. A controversial feature of OTSO is the way it deals with quality aspects (e.g. reliability, portability, performance). The methods assume that these are extra factors that may influence the decision but do not necessarily need to be included in the evaluation criteria. This position has been contested by other researchers who argue that properly assessing quality requirement is a fundamental step to ensure the successful selection and integration of COTS [Carvallo et al., 2003, Beus-Dukic, 2000].

OTSO uses the Analytic Hierarchy Process (AHP) technique [Saaty, 1980] to conduct the decision to select or reject COTS products. We discuss AHP and other techniques used to support the decision-making process in Section 3.2. OTSO presents cost models to estimate the cost to buy and integrate each COTS alternative. The approach breaks down the development costs into three main classes: acquisition costs, further development costs and integration costs. Although having proved to be successful in building the evaluation criteria, this method has limitations on how to conduct the requirements acquisition process. The method assumes that the requirements specification already exists and that it will be part of the evaluation criteria. Another problem with OTSO is that it just mentions the possibility of having unrequired features in COTS but does not provide any strategy on how to deal with them. Although suffering from some weaknesses, the OTSO method served as an initial step for further approaches to be developed.

Another important contribution to guide the selection of COTS is the Procurement Oriented Requirements Engineering (PORE) method [Ncube and Maiden, 1999, Ncube, 2000]. PORE is a template-based approach to support COTS selection. The fundamental idea behind the method is the iterative process of requirements acquisition and product evaluation. The iterative process is illustrated in Figure 3.1. At the beginning of the process, few requirements are specified while there is a large number of candidate products. As the evaluation process continues, the features present in the products inform the requirements process in such a way that requirements become more realistic and clearly defined. At the same time, customer requirements narrow down the number of candidate COTS by short-listing products that successfully satisfy core requirements and eliminating non-compliant ones. By applying the templates available in PORE, it is possible to refine the product list until the most suitable product is selected. According to PORE, the successful COTS is the product that best matches the requirements. Given

that templates are derived from empirical studies about current processes and problems encountered during the selection activity, they provide realistic and useful advice to assist the evaluation of COTS.

PORE provides a process advisor that uses situation rules to determine which are the appropriate techniques, methods and tools to support different situations that may arise during the evaluation of COTS. The process advisor includes approaches such as knowledge engineering techniques, multi-criteria decision-making methods, and requirements acquisition techniques. PORE also provides guidelines on how to design test cases with the objective of guiding the evaluation team to acquire more specific information about products functional capabilities and architectural matching. The method suggests the use of fit criteria to determine whether or not a COTS solution satisfies the requirements. The fit criteria is based on traditional multi-criteria decision making techniques such as AHP and WSM. According to PORE, determining the compliance between features and requirements is a fundamental step to ensure a successful product selection. PORE does not, however, sufficiently describe how the compliance process has to be conducted. For instance, it is not clear how the comparison between features and requirements is performed and how to determine the strategy to eliminate non-compliant products from the candidate list. Another serious limitation of PORE is the lack of advice on requirements prioritisation and negotiation in cases where the requirements cannot be satisfied by any available product.

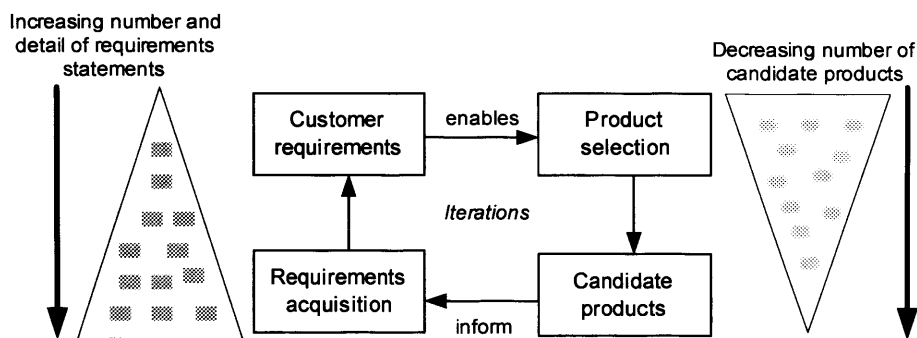


Figure 3.1: Overview of PORE Method (adapted from [Ncube, 2000])

CISD [Tran and Liu, 1997] is a procurement-centric model, it consists of three distinct phases: evaluation, selection and integration. Figure 2.2 illustrates the main steps involved in each phase of CISD model. Tran and Liu describe two forms of COTS evaluation to suit different situations - comprehensive evaluation (CE) and first-fit evaluation (FE). In selection situations with sufficient resources and time, the CE approach is appropriate. In this approach, candidate products are thoroughly evaluated and the result of the evaluation effort is a list of optimal products ranked according to their overall satisfaction to customer requirements. While for situations where it is necessary to make fast and cost-effective selection decisions, the FE approach is more suitable. It simply eliminates products that

failed in a single evaluation stage and selects the first product that succeeds all stages. An obvious limitation of the FE approach is that the selected product is not necessarily the optimal solution.

CISD considers that the three critical areas of the evaluation phase include: functionality, interoperability and architecture, and performance. In the functionality evaluation, the suitability of product features is considered. The interoperability and architecture stage ensures that candidate products are able to be successfully integrated. Finally, the performance evaluation consists of analysis of the overall performance of the integrated system. [Dean and Vigder, 2000] suggests that a key limitation of CISD is that the model assumes a waterfall process model style. This means that each stage depends on the results of the previous stage. We believe that iterative approaches like PORE and OTSO are more appropriate to suit the needs and challenges of COTS-based system development, as we have discussed in section 2.3. Another drawback of CISD is the lack of attention to the requirements process.

COTS-based Requirements Engineering (CRE) [Alves and Castro, 2001] is a selection method that highlights the importance of non-functional requirements as decisive criteria to select COTS. The satisfaction of non-functional attributes is known to be difficult to verify mainly because of their subjective nature [Beus-Dukic, 2000]. Another common problem with non-functional requirements is that they can often interact, such that attempts to achieve one requirement can hurt or help the achievement of another. The method proposes the use of the NFR Framework [Chung et al., 1999] to model non-functional requirements. CRE supports the evaluation of candidate products through the definition of systematic criteria that includes a clear description of quality attributes that candidates have to meet. The method provides specific guidelines on how to acquire and specify non-functional requirements. CRE emphasizes that evaluating and analysing all relevant quality features of COTS candidates requires a great amount of time, typically more than the organization is able to afford. Therefore, it is both necessary and cost-effective to select the most promising candidates for detailed evaluation. One of the key drawbacks of the method is that it is not clear how the product quality is verified. Traditionally these issues are assessed by means of black-box quality testing [Voas, 1998] or acceptance testing [Hausen, 2003]. Another problem with the method is the lack of support to address cases where non-functional requirements are not sufficiently satisfied.

The COTS-Aware Requirements Engineering (CARE) [Chung and Cooper, 2003, Cooper and Chung, 2002] is a goal and agent oriented requirements engineering approach that explicitly supports the use of off-the-shelf products. The approach has been developed following an iterative problem solving strategy. CARE emphasises the importance of keeping requirements as flexible as possible since requirements are likely to be constrained by COTS capabilities. The method classifies requirements as: native (requirements acquired from customers) and foreign (requirements of the COTS components). CARE is based on the  $i^*$  framework [Yu, 1997] to describe the COTS selection process model as well

as to specify the customer requirements. By following  $i^*$ , CARE process model ontology includes actors, goals, softgoals, resources, dependencies and relationships. At this stage, the method only supports the selection phase. As the approach is extended to consider other phases, additional actors, goals, and softgoals can be added to the model. A prototype called CARE assistant tool has been developed as a first attempt to model the interdependencies between native and foreign requirements. The method suggests that bridging the gap between the sets of native and foreign requirements is a core activity of the selection process. Although the approach highlights the importance of mapping system requirements and product specification, it does not provide or suggest any systematic solution to support possible mismatches between both specifications. This research is at an early stage: so far, they have defined the conceptual model of the COTS selection process, but CARE still lacks the definition of precise strategies and techniques to assess the extent to which COTS candidates satisfy the requirements, and therefore allowing the use of the method in practice.

PECA [Santiago et al., 2002] is a COTS selection process developed by researchers from the National Research Council Canada. PECA acronym stands for the four processes involved in the COTS selection: Planning the evaluation, Establishing the criteria, Collecting the data and Analyzing the data. PECA processes are highly interactive and flexible to suit the needs of different organizations. The approach recommends that the evaluation planning to determine the level of rigor of the evaluation effort depends on the severity of risks involved in the project and criticality of strategic objectives. This means that in some circumstances the evaluation must be extremely rigorous while others are successfully accomplished with less rigor.

PECA has been developed based on the ISO 14598 standard, it also adopts a number of well-known techniques to support the selection of COTS. For instance, it suggests the use of GQM technique [Basili et al., 1994] to establish COTS evaluation criteria. PECA highlights that besides analysing how products satisfy usual system requirements, it is also necessary to include architecture/interface constraints, programmatic constraints, operational and support environment into the evaluation criteria. The collecting data step involves the acquisition of relevant information to determine the degree in which products meet the evaluation criteria. PECA does not propose any specific decision-making technique to perform the satisfaction analysis. It mentions the possibility to use weighted aggregation methods. However, no further detail is provided on how such methods can be integrated in PECA process. To support the analysing data step, PECA suggests the use of sensitivity analysis, gap analysis or cost repair analysis. Although these techniques provide elementary support to handle the mismatch between requirements and COTS products. We consider this feature provided by PECA is an advance over other COTS selection approaches. One of the limitations of PECA is the poor definition of the requirements process. The approach simply recommends that the requirements process should be performed in parallel with the evaluation of products. This strategy has been adopted

from PORE method.

Tender management [Ten, 2005] from Telelogic is one of the few commercial tools aimed at supporting the COTS procurement process. The methodology behind tender management has been developed based on best practice across industry and government sectors. Tender management is integrated with Doors requirements management tool, hence allowing useful traceability links between requirements and COTS evaluation criteria. The tender assessment process consists of four phases: preparation, scoring, decision, and completion. For each phase, the roles of people involved in the evaluation process are explicitly defined, facilitating the assignment of responsibilities and the tracking of decisions made through the process. The preparation includes the definition of hierarchical evaluation criteria. Each criterion has a weight associated in order to identify core and distinguishing criteria. The critical criteria will help the early identification of non-compliant tenders. Tender management suggests that evaluators have to define a measurement scale for assessing the satisfaction of requirements, where the scale is simply divided into numerical and symbolic ones. Compared to other selection methods, this procedure seems to be a substantial improvement in the way the satisfaction of requirements is examined. The evaluation of tenders primarily consists of measuring how well each criterion is fulfilled. In order to decide which is the best tender, tender management recommends the definition of precise metrics to assess criterion satisfaction. During this phase, evaluators also prepare and issue the invitation to tender. The next phase consists of analysing the received tenders and scoring each tender. At the end of this phase, scores are aggregated to provide an overall result to inform the decision making process.

The decision phase consists of analysing the results to determine the winning bid. The tool enables the visualisation of the resulting data from different perspectives to facilitate the decision process. For instance, evaluators can check whether the winning product is the better solution in all critical criteria or it is simply marginally better than the other competitors. The sensitivity analysis feature allows evaluators to view the tender's margin which is the difference between the score awarded to a specific tender and the highest score awarded to the rest of the tenders. Finally, in the completion phase the winner bid is notified and contractual and other legal issues are negotiated. Tender management provides many of the benefits of automated tools, such as: a systematic and easy to use methodology, where evaluators just need to fill in information about the decision process. In particular, the mathematics behind the scoring calculation is transparent to evaluators, who do not need to have expertise in decision theory. Besides that, tender management methodology is based on established principles from requirements engineering and decision science. One of the limitations of the tool is the lack of a proper risk analysis strategy. Tender management just allows the storage of identified risks, but it assumes that any subsequent handling of risks is carried out by other systems or processes. Another drawback of tender management is the considerable effort necessary to perform all evaluation stages. In selection situations with limited resources, the tool may be unfeasible to use.

The requirements-driven COTS products evaluation process (RCPEP) [Lawlis et al., 2001] is a formal approach to support the evaluation of COTS. It relies on linear additive weighted summation method (WSM) to guide the decision-making process. RCPEP provides detailed explanation to prepare the evaluation data in order to use WSM. The authors suggest the use of a composite requirements matrix to aggregate scores of product/requirements compliance. The overall product scoring allows the evaluation team to reject poor candidates and pre-select products for further hands-on evaluation. Then, specific scenarios are developed to assess the quality and functional suitability of each product. A key benefit of RCPEP is the explicitly defined assessment process to determine the extent to which COTS products satisfy the evaluation criteria. While its main weakness is the lack of precise guidance on how to derive the evaluation criteria from customer requirements.

So far, we have described approaches intended to support the whole COTS selection process. Several other approaches have been proposed in the literature with the objective of covering more specific aspects of the selection effort. For instance, the CLear And Reliable Information For Integration (Clarifi) approach [Brereton et al., 2002] aims at providing a broker infrastructure to support the use of COTS components available in the marketplace. Clarifi covers four main areas of COTS-based development: component classification, certification, ranking and selection of components, and visualisation of possible solutions. The classification strategy proposed by Clarifi follows the attribute-value strategy to establish the characteristics of components. Within this strategy, each component has a number of properties where each property can be measured in terms of values. This is a typical form of evaluation described in the software measurement area [Fenton, 1994]. The certification step adopts the widely used ISO 9126 quality standard. In Clarifi, the ranking step fundamentally differs from the decision-making approaches used in the previously described selection methods. Clarifi assumes the existence of brokers, who intermediates the interaction between vendors and integrators. While the other approaches assumed that the selection of COTS was performed by the acquirer organization who had to interact directly with the vendors. A very useful feature available in Clarifi is the use of visualisation to guide the comparison among products. Given the objectives of Clarifi to support component brokerage, this approach seems to be more suitable to address the development of systems using fine grained components.

Iusware [Morisio and Tsoukiys, 1997] is a methodology developed with the purpose of guiding the decision making process of COTS selection. The methodology consists of two phases: designing an evaluation model and applying it. The design phase consists of the following steps: identify relevant actors to the evaluation, determine the type of evaluation required (i.e. either formal description of products or ranking of products), define hierarchy of evaluation attributes, associate appropriate measures to attributes, and finally choose an aggregation technique for recommending a product to be selected. The evaluation model proposed by Iusware adapts multicriteria decision aid techniques. We discuss such



decision techniques in further detail in section 3.2. On the one side, Iusware provides a flexible and general approach that enables evaluators with conceptual tools in which they can adapt to solve particular problems. On the other side, the methodology is based on fairly complex mathematical models that may be difficult to be applied by non-experts in decision models. Another weakness of Iusware is the lack of a specific approach to support the acquisition of requirements. The methodology assumes that requirements are already known and they simply need to be organized in the hierarchy of evaluation attributes.

Software evaluation techniques have been widespread in research for some time, much before the increasing popularity of COTS-based development. Feature analysis, for instance, is a generic strategy that has been used to perform comparative evaluation of software engineering methods and tools [Kitchenham et al., 1997]. It involves the identification of requirements that users have for a particular task and mapping those requirements to features of the system. Feature analysis provides a clear methodological framework to guide the evaluation effort. This approach has serious limitations concerning the way it aggregates and consolidates the evaluation results. [Jeanrenaud and Romanazzi, 1904] presents a methodology for evaluating software that employs checklists to determine a quality metric for each item in the checklist. The process is based on metrics to obtain a numerical result that describes the suitability of the software product. This measurement strategy is an effective way to quantify the evaluation results. [Dean and Vigder, 2000] presents arguments showing that this approach is unsuitable to evaluate COTS products, since this methodology was developed with the objective of measuring quality attributes of bespoke software systems, it assumes the source code is available, which is generally not available in COTS products. Procurement strategies have been widely described in the supply management literature [Handfield and Nichols, 1998, Kumar, 2001]. In Traditional domains, supply chain management assumes that the supply chain is fixed and aims to optimize information and material flow based on historical data [Brereton, 2004]. However, such situations do not occur in the highly volatile software market, where products have short and frequent change cycles and companies evolve rapidly.

There has been significant research on acquisition [Kato et al., 2003] and modelling [Finkelstein and Spanoudakis, 1996] of requirements for COTS-based systems. [Lewis and Morris, 2004] describes a number of guidelines and techniques to translate requirements into COTS evaluation criteria. [Rolland, 1999] suggests the use of goal-driven approach to model the impact of COTS products over the requirements engineering process. An impact model is proposed to reason about the impact of changes implied by particular COTS alternatives and how they can be handled in order to successfully integrate the COTS into the organization business process. Goal-driven approaches have also been proposed in [Chung and Cooper, 2003, Rolland and Prakash, 2001, Alves and Castro, 2001] to model the requirements of COTS-based systems. From the perspective of the supplier, the acquisition activity can be supported by development of supplier chain for particular software domains. Sup-

ply chain is an effective way for suppliers to obtain global competitive advantage [Farbey and Finkelstein, 2001]. An important aspect to ensure the success of a particular supplier chain is the coordination of the relationship between suppliers and customers [Brereton, 2004].

A number of experience studies have been conducted to investigate the use of COTS-based systems in different industrial contexts. [Morisio et al., 2000] provides an empirical study of COTS-based development practices adopted in 15 projects from NASA/Goddard Space Flight Center. [Boehm and Abts, 1999] reports some findings on difficulties and lessons learned from COTS integration. [Lauesen, 2004] provides an insightful overview of the COTS tender process based on his experience in developing COTS-based Electronic Patient Recording systems. Some lessons learned from these projects include the observation that tenders generally claim they satisfy all critical requirements to ensure they will be pre-qualified for the trial period, hence customers must be able to identify and deal with false claims. [Finkelstein, 2001] investigates the reasons for the integration failure of a financial system by a higher education institution. The report discusses that the main mistakes made during the selection and integration phases were of organizational nature. In particular, the project team did not pay sufficient attention to analyse the current business process and most of the requirements specified during the project were too specific that managers had difficulty in understanding high level organizational needs. The report also shows that another significant risk was the non-alignment of the system functionality to the user expectations. All these issues suggest that the main reason for the project failure was caused by the poor requirements process.

Another important topic addressed in the COTS literature is the matching problem. This problem has been investigated through two different perspectives: matching between components and matching between requirements and components. The first line of work has been extensively covered by the software architecture research community. [DeLine, 1999], for instance, investigates the sources of component mismatch in the context of components packaging. He provides a catalog of techniques to resolve component packaging mismatch and ensure the successful integration of components within a particular software architecture. The approach concentrates on integration mismatches between middleware components based on standards like CORBA, JavaBeans, ActiveX. [Garlan et al., 1995] discusses that the sources of architectural mismatch are due to a set of assumptions concerning: the nature of the components, the nature of the connectors, the global architectural structure, and the construction process. [Egyed et al., ] also investigates the architectural mismatch among components. The motivation here is that architectural mismatches are caused by inconsistencies between two or more constraints of different architectural parts being composed. The approach consists of high-level component analysis and evaluation of architectural options in order to solve mismatches and consequently, support development decisions. A common assumption behind all these approaches is that components are fine-grained, implementable units. An approach that

fundamentally differs from these approaches to handle architectural mismatch is given by Rolland [Rolland and Prakash, 2001], where a model-based approach is proposed to support the matching of ERP functionality to customer requirements. In this approach, ERP and organizational requirements are expressed using a map representation, where the requirements are represented as the pair of AS-IS and To-Be maps, respectively. The core of the approach is the construction of the matched map that allows the verification of how ERP functionality is aligned to customer requirements. The main limitation of this approach is that it simply supports the identification of mismatches, however it does not suggest how these mismatches can be handled. Rolland's work is closely related to the work we present in this thesis. In [Alves and Finkelstein, 2003], we present preliminary results to address the matching between requirements and COTS features.

## 3.2 Decision Making Techniques

The evaluation of COTS products is considered a form of decision-making where COTS candidates are assessed and ranked according to their relative importance to meet the customer requirements. The generic decision-making process involves the following steps: (1) identify the alternatives, (2) define evaluation criteria, (3) rank the alternatives against criteria. Figure 3.2 shows an overview of the decision-making process in the context of COTS evaluation. It consists of a three level hierarchy representing at the first level, the main goal for the decision making process; the evaluation criteria at the second level; and finally at the third level the alternative COTS candidates to be selected in order to achieve the main goal. The evaluation of COTS can be characterised as a decision problem involving multiple objectives, this type of decision is known in the literature as multicriteria decision aid (MCDA). This is a well established research area that aims to provide quantitative models to support complex decisions. MCDA techniques provide well defined strategies to evaluate and score alternatives. These techniques are based on the notion of underlying preferences.

One of the most widespread MCDA approaches to support the evaluation of COTS products is the AHP [Saaty, 1980]. This technique has been adopted by PORE and OTSO selection methods. AHP is a multicriteria decision technique that can combine qualitative and quantitative factors in the overall evaluation of alternatives. AHP enables decision makers to understand complex decisions by decomposing the problem in a hierarchical structure. Decision makers then make simple pairwise comparison judgments throughout the hierarchy to arrive at overall priorities for the alternatives. [Frair, 1995] suggests the following generic steps to apply AHP to solve a decision problem:

1. Build a decision hierarchy by decomposing the general problem into individual criteria,

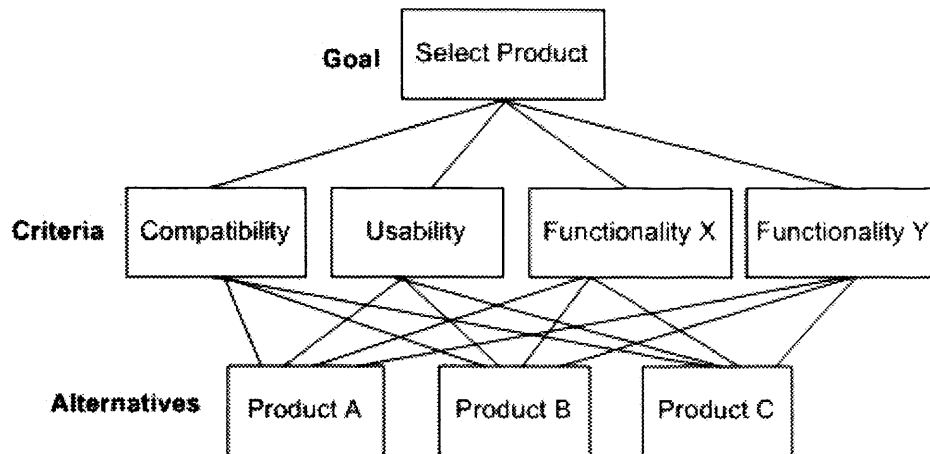


Figure 3.2: Overview of decision-making process

2. Obtain relational data for the decision criteria and alternatives and encode using the AHP relational scale,
3. Estimate the relative priorities of the decision criteria and alternatives,
4. Perform a composition of priorities for the criteria which gives the rank of the alternatives.

The pairwise comparisons between decision criteria uses the fundamental scale illustrated in Table 3.1. This scale allows to express the comparisons in verbal terms which are then translated in the corresponding numbers. [Salo and Hamalainen, 1997, Dyer and Sarin, 1979] have argued that such value comparisons do not represent an appropriate procedure to elicit preference. By using verbal terms, AHP requires decision makers to state their preference in questions like “Which of the alternatives, product A or B, gives the greater performance improvement?” and assuming that the answer is product A “how many times greater is the performance improvement in product A compared to product B?”. To properly answer such questions, decision makers must have a complete understanding concerning the performance of both products and be able to compare their value differences. Given the high level of uncertainty regarding COTS product capabilities, performing such comparisons may be difficult if not impossible. In addition, by simply comparing products in terms of their relative superiority does not necessarily mean that the superior product is good enough (i.e. it sufficiently satisfies customer requirements). Another limitation of AHP is that the pairwise comparisons may become fairly time-consuming if a large number of alternatives and decision criteria need to be evaluated [Kontio, 1995].

Utility theory [Neumann and Morgenstern, 1947, Raiffa, 1982] is another well established

Verbal scale	Numerical values
Equally important, likely or preferred	1
Moderately more important, likely or preferred	3
Strongly more important, likely or preferred	5
Very strongly more important, likely or preferred	7
Extremely more important, likely or preferred	9
Intermediate values to reflect compromise	2, 4, 6, 8

Table 3.1: Fundamental scale for pairwise comparisons

MCDA technique, it is a powerful tool to assist in the process of searching for decisions which best satisfy a multitude of conflicting objectives. Utility theory has been extensively used to solve decision problems in economics and business fields. Although few researchers [Morisio and Tsoukiys, 1997, Giesen and Volker, 2003] have applied utility theory in the context of COTS selection, we believe this is a promising decision technique to help evaluators make rational decisions to choose a suitable product among a set of other COTS alternatives. [Yen and Tiao, 1997] proposes an interesting application of utility theory to analyse the tradeoff between conflicting requirements (we discuss this technique in Chapter 4). The work we present in this thesis has been inspired by this research. A fundamental concept behind utility theory is the decision makers' willingness to maximise overall utility. In the context of COTS evaluation, decision makers aim to maximise quality and other desirable attributes offered by COTS products, while minimising its costs and risks. Obviously, it is not possible to achieve these conflicting objectives simultaneously. Utility theory offers a suitable framework to address situations like this one where decision makers have to structure and value tradeoffs. According to [Raiffa, 1982], in many complex decision problems evaluators face the problem of trading off the achievement of one objective against another objective. Utility functions allow decision makers to represent their preferences and compare the satisfaction (i.e. utility) of different alternatives.

A utility function maps each possible value a decision criterion can assume over a normalised degree of satisfaction. In order to obtain the utility function for a particular attribute, firstly, it is necessary to measure this attribute with appropriate metrics [Fenton, 1994], even when there is no appropriate numerical scale, the measurement is performed at the nominal scale (yes/no; present/absent). The next step consists of establishing decision makers preference over possible outcomes for every decision alternative (i.e. attribute). These outcomes provide the basis for comparison of choices and consequently facilitate the selection of one, satisfactory choice. [Morisio and Tsoukiys, 1997] highlights the importance to distinguish the terms measure and preference. Measure is an objective value assigned to an attribute (this value is usually determined through the use of metrics), whereas preference is a subjective judgement established by the decision maker. For instance, time in seconds is an appropriate measure for the attribute performance, while reducing the time is the decision maker's preference of values for the performance

attribute. To solve a decision problem, measures and preferences have to be obtained for all evaluated attributes. Then, in order to summarise the results in a meaningful way, decision makers have to choose an appropriate aggregation technique. There is a large number of aggregation operators used in utility theory, such as additive, multiplicative, exponential. Because of its simplicity, the additive is the most widely used operator.

A variant of traditional utility theory is the weighted summation method (WSM). According to [Kontio, 1995], WSM can be used to guide the COTS selection in the following way. Decision makers assign numerical weights ranging between 1 and 5 to each requirement present in the evaluation criteria. In addition, each requirement receives another numerical score ranging in the same scale, to represent the degree in which the requirement is satisfied by each COTS product. Although, this technique has the advantage of being very simple to use, Kontio and others [Ncube, 2000, Ncube and Dean, 2002] highlight some problems with WSM that makes this method a rather partial and inappropriate way to compare COTS alternatives. Firstly, the resulting numerical scores can be incorrectly interpreted as if they represent the true differences between alternatives. Secondly, numerical scores are usually imprecise, instead, normalised scores should be preferred in order to compensate the effect of high or low absolute numerical values on each criterion. Thirdly, the direct mapping of numerical satisfaction scores is likely to be biased given the lack of an agreed measure associated to the requirement.

From a completely different perspective, decision models have also been used to support decision problems in the management field. In this area, a well known decision-making model is the rational model [Kreitner and Kinicki, 2002]. This model consists of a problem-solving strategy where decision makers follow four rational stages to reach rational decisions, the stages are: identifying the problem, generating solutions, selecting a solution, and finally implementing and evaluating the solution. This model assumes that decision makers have complete knowledge of all possible alternatives as well the consequences that follow each alternative. In practice, however, fully rational decisions are not always possible because the modelling of complex situations involve a number of uncontrolled constraints and uncertainty. To overcome the limitations of the rational model, [Simon, 1997] proposed the normative decision model to guide the decision process constrained by bounded rationality. The normative model suggests the use of judgemental heuristics to reason under limited information. Simon proposes that decision makers should aim to obtain satisficing solutions, in which the solution is satisfactory and meets a minimum acceptance criteria, but it is not necessary optimal. By following this strategy, decision makers can rely on a more realistic and flexible decision model that is able to deal with uncertain outcomes. Given the limitations and uncertainty involved in the development of COTS-based systems, we consider the idea of achieving satisficing solutions particularly suitable to be the driving force behind the decision-making process.

### 3.3 Risk Management

We have discussed in Chapter 2 that the development of COTS-based systems poses significant risks to acquirer organizations. In addition to the typical risks involved in the development of large software systems, using COTS products brings a new set of risks to the development process, requiring the modification of traditional mitigation strategies and the development of new strategies for risks that are particular to the use of COTS. [Vitharana, 2003] investigates the CBD risks from the perspective of key stakeholders: developers, assemblers, customers. [Rashid and Kotonya, 2001] suggest that many problems involved in CBD are the result of a poor appreciation of the risks involved and lack of strategies to successfully manage them. They discuss that CBD risks are originated from four main factors: the black-box nature of COTS, the quality of COTS, the lack of component interoperability standards, the disparity in the customer-vendor evolution cycles. They also propose a set of intuitive risk management strategies to address frequent CBD risks.

A more formal and comprehensive COTS risk management guide is provided by researchers from the Federal Aviation Administration (FAA) [Shaffer and McPherson, 2002]. Although, most of the risks and mitigation strategies are related to processes and activities performed within FAA, a number of important lessons can be learned from these strategies. For instance, generic strategies like “avoid the modification of COTS products when possible” or “develop and maintain non-technical COTS selection factors” are also appropriate in other domains. Several other works provide risk management strategies to handle risks involved in CBD [Louis, 2003, Li et al., 2004b, Boehm et al., 2003, Engert and Clapp, 2001]. In addition to the agreed importance of handling risks to ensure a successful COTS-based system, some works suggest that risk consideration can aid in determining the degree of COTS evaluation effort as well as choosing appropriate evaluation techniques [Boehm and Port, 2004, Port and Chen, 2004]. An empirical research conducted by Li and colleagues [Li et al., 2004a] suggest that requirement changes and the ability of COTS products to follow these changes are the most frequent risks occurring in COTS-based development. Similar results are described in [Rashid and Kotonya, 2001]. We will explore these risks in further detail in chapter 6.

[Wallnau et al., 2002] proposes a risk-driven COTS selection technique called risk/misfit. This technique is a reinterpretation of MCDA techniques that includes the concepts of utility loss to model the risks and costs involved with the selection of a particular COTS product. Risk/misfit technique exposes and quantifies the necessary tradeoffs involved in the use of COTS by means of cost and risk. It consists of eight well-defined steps describing how risks are identified and quantified, how mitigation strategies are proposed and involved costs estimated, and how to select appropriate risk mitigation strategies. A key shortcoming of risk/misfit technique is the need to estimate the costs involved in each mitigation strategy, for instance, developers should be able to obtain estimate costing to

perform tasks like develop custom wrapper or request vendor to enhance feature. Such information, in many cases, may not be possible to obtain or estimates may not be reliable. As a result, limited benefits may be obtained from the technique.

So far, we have described several research efforts focusing on risks in the context of COTS-based system development. Risk management is a well established software engineering research area and substantial research has been produced in the field. [Boehm, 1991] provides a seminal work on the characterisation of risks that may occur during the development of software systems. The risk management approach proposed by him has been considered a foundation framework in the field. The basic idea behind the risk management strategy proposed by Boehm consists of quantifying the risk exposure of an unsatisfactory outcome. More specifically, risk exposure is the product between the probability of an unsatisfactory outcome and the loss to the parties affected if the outcome is unsatisfactory. In contrast with quantitative risk management approaches, Kontio proposes a qualitative method called Riskit [Kontio, 1997] to support the modelling and handling of risks. Riskit and other risk management approaches provide a taxonomy to classify types of risk and respective management actions [Kontio, 1997, Marvin et al., 1993, Charette, 1990]. The benefits of such taxonomy-based approaches is that they provide a consistent and repeatable framework that can serve as a basis for other risk management strategies and methods to properly cover risks from different domains. Hence, risk management strategies can be improved and derived from previous experience and research. An interesting exercise would be to integrate and extend available risk management taxonomies to incorporate risks that are particular to COTS-based system development.

### 3.4 Summary

In this chapter we have discussed the selection of COTS products. Researchers in the area seem to agree that requirements play a fundamental role in the selection process. In particular, it has been suggested that the selection of COTS should be performed in parallel with the requirements acquisition and modelling. We have presented some important contributions to guide the selection of COTS. Some approaches present in the literature aim to cover the whole COTS selection process, methods such as PORE, OTSO, CISD and PECA provide a systematic and repeatable approach to compare and rank candidate products. A number of other approaches focus on specific aspects of the selection process, instead of covering the whole selection process. Most COTS selection methods assume that the evaluation of COTS candidates is performed against a fixed evaluation criteria, which have been derived from the requirements of the customer organization. Methods like OTSO, CISD and Iusware even assume the existence of requirements, therefore, the requirements process has been poorly supported by such approaches. By assessing COTS candidates against the evaluation criteria, requirements are likely to be “frozen” in the beginning of the evaluation process, which means that either promising candidate products



have to be eliminated because they not meet the stated requirements or that large product modification will be needed to satisfy such requirements.

Another main limitation of proposed selection methods is that they assume the ultimate goal of the evaluation process is the ranking of COTS candidates by using a range of MCDA techniques to support that task. Furthermore, according to these approaches, the evaluation process should finish as soon as the overall ranking among products is obtained, then the best product should be selected (i.e. the product that obtained the higher score). Although, MCDA techniques provide a powerful theoretical model to help comparing alternatives, selection decisions should not rely solely on results from MCDA analysis. Obtaining the ranking among products is an important part of the decision process to select a suitable COTS that meets the needs of the acquirer organization, however, this should not be final step of the selection (as has been considered by many existing methods). By doing that, these approaches have neglected to address a crucial part of the selection of COTS that is the analysis of mismatches that may occur between requirements and COTS features, and consequently, the negotiation of unsatisfied requirements. We believe that such issues have not been properly addressed in the current literature, and consequently further research is necessary.

We have also discussed in this chapter that risk is an important issue to ensure the successful development of COTS-based systems. A number of studies have been conducted aiming at characterising the risks involved in CBD. Few approaches, however, have been proposed with the objective of managing such risks. From the existing risk management approaches, none of them has been fully integrated to the rest of COTS selection activities. Two fundamental reasons for an integrated risk management are: 1) to improve the quality and efficiency of assessments through the simultaneous processes of requirements modelling, COTS understanding and risk analysis; and 2) to provide more coherent inputs to the decision-making process.

## Chapter 4

# Requirements Engineering for COTS-based Systems

In this chapter we discuss the traditional and COTS-based requirements engineering processes. We present the main activities of the RE process. We also describe an important area in the RE that is goal-driven modelling. Next, we present relevant conflict and negotiation strategies that have been proposed in different research fields. Finally, we discuss the main differences between the requirements engineering activities for COTS-based systems and the traditional one.

### 4.1 Traditional Requirements Engineering

Requirements engineering (RE) is the first activity of the software system development process. According to [Sommerville and Kontonya, 1998], the term requirements engineering process refers to *“a structured set of activities which are followed to derive, validate and maintain a systems requirements document”*. This definition provides a pragmatic perspective of the requirements activity by considering the creation of the requirements document at the center of the RE process. Although the technical processes of documenting decisions and modelling requirements play an important role in the engineering aspect of RE, the RE process is also dominated by human, social and organizational factors. [Nuseibeh and Easterbrook, 2000] define requirements engineering as *“the process of discovering purpose for which the systems is intended by identifying stakeholders and their needs, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation”*. This definition emphasises the idea that requirements represent the desire of people affected by the system to be developed. This reflects the human-centred aspects involved in RE. One of the most complete definitions of RE is given by [Zave, 1997]: *“Requirements engineering is the branch of software engineering*

*concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families*". This definition highlights the importance of goals as the driving force for the development of the new system. It also refers to the importance of considering and balancing all factors involved in the RE process.

Requirements engineering is widely recognized to be among the most critical steps of system development [Nuseibeh, 1994, Hull et al., 2005]. In order to implement a system that satisfies the needs of stakeholders, needs must be clearly understood and adequately mapped into requirements specifications. Inadequate requirements engineering has been repeatedly pointed out to be a major source of problems in software development [Sommerville, 2004]. Typically, the main causes of software failure are related to the stakeholder factor, some causes include [Pfleeger, 1998, Maciaszek, 2001]:

1. stakeholder needs are misunderstood or not fully captured,
2. stakeholder requirements change too frequently,
3. stakeholders do not want to cooperate with developers,
4. stakeholders are not prepared to commit efforts to the project,
5. stakeholders have unrealistic expectations,
6. different stakeholder needs will conflict with each other,
7. the system no longer brings benefits to stakeholders.

#### 4.1.1 Traditional RE Lifecycle Process

Different models have been proposed to describe the requirements engineering process. Figure 4.1 shows the spiral model with the main activities involved in the requirements engineering process. The process starts with the elicitation of requirements, then an informal description of requirements is produced. The next step is the analysis and negotiation of requirements, after the requirements have been agreed the requirements document is produced and stakeholders are asked to validate the document. This process is repeated until a decision is made that the requirements document is agreed and accepted by all involved stakeholders. In parallel with all the described processes is the requirements management process, which objective is to manage and keep track of all information and changes made during the RE process. An important step that is not explicitly described in the spiral model of the RE process is the requirements evolution since requirements change during development and evolve after a system has been in operation for some

time [Nuseibeh and Easterbrook, 2000]. It is important to note that the requirements engineering process inevitably varies from one organization to another, and hence, this generic process model has to be adapted and instantiated to address the needs of each organization. We now explore each RE activity in further detail.

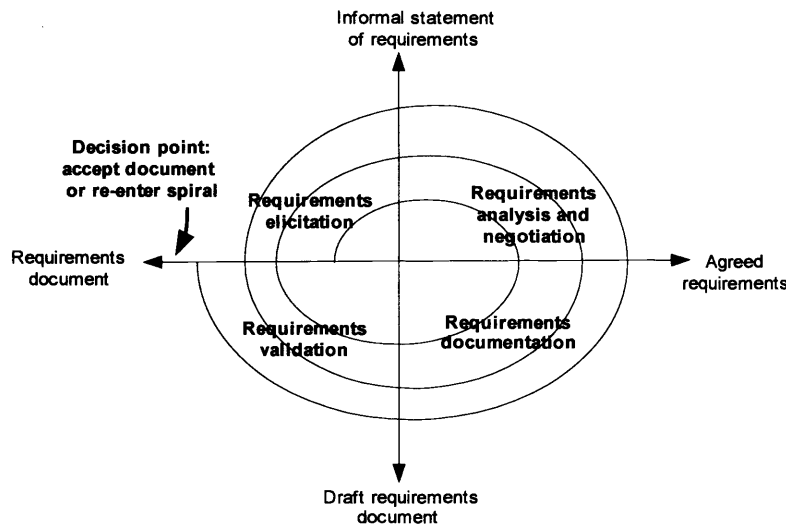


Figure 4.1: Spiral model of the requirements engineering process (adapted from [Sommerville and Kontonya, 1998])

### Requirements Elicitation

The elicitation activity is concerned with the understanding of the problem and organizational context in which the software system will be applied. It also involves the identification of stakeholders and understanding of their goals and constraints. Stakeholders are individuals who are affected by the system. Stakeholders come from different backgrounds, which means that they have very different goals in mind. It is usual that stakeholders may find difficulty in expressing their specific needs or that stakeholders may demand unrealistic goals. It is crucial that the requirements engineer elicit stakeholders' true requirements and be able to reconcile unachievable goals. For a system to bring any benefit to stakeholders, it is necessary their engagement and collaboration in the RE process. The acceptability of the system depends on how well it meet stakeholder needs [Sommerville and Kontonya, 1998]. For all those reasons, the requirements elicitation is a multifaceted and human demanding activity. To facilitate the elicitation activity, a number of techniques have been proposed. The choice of a particular technique should be determined by factors such as the context in which the system will be developed or the type of information that needs to be elicited [Maiden and Rugg, 1996]. A number of elicitation techniques were borrowed from the social sciences, for instance: ethnography, group techniques (e.g. focus groups, brain-

storming). Other common elicitation techniques include: use cases [Jacobson et al., 1999], scenarios (e.g. Inquiry Cycle [Anton, 1997], CREWS [Maiden, 1998]), goal-based methods (e.g. KAOS [Dardenne et al., 1993], NFR [Chung et al., 1999]).

### Requirements Analysis and Negotiation

The analysis of requirements involves two interacting activities. On the one side, requirements engineers focus at building abstract models of the elicited requirements by filling in details and making requirements descriptions more precise and complete. A wide range of modelling methods are available to support the analysis and reasoning about requirements from different perspectives. On the other side, requirements engineers aim to discover possible conflicts and problems with requirements. These conflicting requirements have to be discussed and negotiated to reach an agreement on how requirements can be changed. Conflicts among stakeholders are frequent, in some cases conflicts are even necessary in order to obtain a better understanding of other perspectives involved in the system development in which stakeholders were initially unaware of [Easterbrook, 1991]. Conflicts should be only resolved once all the relevant information is available and it is possible to involve all the relevant stakeholders in the negotiation process. The final requirements specification will be inevitably a compromise among the requirements of different stakeholders. Successful negotiation relies on inter-personal skills and ability to identify common points of view such that individual perceptions and points of view can be brought together. The fundamental idea in negotiating is to avoid a situation in which one side wins and the other loses [Egyed and Boehm, 1999]. An important process to facilitate the negotiation of requirements is the prioritisation of conflicting requirements. Requirements prioritization enables stakeholders to make acceptable tradeoffs among sometimes conflicting goals [Yen and Tiao, 1997] as well as to guide resource allocation based on the requirements importance to the final system [Karlsson and Ryan, 1997]. We are particularly interested in the processes of conflict analysis and negotiation, thus these topics are further explored in section 4.2.

### Requirements Documentation

Requirements documentation is closely related to the requirements analysis activity. Once requirements are agreed, they have to be specified at an appropriate level of detail and formalism. The requirements specification is a channel of communication between stakeholders and developers. Typically, it acts as a contract between both parties. Misunderstandings and errors in the specification will lead to design and ultimately implementation which, while complying with the specification, do not reflect the needs of stakeholders. A good requirements specification must comprise the features of being unambiguously understandable, testable, and modifiable [Hooks and Farry, 2001]. It needs to be unambiguously understandable because it will be read and used by different people from different back-

grounds. It needs to be testable to ensure that the implementation complies with the specification. Finally, it has to be modifiable to accommodate the frequent changes to the requirements.

### Requirements Validation

Requirements validation is concerned with the careful check of the requirements to ensure their consistency and completeness [Sommerville and Kontonya, 1998]. At this stage, the requirements specification must be verified to confirm that it represents an acceptable description of the needs that stakeholders aim to achieve with the system to be implemented. To achieve that, it is necessary to involve relevant stakeholders during the validation process. If stakeholders do not agree with the produced specification it is unlikely that they will accept the final system. Again, in order to ensure that requirements are validated, it is necessary that stakeholders reach a final agreement concerning any conflicting requirement.

### Requirements Management

Requirements management is the process of managing changes to requirements through the system lifecycle. As [Nuseibeh and Easterbrook, 2000] points out managing requirements is not only a process of managing the requirements documentation, it is also a process of recognising change through continued requirements elicitation, reevaluation of risk, and evaluation of the system in its operational environment. Typically changes in requirements are due to the following reasons: errors and inconsistencies in requirements [Finkelstein et al., 1994], new requirements appear as a result of better understanding of the system or because of external changes in the environment, requirements can also be removed during the system development due to costs or schedule constraints [Boehm, 1991]. A major factor to help the management of requirements is the ability of tracing requirements. [Hull et al., 2005] defines requirements traceability as *“the process concerned with understanding how high-level requirements - objectives, goals, aims, aspirations, expectations, needs - are transformed into low-level requirements”*. Several automated tools support the requirements management process, such as DOORS, Requisite Pro, Cradle.

#### 4.1.2 Goal-oriented Requirements Engineering

Goal-oriented modelling is a prominent approach to specify requirements. Goals have been recognized as a leading concept in the RE process [Lamsweerde, 2001] and this area has received increasing attention over the last years. A goal is an objective the system under consideration should achieve. Goal formulations thus refer to intended properties

to be ensured; they are optative statements as opposed to indicative ones, and bounded by the subject matter [Zave and Jackson, 1997]. Several goal-driven RE approaches have been proposed. For instance, the NFR Framework [Chung et al., 1999] provides an explicit representation and analysis of non-functional requirements. In order to carry out this task, the framework first gives the meaning of non-functional requirements as softgoals whose satisfaction cannot be established in a clear-cut sense. The NFR Framework provides a qualitative reasoning procedure for modelling non-functional requirements by determining the degree to which a goal is satisfied/denied. Other features of NFR framework include the ability to explicitly identify interactions between softgoals (e.g. conflict or synergy) and the support to evaluate the impact of decisions. The  $i^*$  framework [Yu, 1997] is a goal and agent-oriented approach to model organizational requirements of social-technical systems.

Another well established research in goal-oriented RE is the KAOS method [Dardenne et al., 1993, Letier and Lamsweerde, 2002]. This approach aims at supporting the whole process of requirements elaboration and modelling. The method consists of identifying and refining high-level goals into subgoals, by representing them in a graph structure inspired on an outer semantic net layer. According to KAOS, a set of subgoals refines a parent goal if the satisfaction of all subgoals is sufficient for satisfying the parent goal. This refinement process continues until subgoals can be assigned to single agents that can be humans, devices and software. KAOS offers two options to modelling goals, they can either be specified using natural language or defined formally in a real-time temporal logic formalism. KAOS provides several contributions to the requirements engineering process. For instance, it supports the identification and management of conflicts between goals [Lamsweerde et al., 1998], and the detection and resolution of exceptional agent behaviour, which are called obstacles that violate the achievement of goals [Lamsweerde and Letier, 2000].

In a more informal way, [Anton, 1997] proposes the GBRAM method to support the identification and refinement of goals into operational requirements. GBRAM provides a number of heuristics and guidelines derived from empirical studies to facilitate the processes of goal identification and refinement. The GQM method [Basili et al., 1994] provides a measurement strategy to assess the satisfaction of goals by asking questions whose answers will tell whether the goals have been achieved or not. According to GQM, the ultimate step of goal-driven approaches is the definition of measures for operational goals. For a detailed survey on goal-driven modelling refer to [Lamsweerde, 2001, Lamsweerde, 2004].

## 4.2 Conflict Management and Negotiation Strategies

A number of studies has highlighted the important role that conflicts play in requirements engineering. Conflict is characterised by a negative interaction between require-

ments, so that a requirement interferes with the achievement of another requirement [Robinson and Volkov, 1997, Easterbrook, ]. Robinson has extensively worked in this area [Robinson and Volkov, 1999, Robinson and Volkov, 1997, Robinson, 1990]. He has proposed the Conflict-Oriented Requirements Analysis (CORA) method to support requirement restructuring by modifying requirements and generating alternatives that remove conflict between stakeholders [Robinson and Volkov, 1999]. CORA explicitly defines an ontology for capturing stakeholder requirements as well as a strategy to restructure requirements as a way to reduce or remove conflicts. The main limitation of CORA is that the restructuring transformations are limited to changes in the definition of requirements and its pre and pos conditions, but the approach does not assess the impact that a single resolution will have over the satisfaction of other dependent requirements. A single resolution may be effective to solve a particular conflict but the overall solution may remain unsatisfactory.

Win Win [Egyed and Boehm, 1999] is a distributed, collaborative framework to negotiate conflicting requirements and investigate architectural solutions. A key principle of the approach is that your project will succeed “if and only if you make winners all the critical stakeholders”. Win Win explores stakeholder interactions, and negotiate mutual agreements on the specifics of the new system being developed. The main purpose of the Win Win negotiation model is to provide a stepwise approach for stakeholders to use in reconciling their individual win conditions. The model includes a tailored domain taxonomy of common requirements conflicts to assist the generation of conflict resolution.

A more theoretical perspective on the conflict management area is given by Easterbrook. In [Easterbrook et al., 1993], he reviews the research on conflict from a social perspective, representing common beliefs and assumptions from various disciplines that study the conflict problem. In [Easterbrook, 1991], a computer-supported framework is proposed to capture multiple perspectives in requirements specification and support the resolution of conflicts that may occur between them. In the same vein, the Viewpoints framework [Finkelstein et al., 1994] has been proposed as a way of managing inconsistent and incomplete information gathered from multiple perspectives. The approach leaves inconsistencies in specifications and use an appropriate logic to continue reasoning, even in the presence of an inconsistency.

[Lamsweerde et al., 1998, Lamsweerde and Letier, 2000] propose a formal framework to tackle various types of inconsistency between requirements. It captures inconsistencies among different stakeholder viewpoints as well as single viewpoint. In [Lamsweerde and Letier, 2000], a particular kind of conflict called divergence is studied in detail. The approach provides formal techniques and lightweight heuristics to detecting and resolving divergences. A key principle of the approach is the management of inconsistencies at the goal level so that requirement conflicts are not further propagated to the system design.



Other streams of research have made substantial progress in the area of conflict management. For instance, the Distributed Artificial Intelligence (DAI) community has extensively investigated the topic over the last years [Russell and Norvig, 1994, Zlotkin and Rosenschein, 1996]. Game theory has been applied to solve conflicts between parties that have opposed or different interests [Petrosian and Zenkevich, 1996]. It provides strategic mechanisms to reach rational and optimal decisions. The conflict management problem has been studied from the perspective of human agents, as well as in computational systems such as knowledge-based systems and multi-agent systems. In DAI area, a common assumption among approaches to handle conflict is that agents have to negotiate conflicting goals in a cooperative fashion.

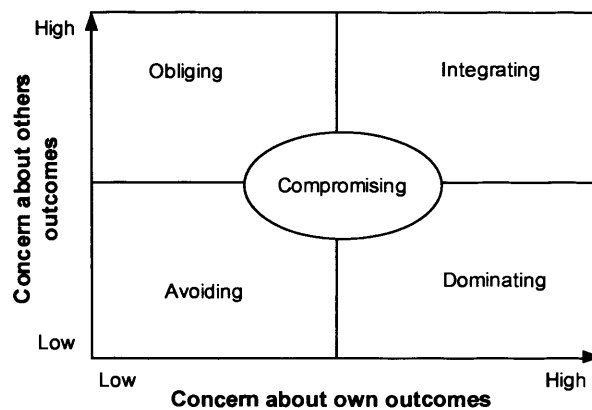


Figure 4.2: Categories of conflict management

From the perspective of organization behaviour theory, conflict has both positive and negative outcomes for organizations, depending on its nature and intensity. According to [Kreitner and Kinicki, 2002], social groups that experience too little conflict tend to suffer from lack of creativity, motivation, decision. On the other side, excessive conflict when not properly managed, can disturb organizational performance because of dissatisfaction, aggressive and stressful environment. As a result, conflict can be distinguished between functional and dysfunctional depending on whether it serves or threatens organization's interests. The aim of conflict management is to control dysfunctional conflicts while stimulating functional conflicts as a stimulus for innovation and development. A well known strategy to manage conflict is to negotiate divergent interests in a give and take decision making process. Parties negotiate because they can use some form of influence to get a better deal than by simply accepting what the other party gives them. [Fisher and Ury, 1992] distinguishes between two types of negotiation - distributive and integrative. Distributive negotiation is typically described as win-lose negotiations, where one party gets what he wants at the expense of the other. Integrative negotiation, on the other hand, is also known as win-win situations where all sides are looking for a solution that maximizes joint gain, it involves looking at the issues being negotiated from multiple perspectives and considering possible tradeoffs (e.g. [Egyed and Boehm, 1999]).

Behavioral approaches [Rahim, 1985] to manage conflicts categorise five different conflict handling styles. These styles are based on desire to satisfy individuals' own concern or to satisfy other parties concerns (see figure 4.2). At one extreme, there is the obliging style when the party neglects his own concern to satisfy the concern of the other. The opposite style to obliging is called dominating in which the party behaves selfishly to achieve his own goals without concern to others. In the integrating style, both parties seek to cooperatively solve the problem by achieving a mutually beneficial solution. While in the avoiding style, both parties agree that the conflict exists, but they decide to passively withdrawal or suppress it. At the center, the compromising style involves moderate concern from both parties such that they make concessions in order to reach a settlement.

### 4.3 COTS-based Requirements Engineering

As we have discussed in Section 2.3, the use of COTS products brings new circumstances to the development process that did not occur in the traditional software development. The successful development of COTS-based systems requires a development process tailored to CBD. In particular, the COTS-based requirements engineering process has some fundamental differences from the traditional RE process [Alves and Finkelstein, 2003]. In traditional software development, the requirements engineering activity basically consists of eliciting stakeholder needs, refining the acquired high-level goals into non-conflicting requirements, and finally validating these requirements with stakeholders [Nuseibeh and Easterbrook, 2000, Sommerville and Kontonya, 1998]. The main objectives of the requirements engineer is to ensure that the requirements specification meets stakeholder needs and represents a concise and clear description of the system to be developed. Broadly speaking, the specified requirements will be translated into software architecture, and ultimately implemented. Therefore, it is reasonable to assert that stakeholder requirements play a controlling role in system development [Wallnau et al., 2002]. In contrast to the traditional development paradigm, requirements engineering for COTS-based development must consider new sources of influence created by COTS marketplace [Alves, 2003]. In this setting, traditional requirements engineering techniques have to be adjusted as well as new ones have to be developed to fit into the marketplace dynamics. We now discuss the impact of COTS over the traditional requirements engineering activities that we have described in section 4.1.1.

#### Requirements Elicitation

The elicitation of requirements for CBD is quite similar to the traditional one. This means that the same types of techniques can be applied to elicit high level requirements from stakeholders willing to use COTS products to build their systems. The main difference now is that requirements should not be specified at a great level of detail, at least during

the first stages of the development process when it is not yet clear which are the functionalities provided by available COTS. Instead, requirements should be specified as flexible as possible in order to increase the number of COTS candidates that might satisfy these requirements [Wallnau et al., 2002]. A further reason for specifying flexible requirements is to avoid situations in which promising COTS are eliminated too early because they don't fit all the requirements. Another fundamental difference between the elicitation process for CBD and the traditional one is that the set of requirements that stakeholders wish to satisfy with the COTS system does not need to be complete. The reason for that is because COTS products are unlikely to meet every requirement of stakeholders. Consequently, it is unwise to spend too much effort and time trying to elicit a comprehensive set of requirements that eventually are not going to be fully satisfied. In CBD, the requirements elicitation process should be done together with the identification of COTS products [Ncube, 2000]. As soon as the core requirements are elicited, the search for possible COTS candidates can start, and the next requirements activities should be performed simultaneously with the evaluation of COTS products. It is sometimes difficult to clearly distinguish the boundaries between these activities.

#### Requirements Analysis and Negotiation

In CBD, the analysis of requirements is a highly interactive and incremental process where the refinement of requirements is driven by the availability of COTS products. More specifically, by observing features present in COTS candidates, stakeholders may recognise new requirements that were not perceived as requirements initially [Finkelstein and Spanoudakis, 1996]. In addition, the examination of product features is a valuable way to refine and better understand how high level requirements can be satisfied in practice. This process can be compared to prototyping techniques that allow users to experiment with the software that will be built to support their work. The main difference here is that users are experimenting the final product rather than a prototype version of the system. The analysis process should focus on refining those requirements that enable the effective discrimination between products [Ncube, 2000]. Discriminating requirements is a fundamental step for assessing COTS products, while requirements that are satisfied by most products are likely to be neglected. As we have discussed in Section 2.2.1, given that COTS are developed to satisfy generic requirements of the marketplace, some requirements of stakeholders may not be satisfied by any available product. As a result, stakeholders have to be prepared to engage in an extensive process of requirements prioritization and negotiation.

In the context of COTS-based development, the activities of prioritizing and negotiating requirements assume an even more crucial role than in traditional system development. Existing techniques to support the negotiation of requirements are no longer valid. Requirements engineers do not have complete control over the functionalities offered by COTS products, and even after stakeholders have reassessed and negotiated their requirements,

there is no guarantee that prioritised requirements will be satisfied by available products. Some requirements may be satisfied by other means such as implementing bespoke parts of the system or acquiring complementary products, however, these solutions simply may not be economically feasible. In order to successfully negotiate their requirements, stakeholders have to perform continuous tradeoffs and analyse risks involved in satisfying a particular requirement against the limitations of available COTS products. Another important step of the requirements analysis is to define requirements as measurable as possible to enable the evaluation of COTS. The reason for that is because it is difficult, if not impossible, to measure the degree in which COTS candidates satisfy each requirement if the requirement is not quantifiable.

### Requirements Documentation

The requirements documentation is a formal connection between the customer and the supplier and it acts as the basis of a contract for the system procurement. In COTS-based development, the requirements documentation will serve as the basis for the evaluation of COTS. The requirements document is generally known as invitation to tender (ITT) or evaluation criteria. The use of ITT document is appropriate for governmental institutions that need a public tender process or organizations acquiring a large and complex COTS product, and hence requiring a fairly formal evaluation process. Normally, the ITT is sent to a reduced number of suppliers who have already passed a pre-qualification process [Lauesen, 2004]. Then, suppliers have to respond the tender stating whether their products satisfy or not the requirements described in the ITT document, after that stakeholders have to check the responses and require selected suppliers to demonstrate their products as a way to confirm they effectively satisfy the requirements. This is generally a long process that requires considerable involvement from stakeholders and potential suppliers. A more informal and straightforward strategy to conduct the evaluation of COTS is simply engaging the relevant stakeholders to assess products against the evaluation criteria without asking suppliers to actively participate on the tender process. It is important to differentiate the ITT and evaluation criteria. The ITT states what is required to be supplied while the evaluation criteria indicate how to compare COTS alternatives.

### Requirements Validation

In CBD, the validation of requirements is the result of a continuous process of COTS evaluation and requirements negotiation in which the final set of requirements will be inevitably a compromise between what stakeholders want to achieve with the new system and what is offered by available COTS products. In contrast to the traditional software development in which the validated requirements should represent the real needs of stakeholders so that the system design and implementation can be conducted based on the requirements document; in CBD, validated requirements are to a large extent the reflex

of the features present in COTS products.

### Requirements Management

The idea that requirements have to be managed through the lifecycle of the system is even more imperative in COTS-based systems. The requirements engineering is no longer regarded as being predominantly an early activity of the system development process. Instead, the requirements engineering is an ongoing and continuous process involving design decisions, tradeoff analysis, negotiation and risk management that will span the entire system lifecycle [Carney, 2002]. Systems using COTS products have their evolution cycles much shorter and frequent than traditionally developed systems. In this setting, requirements change because of two main reasons. Firstly, modifications in COTS products may affect the pre-established requirements. As a result, different evolution cycles between stakeholder requirements and COTS upgrades may lead to a temporary instability in requirements that can be eventually controlled in a later stage. Product modifications may result in severe mismatches between the new features and the stakeholder requirements. Secondly, requirements have to change as a result of limitations in COTS features. For all these reasons, the requirements management process for CBD is also concerned with handling unwanted changes, performing tradeoffs and controlling risks.

## 4.4 Summary

This chapter described the requirements engineering process. Traditionally, the main activities of the RE process are: elicitation, analysis and negotiation, documentation, validation and management. Goal-driven modelling is a prominent area in RE research. As pointed out by [Lamsweerde, 2001], goals capture at different levels of abstraction, the objectives the system under consideration should achieve. Given the known stability of goals, they provide the rationale for requirements in such a way that a requirement represents a particular way of achieving a goal. Goals have been successfully applied to model different classes of software systems. As we have described in Chapter 3, COTS selection methods such as PORE and CARE have explored goal-driven approaches to specify the requirements for COTS-based systems.

We have also discussed in this chapter that negotiation and conflict management strategies are well established topics in the management science, sociology, and distributed artificial intelligence. Some recent requirements engineering research has addressed the problem of understanding and handling conflicting requirements. Important work has been done in the area of requirements negotiation and inconsistency management. Even though, it has been recognised that requirements negotiation is a fundamental part of COTS-based development, little progress has been made towards facilitating the negotiation of

requirements for COTS-based systems.

Finally, we have discussed the key characteristics of the requirements engineering process for COTS-based development. Although the same activities of the traditional RE process are still present, there are important differences between both processes. A fundamental difference is that in CBD the boundaries of the requirements engineering activity are more blurred, the requirements process is performed simultaneously with the evaluation of products and architectural design. Another important feature of the requirements process for COTS-based development is that requirements have to be continuously reassessed, negotiated and modified because of frequent changes in COTS products. These issues bring new challenges to the RE process that did not occur in traditional software development.

# Chapter 5

## Motivation

Developing COTS-based systems requires the investigation of several issues from different disciplines. As we have discussed in Chapter 3, on the one side, the selection of COTS products involves a range of technical issues, such as eliciting and modelling requirements for the new system to be acquired, supporting the decision making process, and managing risks. On the other side, it also encompasses a number of non-technical issues, for instance, managing relationship and agreement with vendors, managing stakeholder expectations, establishing new business strategies. In order to improve the quality of decisions made during the selection of COTS products, acquirer organizations should be provided with processes and strategies to handling technical and non-technical concerns that may arise during the selection of COTS. In this research we aim to address both classes of issues.

The research paradigm adopted in this thesis is the engineering approach. This paradigm typically involves investigating a problem; applying theories, methods and techniques in order to find solutions to the studied problem; and testing the feasibility of the proposed solution. In this chapter, we address the first stage of the engineering paradigm, that of understanding the problem. We also establish the specific objectives of this research. Finally, we present a running example to facilitate the explanation of the solution we propose in Chapter 6.

### 5.1 Understanding the COTS Selection Problem

[Jackson, 2001] stresses the importance to focus directly on the problem before trying to solve it. In this section, we follow this advice by deeply examining the problem we aim to address in this thesis: the selection of COTS products. One of the most effective ways to understand a software engineering problem is to explore it empirically [Dawson et al., 2003]. By following a hands-on problem exploration approach, it

is possible to learn which are the real issues involved in the problem. In addition, it is easier to identify the concerns and difficulties to be addressed in order to solve the problem. We now describe a preliminary case study conducted with the purpose of understanding the COTS selection problem. From the results of the case study we establish a set of requirements that approaches aiming to support the selection of COTS would need to fulfil.

### 5.1.1 AFTN Message Switch Case Study

This initial case study covered the evaluation and selection of an Aeronautical Fixed Telecommunications Network (AFTN) message switch product. This case study was originally proposed by David Bush from the Requirements Engineering Specialist Group of the British Computer Society for an event on COTS procurement. The event was held at University of Central England in April, 2003. During this event four researchers working in the COTS area (including the author of this thesis) were asked to show how methods, techniques and tools that they advocate can be used to address the problems and issues this particular selection problem presents. The case study information consisted of a hypothetical requirements specification and a list of six commercial message switch products. According to the technical background included in the case study: *“The AFTN is a global messaging network. It provides the exchange of messages to improve the safety, regularity and efficiency of international air navigation. Messages exchanged through the AFTN include flight plans, NOTAMs, meteorological messages, distress messages, flight regularity messages, administrative and service messages. Message switching systems provide the switches that route messages around the AFTN to their required destination... AFTN messaging performs an essential role in the provision of an air traffic control service” [Bush, 2003]*

We started this exploratory study by examining the requirements specification present in the case study description. It consists of several requirements ranging from functional and non-functional requirements (e.g. safety, security, performance, and usability) to capacity constraints stating the minimum infrastructure that the selected message switch must meet, testing and deployment requirements, and contractual requirements. Requirements were defined at different levels of detail, for instance, some performance requirements were precisely and quantitatively defined such as: *The system needs to support a message rate of 50 AFTN, CIDIN and TELEX messages per second and provide sustained message input of 40 messages/sec.* These requirements state measurable attributes that allow the objective and immediate assessment of how well candidate products meet them. In contrast, other performance requirements were not sufficiently specified to allow their direct verification. For instance, in order to assess how message switches satisfy the requirement *ensure there is no accumulation of messages*, it is necessary to examine environmental assumptions and determine which is the acceptable number of queued messages. Obtaining such information



to make this requirement more measurable is vital to allow the assessment and comparison of message switches regarding their effectiveness in achieving this particular requirement.

The requirements specification presented safety requirements in a rather generic way, for instance, *the switches must not contribute to an unacceptable level of risk to safety of operations, either by commission or omission*. In addition, the document states that *the fulfilment of these requirements must be adequately demonstrated in a safety case*. Although safety is clearly a critical requirement for AFTN messaging systems, they were not sufficiently described in order to objectively guide the evaluation of how well message switches meet them. To perform an objective evaluation of products, it would be necessary to understand issues such as: what do stakeholders mean by unacceptable safety level? which information should be included in the safety case? We observed that these issues needed to be clarified and refined into more concrete requirements. Ultimately, this refinement process would lead to a better understanding of the safety requirements and facilitate the assessment of message switch products. Given the high importance of safety requirements, the evaluation team must ensure that the successful message switch product sufficiently meets these requirements. This means that safety requirements may act as key factors to discriminate between products.

Once we have carefully analysed the requirements specification, we concluded that information available was insufficient to allow the effective assessment of products. As next step, we decided to start exploring the six message switch products described in the case study document. At first instance, our objective was to obtain more information about the AFTN message switch domain and learn which are the features provided by products. After examining the specification of products available on the Internet, we observed that some message switches assure safety of operations by providing capabilities such as: failover and recoverability, automatically redirecting message traffic and ensuring minimum system availability. With this information in hands, we could delimit the scope of how safety requirements can be met.

At second instance, we performed a mapping between requirements and features provided by products. We decided to build a compliance table organised in the following way: in the rows of the first column were the requirements statements and the other columns presented the six products. Then, for each requirement statement we tried to find related information in the products description. The main sources of information we could obtain were message switches commercial documentation and third-party assessment reports. As this type of information tends to be quite biased and marketing-oriented, we had little confidence that products satisfactorily meet the requirements. In particular, considerable risks may occur if there is insufficient evidence that the chosen message switch satisfies its safety requirements. During the compliance checking exercise, we observed that most message switches being evaluated provide a number of capabilities that were not requested in the requirements specification, such as: system statistics, built-in message templates and keyboard macro facilities.

Even though we had limited information available which means that we were not able to fully verify product features, message switches seemed not to fully satisfy some of the desired functionalities described in the requirements specification. For example, we observed that some message switches did not meet the required CIDIN relay function rate of 50 packets per second at peak time. Moreover, it seemed to be quite difficult to ensure that the following requirement is satisfied: *present information consistently with related systems that users are operating*. Although message switch products belong to a highly specialised application domain in which systems have to comply with rigorous standards and procedures, it may be the case that some promising products do not offer a graphical interface similar to systems users currently operate. An interesting comment is that capacity requirements, such as *support 1900 channels, 1750 circuits and more than 40 direct asynchronous connections, for either CIDIN or AFTN use* cannot be compromised because these requirements state the minimum infrastructure required for the successful message switch.

During this study we also had to deal with situations where it was impossible to verify if requirements were satisfied or not. For instance, the fully satisfaction of requirement *ensure there is no accumulation of messages* was quite difficult to guarantee. In this respect, one particular message switch documentation states that *...Although a single server PC handling message loads above 150,000 messages per day might eventually introduce message delays, the system continues to function in a degraded fashion*. According to the requirements specification, this product capability somewhat differs from what stakeholders wanted. Let us suppose that stakeholders are not entirely satisfied with the server performance offered by this product. In this case, a natural action would be to assess the impact of this mismatch, then either investigate alternative ways to satisfy this requirement or try to negotiate it. In particular, we found necessary to examine issues such as: what does exactly the term degraded fashion of operation mean? how acceptable/unacceptable is the server performance? is the system able to respond to operator commands to alleviate the overload situation? By analysing such issues, we believe that it would be possible to get a better understanding of the mismatch and allow evaluators to handle it.

An important issue we had to consider is that, according to the case study description it is estimated that the current system has 75% chance of failing to cope with traffic next summer (11 months away). Consequently, the message switch selection process had to be performed in a short timescale to ensure that the new message switch system would be fully operational before that. We believe that this is a major risk faced by the project that should be explicitly handled. Possible ways to manage this risk include: (1) providing effective techniques and mechanisms to perform a straightforward evaluation of message switch products and selection of the best candidate, (2) ensuring that the chosen product is fully available, (3) having a close relationship with the supplier.

**Requirements for a COTS selection method**

1. Be systematic, easy to use and cost effective.
2. Support the requirements process.
3. Support requirements elicitation and modelling.
4. Support requirements negotiation.
5. Support decision making process.
6. Identify and handle mismatches between requirements and products.
7. Analyse and manage risks involved with the selection of COTS.
8. Analyse impact of decisions.

Table 5.1: Set of requirements a COTS selection method would need to satisfy

## 5.2 Requirements for a COTS Selection Method

The AFTN case study proved to be a valid exploratory exercise. We have obtained valuable insights of the COTS selection problem from a practical perspective. From the experience gained in conducting this case study, we can present important observations concerning the challenges and issues that need to be addressed during the evaluation and selection of COTS. Although the case study description provided an initial requirements specification, this may not be the case in every procurement situation. In [Ncube, 2000], Ncube reinforces the importance of eliciting requirements and highlights the weakness of other COTS selection methods that wrongly assume requirements already exist. This assumption may imply that requirements are inflexible and static. However, we have extensively discussed in Section 2.2.1 that having requirements negotiable is a fundamental need in the COTS-based paradigm. As a result, we believe that eliciting requirements is a necessary activity of COTS-based development. In addition to that, we observed in the case study that some requirements were specified in an abstract way so that it was difficult to check their satisfaction without further clarification. From this observation, we consider that in order to enable the effective assessment of how COTS products meet a particular requirement, this requirement has to be modelled and refined as a way to make the requirement as measurable as possible.

During this study we performed a mapping between the requirements and the six message switch products being evaluated. Even though we could not verify whether several requirements were satisfied by products simply because we had no information available about these message switch features, we observed that in some cases requirements were not sufficiently satisfied or not satisfied at all. In other cases, products provided features that were not included in the requirements specification. This fact reinforces our earlier assumption presented in Chapter 2 that several mismatches may occur between what cus-

tomers want and what is offered by COTS products and that the successful selection of COTS largely depends on the evaluation team's ability in handling these mismatches. A direct result of these inevitable mismatches is that requirements have to be continuously negotiated to accommodate the limitations of COTS products. Hence, we underpin the need to fully support the requirements negotiation activity during the COTS evaluation and selection processes.

Another important observation from this study is that the selection of COTS products involves not only making decisions to select or reject products, but also assessing the impact of decisions made through the evaluation process. As we have observed during this study, impact analysis seemed to be a valuable strategy to reason about mismatches between requirements and COTS products. Several works have extensively applied decision-making techniques to support the evaluation of COTS [Kontio, 1996, Maiden and Ncube, 1998b]. However, few approaches have supported the reasoning about the impact of decisions [Santiago et al., 2002]. We consider that impact analysis is a fundamental part of the decision making process, and hence it needs to be better supported.

Even though this study had serious shortcomings due to the limited information about the message switches that we were able to obtain. We believe this is not an isolated problem and that real COTS selection projects need to deal with this problem frequently. This lack of appropriate information can bring a number of risks. In addition to that, we observed that several other factors present in the case study could result in risks. For instance, according to the case study description failing to conduct a fast and effective selection of message switch may result in serious danger for the system operation over summer. This is a severe risk that should be carefully addressed. In contrast, other risks faced by this selection project may not have such serious impact, for instance, suppose that none product present information consistently with other systems, this is clearly an unwanted result but it does not necessarily lead to any major risk. This suggests that evaluators may need mechanisms to identify and prioritise risks. As discussed in Section 3.3, we consider that risks have to be properly managed early in the evaluation of products. Other researchers have also observed the need to address risks in COTS-based development [Louis, 2003, Li et al., 2004b, Boehm et al., 2003, Engert and Clapp, 2001].

From the observations obtained with the message switch study together with discussions presented in the previous chapters we established a set of requirements for an approach aiming at addressing the COTS selection problem would need to fulfill (see Table 5.1). It is important to note that we did not include in this set of requirements some rather obvious aspects of the COTS selection process such as acquiring information about COTS products or evaluating COTS candidates. We assume that such activities are ordinary requirements any approach intended to guide the evaluation of COTS must satisfy. In contrast, we focus on aspects of the COTS selection that have been commonly neglected or poorly addressed. We argue that by addressing the set of requirements, a COTS selection method would be able produce better information that will ultimately lead to better decisions to be taken

by organizations procuring COTS products.

### 5.3 Research Objectives

As we have described in Section 1.3, the overall objective of this thesis is to improve the requirements engineering process for COTS-based systems. From discussions presented in Chapter 3, we concluded that existing approaches fail to address core challenges organizations face in selecting COTS products. Thus, we argued that new approaches are needed to properly address these problems. From the findings of the message switch case study, we have established a set of requirements to effectively tackle the COTS selection problem. Given these important results, we state the specific objective of this thesis that is to develop a novel method capable of achieving the set of requirements defined in Table 5.1. In building a COTS selection method, we follow the definition given by [Brinkkemper, 1996] of what constitutes a software engineering method: “*A method is an approach to perform a system development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development activities with corresponding development products*”. The key features of the method we present in this thesis are the following:

**Goal-centric perspective** We aim to support the whole requirements process for COTS-based systems, from requirements elicitation and modelling to negotiation and evolution. To achieve that, we adopt a goal-oriented strategy to specify the requirements for the new system. Our interest in goal-based specifications is centred on the fact that they allow the progressive elaboration of requirements by means of goal refinement, without the need to over-specifying potential alternatives to meet the goals. Based on discussions presented in Chapter 4 concerning the importance to elaborate flexible requirements when developing COTS-based systems, we are particularly interested to address in our method the processes of requirements prioritization and negotiation. In this context, a goal-driven requirements engineering process seems to be specially suitable because it helps to reason about the impact of decisions and facilitates the tradeoff analysis of unsatisfied goals.

**Management of mismatches and risks** Effectively supporting the matching between requirements and COTS products is a core feature of our method. Although the matching analysis is considered a key aspect of the selection of COTS, this area has been largely neglected by existing COTS selection methods. We aim to provide a well defined strategy to identify, analyse and manage mismatches as well as risks that may arise in the selection process.

**Comprehensive guidance for the decision making process** The way most existing approaches support the decision process to select COTS product generally consists of evaluating and ranking alternatives by using quantitative decision-making techniques. These approaches

**Goals for the online bookshop system**

1. The system shall be successfully integrated with Black Books' legacy system,
2. The system shall allow users to search for books using different search criteria,
3. The system shall display full catalogue of books,
4. The system shall provide flexible customer shopping cart,
5. The system shall support safe payment transactions,
6. The system shall be easy to use,
7. The system shall have a good response time,
8. The system shall allow customisation of the transaction process,
9. The system shall keep track of users purchase history,

Table 5.2: Set of high level goals for the online system of Black Books bookshop

partially solve the selection problem as they simply examine the extent to which COTS candidates satisfy requirements, however, they do not give any support to analyse how unsatisfied requirements will affect the decision process. We aim to support the initial decision making process as well as analyse the impact of decisions.

## 5.4 Introducing a Running Example

In this section we describe a simple COTS selection example to illustrate the method we present in the next chapter. Let us imagine a bookshop called Black Books. The bookshop is suffering from customer losses and declining sales over the last months. The company management believes that the company losses are due to competition from web-based bookshops. In order to keep competitive advantage and follow market trends, managers decided to launch an online service. Following a meeting with the board of directors, it was decided that the software solution to provide the web-based functionality would be acquired off-the-shelf. This decision was influenced by the fact that there are various packages available in the market that can deliver the desired functionality in a short period of time. Therefore, Black Books would be able to quickly follow the trend and retain its pool of customers. The board committee appointed the IT manager to be responsible for the selection of the software package as well as other three skilled staff from different backgrounds to compose the evaluation team. After a few meetings, the team agreed on the key goals the successful software product would need to satisfy (see Table 5.2). We believe that the situation described in this example represents usual steps and decisions that organizations would need to face when developing a COTS-based system.

## 5.5 Summary

This chapter has investigated the COTS selection problem in further detail. We have presented an exploratory case study with the purpose of examining the selection of COTS products from a practical perspective. This case study involved the evaluation and selection of an AFTN message switch product. From the experience gained with the study, we have elaborated a set of requirements that a COTS selection method would need to fulfil. As we have discussed in Chapter 3, existing COTS selection methods fail to satisfy some of these requirements. In the next chapter, we present our proposal to successfully meet these requirements: the TAOS method.

## Chapter 6

# The TAOS Method

The objective of this chapter is to describe the TAOS (Tradeoff Analysis for COTS-based Systems) method. TAOS is the method we developed to address the set of requirements presented in Section 5.2. TAOS is a systematic and interactive approach that guides acquirer organizations to evaluate and select large-scale COTS systems. The use of TAOS is illustrated with the online bookshop example. The method is structured in 5 phases as shown in Figure 6.1.

The first phase is the *specification of goals*. This phase starts with the acquisition of high level stakeholder goals that are continuously refined into more concrete and measurable requirements. During this phase, priorities are assigned to goals and interactions between goals are identified. Once the main goals for the COTS-based system are established, the *assess COTS* phase starts. These two phases are highly intertwined and the flow of information between them is bidirectional. The *perform matching* phase examines how well the COTS alternatives meet stakeholder goals. TAOS uses concepts from utility theory [Keeney and Raiffa, 1993] to determine the degree of product satisfaction, and hence help the decision-making process. The next phase of TAOS is called *analyse mismatches and manage risks*. It consists of analysing possible mismatches between goals and features offered by COTS products. This phase also identifies and manages risks posed by COTS candidates. The analysis of mismatches and risks represents an important qualitative strategy to complement the results obtained with the computation of satisfaction scores during the matching phase. By explicitly analysing such issues, it is possible to reason about the extent to which COTS alternatives diverge from the initial goals and how appropriate tradeoffs can be made to accommodate the limitations of COTS. Finally, the last phase of TAOS is called *select COTS solution*. Note that after decisions are made along the evaluation process, the acquirer organization may decide to select several products in order to satisfy the desired functionality. Each phase of TAOS presents a number of techniques, guidelines and templates to assist the selection of COTS.



## 6.1 Main Principles

TAOS method is based on the following principles:

*Systematic and well defined* - The method provides a systematic and well defined approach to examine both technical and non-technical aspects that influence the COTS selection decision-making process,

*Simple* - The method must be simple to use to ensure that the selection of COTS can be performed in a fast and straightforward way,

*Interactive Process* - The phases of TAOS follow an interactive modelling approach. For explanation purposes, each phase is described separately but in practice processes are highly intertwined,

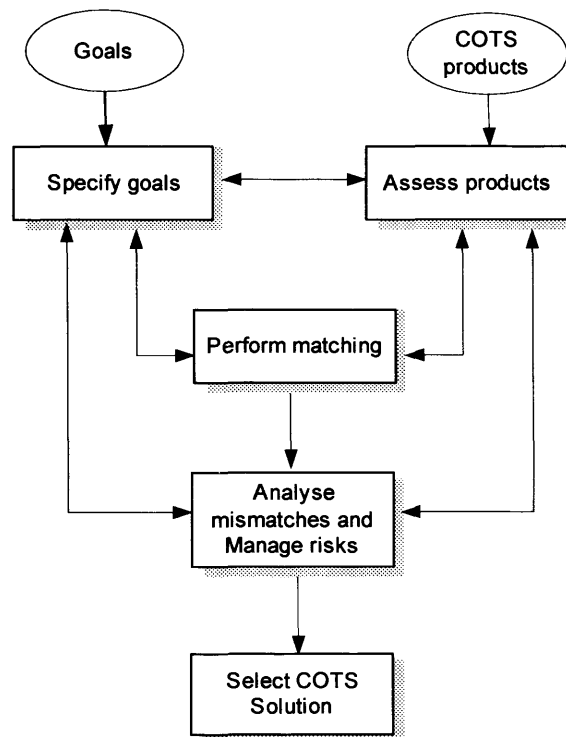


Figure 6.1: Overview of TAOS Method

*Flexible* - TAOS can be adjustable to accommodate different project needs. Depending on the amount of resources allocated for the selection process and the nature of COTS product (i.e. if the product to be acquired is a critical part of the system), different techniques proposed in TAOS can be applied,

*Software Engineering Best Practices* - The development of TAOS was inspired

by software engineering best practices, as described in [Sommerville, 2004, Robertson and Robertson, 1999],

*Goal-oriented requirements engineering* - An important characteristic of TAOS is the adoption of goal-oriented approach. As discussed in Chapter 4, goals provide suitable mechanisms to specify requirements for COTS-based systems,

*Qualitative and quantitative decision-making* - TAOS provides quantitative and qualitative techniques to inform the decision-making process. The ranking of COTS candidates is quantitatively obtained by using typical multi-criteria decision making techniques, while the use of scenarios and heuristics provide a qualitative exploration of the matching/mismatching between stakeholder goals and COTS products.

In the following sections we detail each phase of the TAOS method.

## 6.2 Specify Goals

The specification of stakeholder requirements is considered the first activity of any system development [Nuseibeh and Easterbrook, 2000]. In COTS-based systems, requirements act as criteria to compare and evaluate COTS candidates. TAOS adopts best practices from well established goal-oriented approaches [Lamsweerde, 2001, Chung et al., 1999] to specify the requirements for the system. The use of goal-oriented requirements modelling allows the evaluation team to specify stakeholder needs at different levels of abstraction. Goals can range from high level, strategic objectives (such as “Support secure payment transactions”) to low level, operational concerns (such as “Support Secure Socket Layer (SSL) technology”). An important observation is that higher level goals are more stable than low level ones, and hence are less likely to change over time. The use of SSL technology may be a good way to ensure the security of the system, however, it may impose unnecessary constraints to the solution domain. On the other hand, if the evaluation team simply states that the payment transactions should be secure, it means that other technologies can be used, and therefore the number of possible COTS alternatives that satisfy this goal increases. The decision to specifying generic or restrictive goals depends on the stage of COTS evaluation. Generally, in the beginning of the process generic goals are more suitable, while at later stages, more discriminative goals should be preferred.

Figure 6.2 depicts the six steps involved in the goal specification phase. These steps should be performed in an incremental and interactive fashion and the knowledge obtained in one step may help the evaluation team to better conduct other steps. Note that the goal management activity spans the whole goal specification phase as well as other phases of TAOS. The reason for that is because goals are continuously refined, negotiated and changed during the evaluation of COTS. The goal meta-model is illustrated in Figure 6.3. It involves high level goals that are refined into more objective subgoals until they reach the level of

operational goals. Operational goals can interact between each other. Additionally, each operational goal is associated to a satisfaction function, acceptable interval and weight. In the following sections we describe the six steps of the goal specification phase.

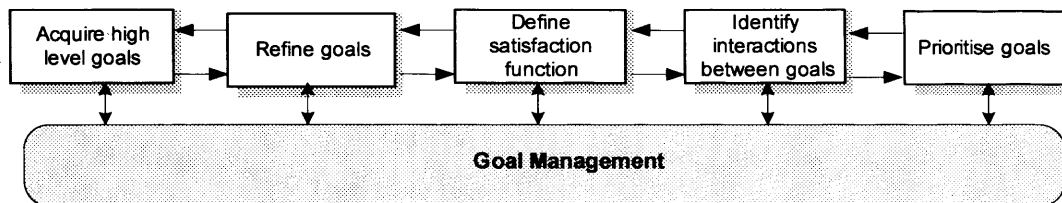


Figure 6.2: Overview of Goal Specification Phase

### 6.2.1 Acquire high level goals

The first step of the goal specification phase consists of eliciting key organizational objectives and constraints as well as understanding the application domain. To obtain high level goals, we can use typical elicitation techniques, such as: interviews, questionnaires, scenarios analysis. Group elicitation techniques, such as brainstorming sessions are quite effective to reach early consensus among stakeholders while obtaining a diverse understanding about the COTS selection strategic objectives. The choice of elicitation technique depends on the resources available and the kind of information necessary to initiate the COTS procurement. As general rule, the elicitation should not last long, as soon as the key strategic goals are obtained and agreed, the refinement of goals should start. In addition, the initial set of goals will guide the definition of the system boundaries and ultimately allow the identification of COTS alternatives available on the market that seems to satisfy the goals.

Given that COTS products may not satisfy every requirement of stakeholders (see Section 2.2.1), it is wise to conduct the elicitation process in such a way that effort is concentrated on specifying truly important goals rather than eliciting an extensive set of stakeholder wishes. A good way to ensure that is to capture the rationale behind each goal along the elicitation process. As we will discuss in Section 6.3.2, the list of goals will compose the pre-qualification questionnaire to be sent to all potential COTS suppliers.

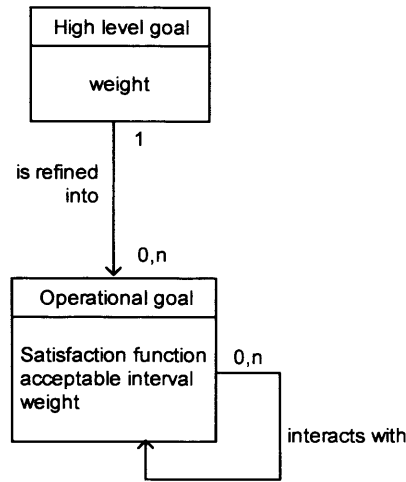


Figure 6.3: Goal meta-model

### 6.2.2 Refine Goals

The next step of the goal specification activity is the refinement of high level goals. The objective of this activity is to represent goals in a more precise and measurable way. Goals are structured in a refinement tree such that high level (or parent) goals are decomposed into more concrete (offspring) subgoals that correspond to richer and more realistic representations of parent goals. The goal refinement tree captures relationship among goals using AND/OR refinement links. AND refinement relates to a goal that is just satisfied when all its offspring subgoals are satisfied. OR refinement refers to a goal that is sufficiently satisfied if at least one of its offspring subgoals is satisfied.

A general guideline for verifying the satisfaction of parent goals is that if a goal is decomposed through an AND link and all subgoals are satisfied then the parent goal is satisfied; otherwise, if at least one subgoal is denied then the parent goal is denied. Similarly, if a goal is decomposed through an OR link and at least one subgoal is satisfied then the parent goal is satisfied; otherwise, if all subgoals are denied then the parent goal is denied. Figure 6.4 shows part of the decomposition of goals for the online bookshop system.

The refinement of goals should be done in parallel with the evaluation of products. The reason for that is because the analysis of COTS features can help stakeholders to clarify vague goals as well as to reveal new goals that were not discovered through traditional elicitation techniques. In order to examine if a COTS candidate satisfies a particular goal, first it is necessary to define mechanisms to measure this goal. The refinement of goals continues until it is possible to objectively measure their satisfaction, at this stage subgoals are called *operational goals*.

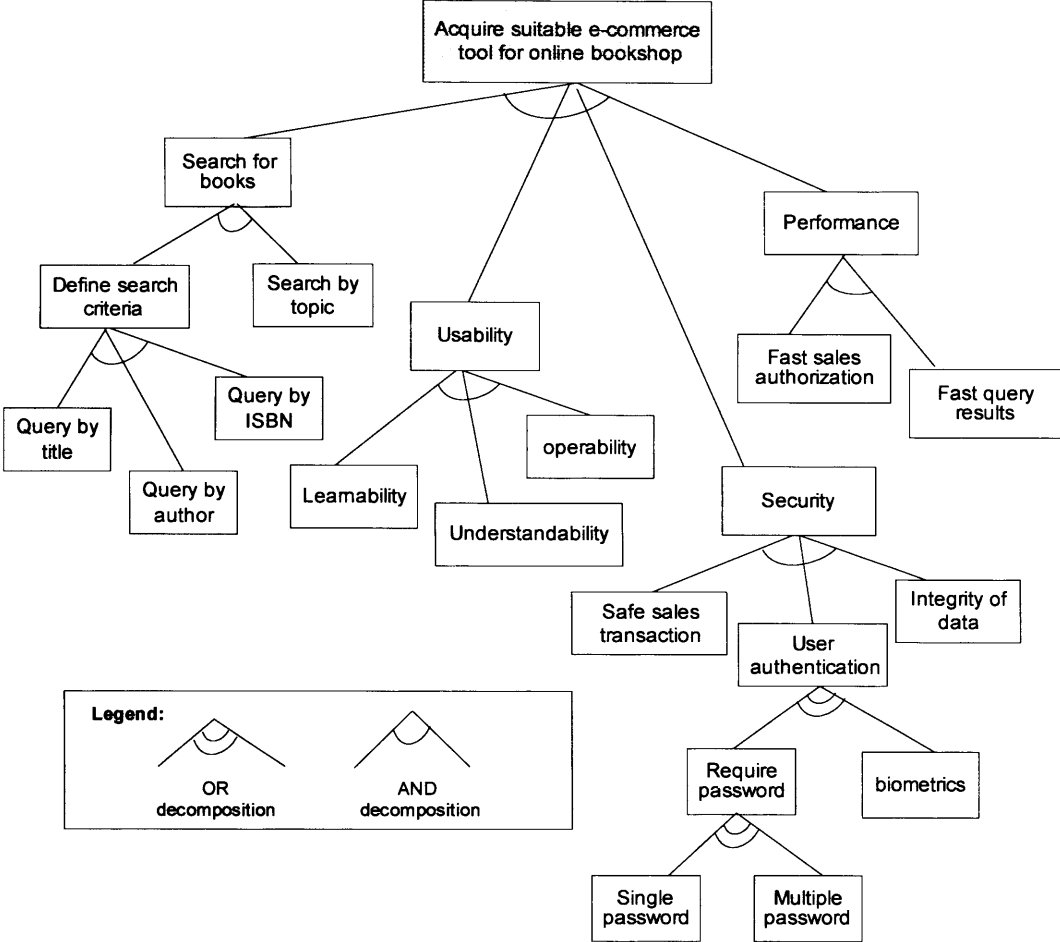


Figure 6.4: Goal decomposition for the online bookshop system

Once we have appropriate measures for operational goals, then we can measure the extent to which COTS candidates match them. To achieve that, metrics are used to quantify the behaviour and quality of operational goals. For instance, in order to measure the learnability of e-commerce packages, we could use various metrics such as: percentage of functions learned, time to learn a task, or numerical rating scale for ease of learning [Dix et al., 1998]. Figure 6.5 illustrates the refinement tree showing that each of these metrics represent different alternatives to measure the learnability of the system. The suitability of each metric depends on the kind of information available during the COTS assessment and of course the amount of effort needed to assess them. The more a goal hierarchy is decomposed and goals become more tangible, the easier is to identify metrics to assess operational goals. Note, however, that the evaluation team needs to balance the benefits to have a comprehensive goal specification against the time and effort necessary to build the tree refinement.

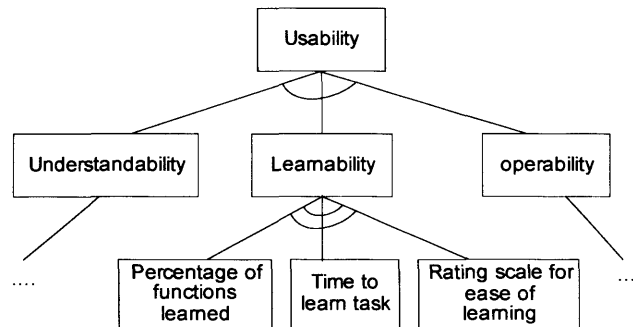


Figure 6.5: Refinement alternatives to measure system learnability

TAOS adopts the work proposed by Fenton to define metrics for operational goals. [Fenton, 1994] classifies metrics into five main categories: nominal, ordinal, interval, ratio, absolute. Table 6.1 shows operational goals associated with possible metrics for each scale of measurement. Generally, numerical scales should be preferred whenever possible as they are the easiest ones to manipulate. Note, however, that some operational goals are subjective by nature and cannot be measured by means of objective, numerical metrics. Consider that we want to measure the reputation of suppliers. One could try to derive reputation in terms of other attributes such as: number of customers in the same business, number of years the supplier has operated on the market, etc. By associating derived metrics to quantify the reputation of suppliers, we obtain more accurate measurements but, as a drawback, the complexity to evaluate the metric can substantially increase. A more lightweight alternative is to measure reputation by using a subjective scale of measurement ranging from 1 to 10, where reputation could be ranked from worst to best along this scale. Such ranking scales can be a useful and simple way to measure subjective operational goals as long as the evaluation team understands their imprecision as the measurements reflect the judgement of the person measuring it. Figure 6.6 provides a number of guidelines on how to conduct the refinement of goals.

Scale of measurement	Example	Type of metric
nominal	automatic e-mail notification	boolean[yes, no]
ordinal	interface configuration	[not configurable, partially configurable, fully configurable]
interval	committed hours of availability	number of hours per month
ratio	failures detected	ratio[number of failures]
absolute	system scalability	number of users simultaneously using the system

Table 6.1: Scales of measurement and possible metric examples

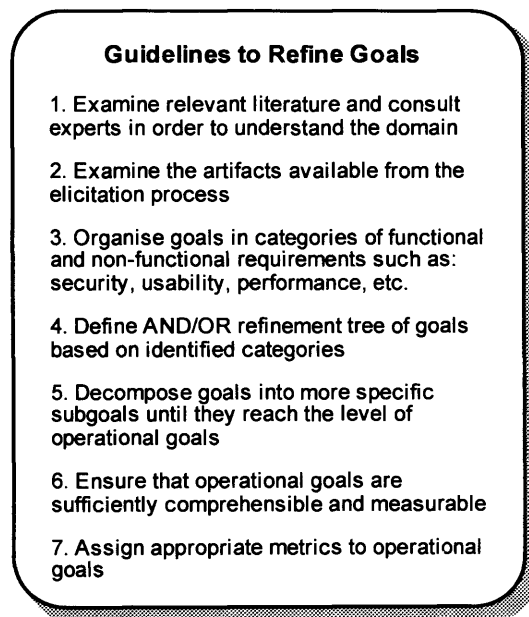


Figure 6.6: Guidelines to refine goals

### 6.2.3 Define Satisfaction Functions

Once goals are refined and appropriate metrics identified, we now examine how the satisfaction of such goals can be measured. Determining the degree in which COTS products satisfy operational goals is a core activity of TAOS method. In order to objectively measure the satisfaction degree of COTS products, we designed a process to obtain satisfaction function of operational goals. This process adopts concepts from decision science, more specifically we employ utility theory [Keeney and Raiffa, 1993] to determine the overall

satisfaction of COTS, and therefore, inform the decision-making process. In the context of COTS evaluation, satisfaction function is a measurement that expresses the relative value in which a COTS product brings towards the satisfaction of goals. In this section we describe a systematic process to define the satisfaction function of operational goals. The process consists of the following steps:

### Step 1. Obtain Acceptable Interval

As discussed in Chapter 4, goals cannot be met absolutely, instead they are satisfied to a certain degree within acceptable limits [Chung et al., 1999, Lamsweerde, 2001]. We define acceptable interval as the range of values in which a COTS product sufficiently meet operational goals. The acceptable interval ranges from the *target level*, i.e. the optimum value that stakeholders consider the goal to be fully satisfied, to the *worst level*, i.e. the lowest value of the acceptable interval in which the goal starts to be considered unsatisfied. For instance, suppose that we want to examine to which extent e-commerce COTS candidates meet the operational goal “fast sales authorization”. A suitable numeric metric to measure this operational goal is time in seconds. We now need to define the interval of values (in seconds) that response time is acceptably satisfied. Let us say that stakeholders are happy to accept values between 10 and 15 seconds, the most desirable outcome is the authorization of sales within 10 seconds, while any value higher than 15 seconds is considered unacceptable. We can use the acceptable interval to evaluate how a particular product (e.g COTS A) satisfies operational goal “fast sales authorization” in the following manner. COTS A is said to satisfy this operational goal if it is able to authorise customer sales between 10 and 15 seconds. If COTS A takes 15 or more seconds to authorise a sale, then the operational goal is considered to be unsatisfied. Now if it takes 10 or less seconds, the operational goal is fully satisfied.

For operational goals associated to numerical scale of measurement, the acceptable interval maps the highest desired value as the target level and the lowest desirable value as the worst level. Similarly, we can define acceptable intervals for operational goals associated to other scales of measurement. For example, as shown in Table 6.1, automatic e-mail notification is measured using a boolean metric. These metrics state the absence or presence of a particular property. Hence, e-commerce products can either satisfy or not this operational goal, which means there are no intermediate values on the acceptable interval. Consider now the operational goal “interface configuration”, which is measured by means of an ordinal metric. The acceptable interval for this operational goal can range from not configurable as worst value to fully configurable as target value, and partially configurable as intermediate value within these limits. Figure 6.7 illustrates possible acceptable intervals for all these operational goals.



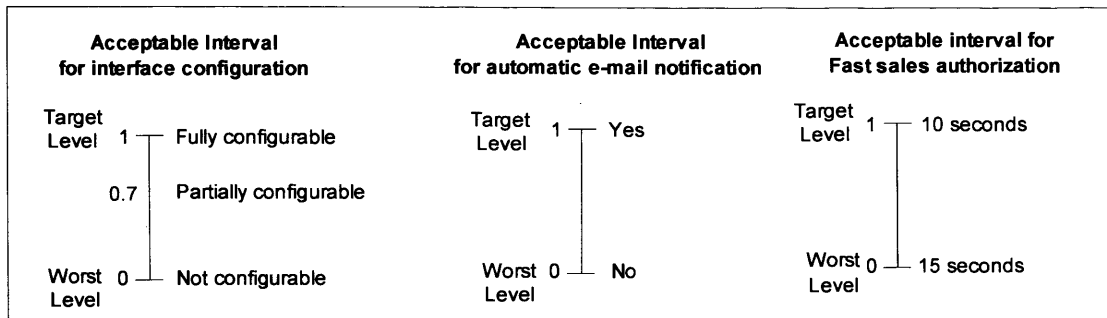


Figure 6.7: Acceptable interval for different operational goals

### Step 2. Decide the structure of satisfaction function

In order to obtain individual satisfaction functions for operational goals, firstly, we need to determine the structure of their satisfaction function according to stakeholders preference towards risk. If stakeholders are risk averse, risk prone or risk neutral towards the expected satisfaction of goals then the satisfaction function is concave, convex or linear, respectively [Keeney and Raiffa, 1993]. For instance, suppose that stakeholders want to obtain the satisfaction function for “fast query results”. After analysing the constraints involved in obtaining a satisfactory response time, they decide that it is preferred to pay more for an additional unit increase of response time if the response time is considered too low, than if the response time is already at a satisfactory level. From this reasoning, it follows that stakeholders are risk averse and that the satisfaction function of operational goal “fast query results” is concave. Consider now that stakeholders are indifferent to an additional unit increase of “safe sales authorization” no matter the current value of this operational goal. In this case, we say that stakeholders are risk neutral and the satisfaction function is linear. When deciding the structure of satisfaction functions, the evaluation team and stakeholders have to keep in mind that eliciting non-linear functions is a complex task that requires the use of specific approaches such as midvalue splitting technique to interpolate the midvalue points of the curve in order to obtain satisfaction functions.

On the other hand, eliciting linear functions is significantly simpler. It is sufficient to have two points (i.e. the target and worst values) to draw a linear function, this process can be viewed as an approximation of stakeholders preference. In our case, it is reasonable to expect that several operational goals need to be assessed, which means that a considerable number of satisfaction functions need to be determined. In this setting, it is clear that linear functions should be preferred because of their simplicity to be obtained and manipulated. It is important to note that linear functions are not always appropriate. The evaluation team has to carefully balance the decision to define linear functions, which means, simplifying the evaluation process at the cost of imprecision. For simplicity, we assume in this thesis that all satisfaction functions have a linear structure. For a detailed

explanation on how to obtain non-linear satisfaction functions, the interested reader may refer to [Yen and Tiao, 1997, Keeney and Raiffa, 1993].

### Step 3. Define Linear Equations

By using concepts from utility theory, we say that the satisfaction function of an operational goal is a mapping from the values of its acceptable interval to the range of normalised satisfaction degrees. Consider that  $x_{target}$  and  $x_{worst}$  are respectively the target and worst values to satisfy  $g_i$ . Then, we have that:

$$\xi \quad Sat_{g_i}(x_{worst}) = 0 \quad (6.1)$$

$$\zeta \quad Sat_{g_i}(x_{target}) = 1 \quad (6.2)$$

$$Sat_{g_i}(g_i) : V \rightarrow [0, 1] \quad (6.3)$$

where  $V$  is the set of values within the acceptable interval of  $g_i$ .

To obtain linear satisfaction functions, we simply use the target and worst levels to determine the function parameters. A linear satisfaction function has the following form:

$$Sat_{g_i}(x_k) = a_k x_k + b_k \quad (6.4)$$

where  $a_k$  and  $b_k$  are constant parameters defined as:

$$a_k = 1/(x_{target} - x_{worst}) \quad (6.5)$$

$$b_k = -x_{worst}/(x_{target} - x_{worst}) \quad (6.6)$$

If the acceptable interval is increasing, in other words if  $x_{target} > x_{worst}$  then the linear equations can be defined as follows:

$$Sat_{g_i}(x_k) = 0 \quad \text{if } x_k \leq x_{worst} \quad (6.7)$$

$$Sat_{g_i}(x_k) = a_k x_k + b_k \quad \text{if } x_{worst} < x_k < x_{target} \quad (6.8)$$

$$Sat_{g_i}(x_k) = 1 \quad \text{if } x_k \geq x_{target} \quad (6.9)$$

On the other hand, if the acceptable interval is decreasing, which means that  $x_{target} < x_{worst}$ , then the linear equations are defined as:

$$Sat_{g_i}(x_k) = 0 \quad \text{if } x_k \geq x_{worst} \quad (6.10)$$

$$Sat_{g_i}(x_k) = a_k x_k + b_k \quad \text{if } x_{worst} < x_k < x_{target} \quad (6.11)$$

$$Sat_{g_i}(x_k) = 1 \quad \text{if } x_k \leq x_{target} \quad (6.12)$$

By solving these linear equations, we can determine the satisfaction function of operational goals. For example, consider again that we want to obtain the satisfaction function of the operational goal “fast sales authorization”. As we have shown in Figure 6.7, the acceptable interval for this operational goal is decreasing and ranges between 15 and 10. Now using equations (6.10), (6.11) and (6.12) we can define the set of linear equations and therefore, elicit the satisfaction function for this operational goal (see Figure 6.8). Note that the described strategy is suitable to obtain the satisfaction function of operational goals associated with a numerical scale of measurement. For operational goals associated with other scales of measurement, it is necessary to determine satisfaction degrees for each measured values. Figure 6.8 shows the acceptable interval and satisfaction function for the operational goal “safe sales authorization”, which is measured by an ordinal metric.

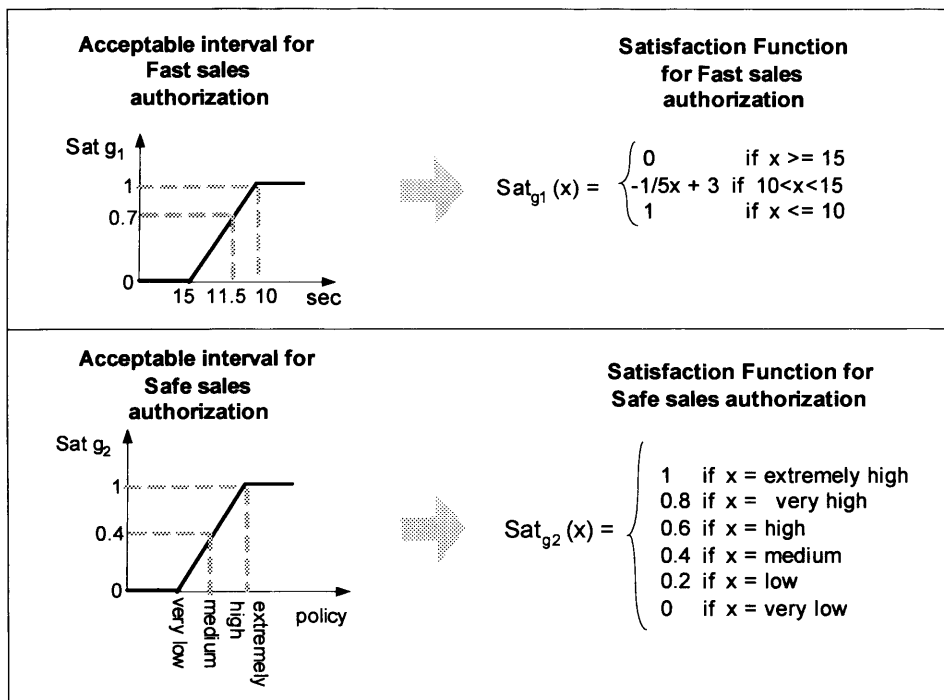


Figure 6.8: Satisfaction degrees for operational goals fast sales authorization and safe sales authorization

### 6.2.4 Identify interactions between goals

As we get a better understanding about stakeholder needs through the process of goal refinement, it is quite natural to find interactions between goals that originally appeared to be unrelated. These interactions normally occur between quality goals (also known as non-functional goals) and can be either positive or negative. A positive interaction, called synergy, occurs when the achievement of a goal contributes to achieve another goal. For example, the ease to use the e-commerce system will contribute to the effectiveness of the

book searching process. This means that the easier to use the system, the easier is for customers to find desired books. Another positive interaction occurs between subgoals understandability and changeability of the system. The ability to change the system is particularly important when developing COTS-based systems since these systems need to be modified and updated frequently. More precisely, the easier to understand the system behaviour and interface interactions the lower will be the effort needed to change the system. Usual changes include elimination of faults, modification of product in order to integrate with other systems, or simply adaptation to comply with organizational changes. It is important to mention that the understandability of COTS products is generally partial given the black-box nature of such components. Therefore, to some extent, the facility to change the system is likely to be compromised. We defined some heuristics to guide the identification of positive interactions between goals:

- Does the achievement of this goal depend on the completion of another goal?
- Does the achievement of this goal increase the satisfaction of another goal?
- Does the failure of this goal cause the failure of another goal?
- What other goal must be achieved in order to satisfy this goal?

Negative interactions characterise conflicting situations between goals. A conflict exists between two goals when one goal interferes with the satisfaction of another goal. Note, however, that not all negative interactions between goals give rise to conflicts. The occurrence of conflict will depend whether stakeholders accept or not the negative interaction. For example, “multiple password” has negative impact over “fast sales authorization” since the time necessary to authenticate “multiple password” will increase the time to authorise customer sales. Depending on how severe is the negative impact of choosing a “multiple password” authentication strategy over “fast sales authorization”, stakeholders may decide if there is a conflict between them or not. The reason for differentiating negative interaction and conflict is because the occurrence of conflict implies that these unwanted situations should be examined and managed, which is not necessarily required for minor negative interactions between goals. To facilitate the identification of negative interactions between goals, we present the following heuristics:

- Does the increase of satisfaction of this goal causes a decrease of satisfaction of another goal?
- Does the decrease of satisfaction of this goal causes an increase of satisfaction of another goal?
- Does the satisfaction of this goal hurt the satisfaction of another goal?
- Does the failure of this goal increase the satisfaction of another goals?

- Does the interaction between goals prevent their mutual satisfaction?

If stakeholders agree on the existence of a conflict, it has to be properly analysed and managed prior the final COTS selection. As shown in Figure 6.9, user authentication can be achieved by means of password or biometrics mechanisms. Suppose that using password to authenticate customers is preferred to biometrics because biometrics authentication is an expensive technique and it is not currently a viable solution in the context of e-commerce systems. Such decisions are typical when goals are decomposed through an OR link, and involve stakeholders having to choose among different implementation alternatives. The next decision step consists of choosing between single or multiple password authentication procedures. Single password contribute positively to the usability of the system, on the other hand, it is not an effective solution to ensure high security of sales transaction because single passwords are quite easy to be intercepted by third parties. While the multiple password option may involve extra effort for customers to log on their account (which may hurt usability goals) this option provides better security than the single password alternative. Once the implications involved with each implementation solution are sufficiently understood, stakeholders are able to better reason about their preferences and decide which tradeoffs they are willing to make. In our example, the tradeoff decision is whether to compromise the security or usability of the system.

Given that COTS products represent possible implementation solutions of stakeholder goals, the decision to select a particular product needs to consider existing conflicts between goals and how each COTS solution influences the resolution of conflicts. Consider, for instance, that COTS A implements single password to authenticate users while COTS B provides a multiple password strategy. As we have discussed, each strategy compromises the achievement of other goals. Therefore, the decision to choose COTS A or B needs to consider the tradeoffs involved with each implementation solution. More specifically, decision makers should examine questions such as:

- Should COTS B be preferred to A because the multiple password solution implies better system security, yet hurts usability and response time to authorise sales?
- Should COTS A be preferred to B because the single password solution contributes to the system usability at the cost of unsatisfactory security?
- How severe is the conflict between each pair of goals? For instance, how much does the multiple password strategy hurt the usability of the system? Similarly, how much does the single password strategy affect the system security?
- What is the importance to achieve each involved goal?
- Which are the risks involved if the conflicting goals are not sufficiently satisfied?

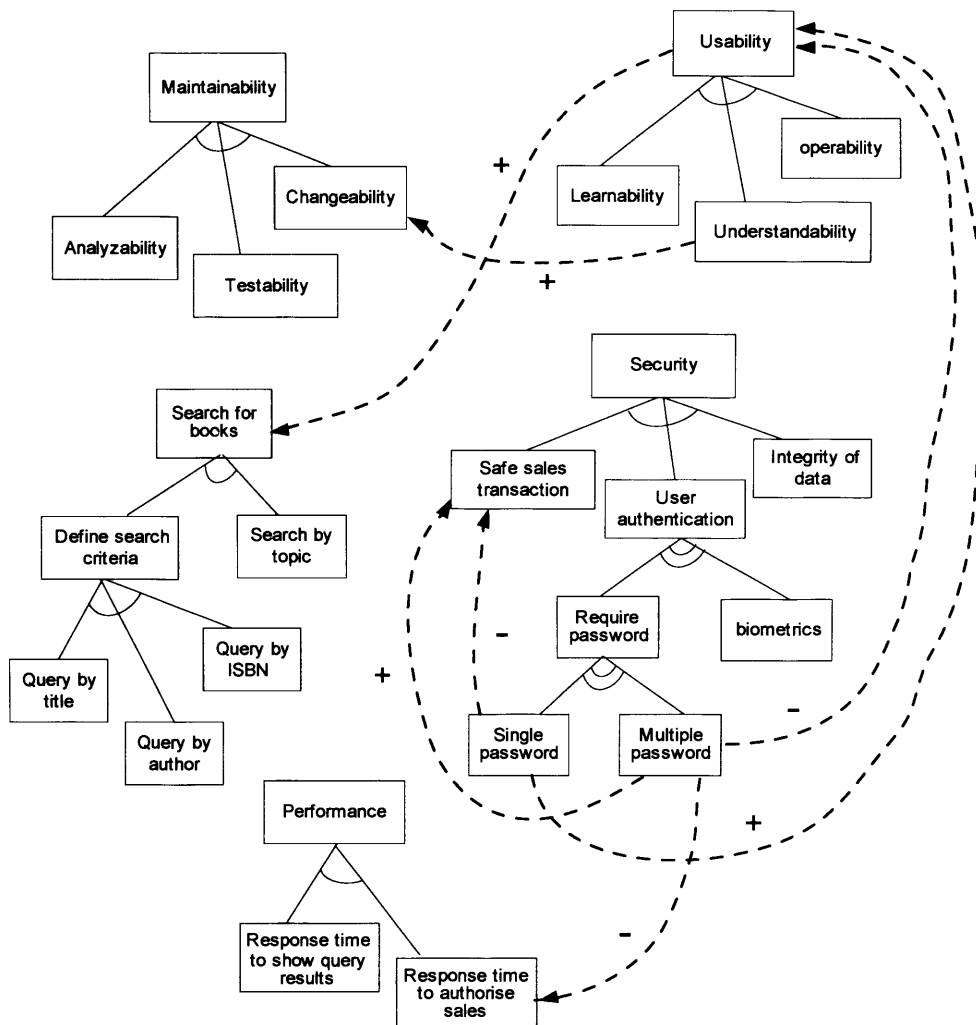


Figure 6.9: Positive and negative interactions between goals

The objective of these questions is to understand the nature of conflicting goals, analyse its implications, and explore associated risks. By answering such questions, stakeholders can reason about the impact of their decisions and explore how possible tradeoffs can be performed. In order to fully understand the impact of tradeoffs, we first need to examine how COTS products satisfy goals. In Section 6.4, we describe a strategy to assess the satisfaction of products, then in Section 6.5 we discuss in further detail the tradeoff analysis.

### 6.2.5 Prioritise Goals

Goal prioritization is a core activity of the requirements engineering process for COTS-based systems. Stakeholders have to engage in an extensive prioritization process in order to distinguish core goals (i.e. critical needs that should always be satisfied) from desirable goals (i.e. the ones that could be traded off). More than simply assigning weights to reflect stakeholder preferences, the prioritisation process should also take into consideration the associated risks and costs to satisfy goals. More specifically, if the evaluation team observes that a goal is not satisfied by any COTS product, then the final priority of this goal should take into account the effort to implement the desired functionality by other means. Another important issue to consider when prioritising goals is the unexpected interaction between the desired functionality and other parts of the system. As a consequence, initial priorities should be continuously reassessed through the COTS evaluation process, in order to reflect constraints and risks as soon as they are identified.

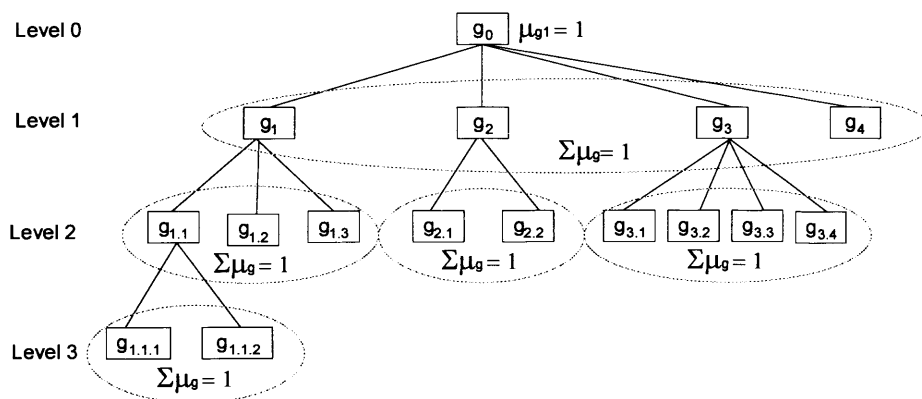


Figure 6.10: Goal Weights

Goal prioritisation is concerned with assigning weights to goals in order to represent stakeholder preferences. We describe an approach consisting of five steps to obtain stakeholder preferences. First let us make some observations. We call the highest level of the refinement tree as level 0, the next level of subgoals is level 1, and so forth. At each level, the

weights of goals from the same parent goal are normalised, so that their sum is 1. In level 0, we describe the business goals that motivated the acquisition of a new software system. The lowest level represents the operational goals. Figure 6.10 illustrates these concepts in the graphic goal hierarchy.

### Step 1. Obtain the relative importance of goals

Consider that we have four goals  $g_1, g_2, g_3, g_4$  and we want to know their relative value. First, rank the importance to achieve each goal. Suppose that stakeholders judge that  $g_1$  is preferred to all other goals. Similarly,  $g_2$  is preferred to  $g_3$  and  $g_4$ . Finally,  $g_4$  is preferred to  $g_3$ . After comparing and obtaining the relative importance between goals, we have that:

$$g_1 \succ g_2 \succ g_4 \succ g_3 \quad (6.13)$$

This would imply that the weight  $\mu$  of each goal is defined as follows:

$$\mu_{g_1} > \mu_{g_2} > \mu_{g_4} > \mu_{g_3} \quad (6.14)$$

### Step 2. Get more refined inequalities between goals

Now we ask stakeholders if their preference to achieve  $g_1$  is higher than to achieve all other goals  $g_2, g_3, g_4$ . If in this paired comparison, stakeholders consider that  $g_1$  is preferred then we can conclude that  $\mu_{g_1} > 0.5$ . If not, continue using the same procedure for the other goals. This procedure only provides inequalities among  $g_i$ . However, if there is indifference among all goals, we can obtain precise numerical weights, so that:

$$\mu_{g_1} \approx \mu_{g_2} \approx \mu_{g_3} \approx \mu_{g_4} = 0.25 \quad (6.15)$$

### Step 3. Assign each goal its weight based on the obtained inequalities

In this step, stakeholders discuss how the satisfaction of each goal contributes to the achievement of strategic objectives. An effective way to accomplish this step is through brainstorming sessions. The main objective of these sessions is to reach an agreement among stakeholders about their preferences and ultimately assign weights to goals. It is important to make sure that stakeholders understand that the reason for prioritising goals is to differentiate core goals from nice to have ones, such that core goals will help discriminating COTS candidates. The rationale behind goal prioritization should be clear. At the end of this step, the evaluation team has to agree with stakeholders on a list of



core goals that should not be compromised. This list will be used to facilitate the analysis of mismatches (see Section 6.5).

**Step 4.** Repeat the initial steps to obtain the weight of goals at levels 1, 2, 3, ..., n.

Now we continue the comparison process by obtaining weights of subgoals belonging to the same parent goal. For instance, it makes sense comparing the relative importance of throughput and response time (both are decompositions of the performance goal) while it might be tricky comparing response time and easy of installation (which is an offspring goal of usability). Note, however, that we compare goals of different types only when a conflict between them is identified. Later, we describe a specific technique to analyse preferences and tradeoffs of conflicting goals.

**Step 5.** Obtain composed weights for offspring goals

The final weight of a goal is called composed weight. It is obtained by multiplying the weight of each goal with the weights of all its parent goals. The reason for this calculation is because the importance to achieve each offspring goal also depends on the importance to achieve its parent goal. Suppose that we have:

$$\mu_{g_0} = 1 \quad (6.16)$$

$$\mu_{g_1} = 0.3 \quad (6.17)$$

$$\mu_{g_{1.1}} = 0.2 \quad (6.18)$$

$$\mu_{g_{1.1.1}} = 0.4 \quad (6.19)$$

Now we can obtain the composed weight  $\omega$  for offspring goals  $g_1$ ,  $g_{1.1}$  and  $g_{1.1.1}$ , so that:

$$\omega_{g_1} = \mu_{g_0} \times \mu_{g_1} \quad (6.20)$$

$$\omega_{g_1} = 1 \times 0.3 = 0.3 \quad (6.21)$$

$$\omega_{g_{1.1}} = \mu_{g_0} \times \mu_{g_1} \times \mu_{g_{1.1}} \quad (6.22)$$

$$\omega_{g_{1.1}} = 1 \times 0.3 \times 0.2 = 0.06 \quad (6.23)$$

$$\omega_{g_{1.1.1}} = \mu_{g_0} \times \mu_{g_1} \times \mu_{g_{1.1}} \times \mu_{g_{1.1.1}} \quad (6.24)$$

$$\omega_{g_{1.1.1}} = 1 \times 0.3 \times 0.2 \times 0.4 = 0.024 \quad (6.25)$$

In a similar manner we get the composed weights for the other goals. Figure 6.11 shows the goal tree graph with respective goal weights  $\mu$  and composed weights  $\omega$ .

This technique assists stakeholders to obtain goal weights by performing pairwise comparisons between goals sharing the same parent goal. The advantage of comparing goals

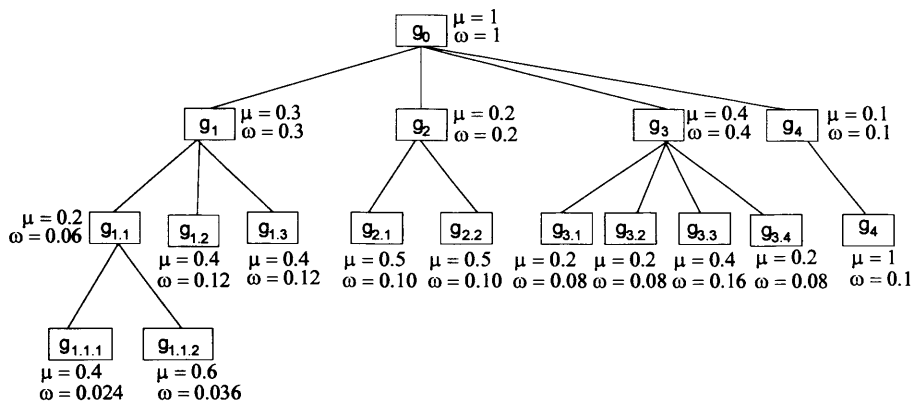


Figure 6.11: Goal weights and composed weights

in pairs is because it allows decision makers to reason about the importance of a particular goal in comparison to other goals, which is a more objective approach than simply attaching numerical scores to goals. In fact, it is more natural and easier for people to provide relative/comparative preference information of the form “I prefer to satisfy  $g_1$  to  $g_2$ ” than absolute information such as “I would like to satisfy  $g_1$  to the degree 0.6”.

The limitation of this approach is that it does not support the analysis of the relative priority of conflicting goals. Therefore, we need a complementary approach to reason about the priority of conflicting goals in order to facilitate the analysis of tradeoffs. From decision science [Keeney and Raiffa, 1993], we use the concept of marginal rate of substitution between two conflicting goals to analyse the relative priority of them. The marginal rate of substitution (MRS) indicates the rate at which stakeholders are willing to sacrifice one goal in exchange for more units of another goal. Consider that goals  $g_1$  and  $g_2$  conflict. If stakeholders are willing to sacrifice a lot in the satisfaction degree of  $g_1$  for a small increase in the satisfaction of  $g_2$ , we can say that  $g_2$  is more important than  $g_1$ . Now suppose that stakeholders are willing to sacrifice very little of  $g_1$  for a little increase in the satisfaction of  $g_2$ , this suggests that the two goals have similar importance. From this intuition, we observe that the relative priority of goals is proportional to the ratio of changes in their satisfaction degree. More formally, the MRS is the rate  $\lambda_{1,2}$  that stakeholders agree to sacrifice of  $g_1$  satisfaction for a unit increase of  $g_2$  satisfaction. Suppose that stakeholders prefer to decrease the satisfaction of  $g_1$  by 0.2 for an increase of 0.1 in the satisfaction of  $g_2$ , then we say that the MRS of  $g_1$  for  $g_2$  is 2 (i.e.  $\lambda_{1,2}=2$ ). Based on the concept of marginal rate of substitution, we now describe a simple technique to obtain the relative priority of conflicting goals.

Step 1. Represent the acceptable interval of conflicting goals as axes X and Y of a Cartesian Graph

Say that we represent the acceptable interval for  $g_1$  in axes X and the acceptable interval for  $g_2$  in axes Y. As Figure 6.12 shows the values along both axes represent the normalised satisfaction degree for each goal.

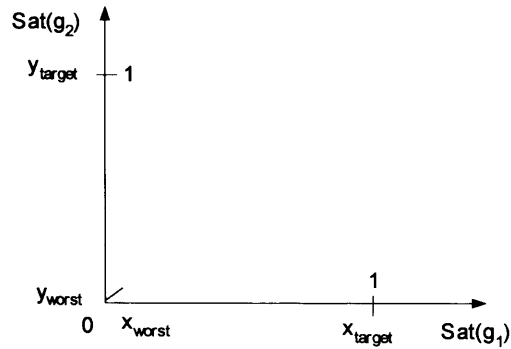


Figure 6.12: Satisfaction degrees of goals  $g_1$  and  $g_2$

Step 2. Hold the satisfaction degree of  $g_1$  fixed and look at the substitution rates as function of  $g_2$

Let us keep the value of  $g_1$  fixed at point  $Sat_{g_1}(x_i)$  and let us obtain the substitution rates for points  $Sat_{g_2}(y_a)$ ,  $Sat_{g_2}(y_b)$  and  $Sat_{g_2}(y_c)$ . These values represent an increasing in the satisfaction degree of  $g_2$ , respectively. Now we need to ask questions such as:

- If we increase the value of  $g_2$  from  $(y_a)$  to  $(y_b)$ , how much are you willing to give up of  $g_1$ ?
- Now, in order to increase the value of  $g_2$  from  $(y_b)$  to  $(y_c)$ , how much are you willing to give up of  $g_1$ ?

It might be difficult for stakeholders to precisely value the amount of  $g_1$  they are willing to sacrifice. In such cases, we can still infer important results by knowing whether the substitution rates increase or decrease with an increase of  $g_2$  values. The following questions facilitate detecting the variation in the substitution rates:

- Do you prefer to sacrifice more of  $g_1$  at point  $(y_a)$  or  $(y_b)$ ?
- Similarly, do you prefer to sacrifice more of  $g_1$  at point  $(y_b)$  or  $(y_c)$ ?
- Finally, do you prefer to sacrifice more of  $g_1$  at point  $(y_a)$  or  $(y_c)$ ?

The variation of the substitution rates indicates an increase or decrease in the relative priority between goals. We developed the following heuristics to interpret the variation

of the substitution rates between conflicting goals, and therefore, facilitate the analysis of how tradeoffs can be performed.

- H1. If the substitution rates decrease with an increase in  $(y_j)$ , then the more of  $g_2$  we have, the less of  $g_1$  we would be willing to give up to gain a given additional amount of  $g_2$ . This means that the relative priority between  $g_1$  and  $g_2$  diminishes as long as  $g_2$  increases, then  $g_2$  becomes less important in comparison to  $g_1$ .
- H2. Now, if the the substitution rates increase with an increase in  $(y_j)$ , then we observe that increasing the value of  $g_2$ ,  $g_1$  becomes less important relative to  $g_2$ , and that we are willing to substitute more  $g_1$  per additional unit of  $g_2$ . Thus the relative priority between the goals increases and it is worth to sacrifice more of  $g_1$  to increase the satisfaction of  $g_2$ .

Figure 6.13 illustrates this situation where the MRS of  $g_1$  decreases with an increase in  $g_2$ , which means that the sacrifice of  $g_1$  is less at  $(y_c)$  than at  $(y_a)$ . Therefore, the relative importance between  $g_1$  and  $g_2$  decreases the more of  $g_2$  you have.

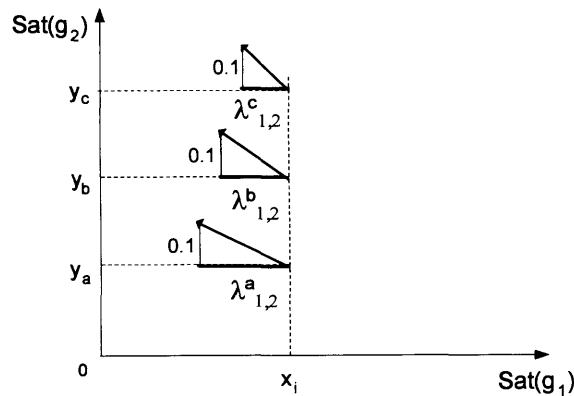


Figure 6.13: Substitution rates for goal  $g_1$

Step 3. Now hold the value of  $g_2$  fixed and look at the substitution rates as function of  $g_1$

In a similar manner, let us keep the value of  $g_2$  fixed at point  $g_2(y_j)$  and obtain the substitution rates for points  $g_1(x_e)$ ,  $g_1(x_f)$  and  $g_1(x_g)$ . These values, in that order, represent an increasing in the satisfaction degree of  $g_1$ . Again we need to ask questions such as:

- If we increase the value of  $g_1$  from  $(x_e)$  to  $(x_f)$ , how much are you willing to give up of  $g_2$ ?
- Now consider that we want to increase the value of  $g_1$  from  $(x_f)$  to  $(x_g)$ , how much are you willing to give up of  $g_2$ ?

If it is not possible to obtain precise amount of how much stakeholders are willing to sacrifice of  $g_2$ , we can ask the following questions to understand the variation of the substitution rates.

- Do you prefer to sacrifice more of  $g_2$  at point  $(y_e)$  or  $(y_f)$ ?
- Similarly, do you prefer to sacrifice more of  $g_2$  at point  $(y_f)$  or  $(y_g)$ ?
- Finally, do you prefer to sacrifice more of  $g_2$  at point  $(y_e)$  or  $(y_g)$ ?

The same heuristics described in step 2 can be applied here. Of course, now we are exploring the substitution rates of  $g_2$  as a function of  $g_1$ , in other words we want to obtain the amount we are willing to sacrifice of  $g_2$  for an unit increase of  $g_1$ .

Figure 6.14 depicts the substitution rates of  $g_2$  as a function of  $g_1$ . Here  $\lambda_{2,1}$  is bigger at  $(x_f)$  than at  $(x_e)$ , and bigger at  $(x_g)$  than at  $(x_f)$ . This means that we are willing to sacrifice more of  $g_2$  to increase even more the satisfaction of  $g_1$ , and therefore  $g_1$  becomes more important in comparison to  $g_2$  as the satisfaction of  $g_1$  increases.

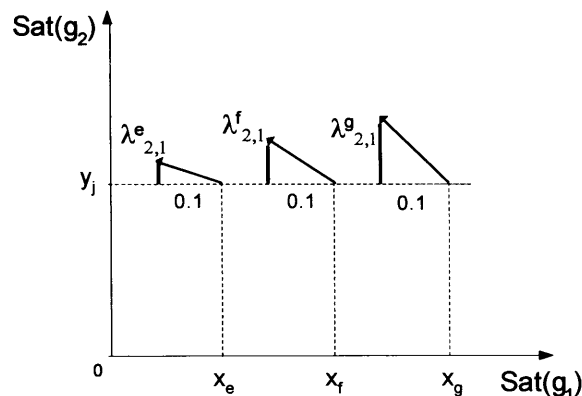


Figure 6.14: Substitution rates for goal  $g_2$

#### Step 4. Reassess the initial priority given to conflicting goals

Once we understand how the marginal rate of substitution varies when changing the values of  $g_1$  and  $g_2$ , we can now analyse the importance of achieving a goal whose satisfaction is harmed by another goal. At this moment, stakeholders are asked to reassess their preferences and balance initially ambitious goals.

To illustrate how this technique can be used in practice, consider we have identified a conflict between goals “fast sales authorization” (i.e.  $g_1$ ) and “safe sales authorization” (i.e.  $g_2$ ). The satisfaction degree and acceptable interval of both goals is illustrated in Figure 6.8. Following the described steps to obtain the relative importance between goals,

we start by drawing the acceptable interval of each goal in a cartesian graph. Then, let us fix the value for “fast sales authorization” at level 0.7 (i.e. by authorizing sales in 11.5 seconds) and observe how much we want to sacrifice the performance of the system for an increase in the security policy. Let us consider satisfaction degrees for “safe sales authorization” varying from 0.2, 0.4 and 0.6 (i.e. the security policies are low, medium and high, respectively). Note that in this case the satisfaction degree of  $g_2$  varies 0.2, instead of 0.1 as for the general case, because the goal “safe sales authorization” is measured by means of an ordinal scale which satisfaction degree is mapped varying 0.2 units for each policy level. Now let us consider that stakeholders have the following tradeoff preferences:

- If the security policy is at low level, we are willing to sacrifice 0.2 degrees of the sales authorization response time (i.e.  $Sat_{g_1}(12.5) = 0.5$  and therefore authorizing sales in 12.5 seconds) to increase the security policy from low to medium.
- Now if the security policy is at medium level, we are quite happy with that and we just want to sacrifice 0.1 degree more of the sales authorization time (i.e.  $Sat_{g_1}(13) = 0.4$  which means authorizing sales in 13 seconds) to increase the security policy from medium to high.
- From this point, stakeholders are reluctant to continue sacrificing the sales authorization response time for an increasing in the security policy and decide that as long as the security policy is at least high, they are not willing to sacrifice more of the sales authorization response time (i.e. sacrificing the time to authorize sales from 11.5 to 13 seconds is the maximum they pay to increase the security policy from low to extremely high).

Figure 6.15 shows how the MRS of goal “fast sales authorization” varies according to different levels of security policy. The MRS is 0.2 at level  $g_2(low) = 0.2$ , then it diminish to 0.1 at levels  $g_2(medium) = 0.4$  and  $g_2(high) = 0.6$ . Finally the substitution rate is zero for values of  $g_2$  higher than  $g_2(high) = 0.6$ , which means that if the the safe policy is at least high, stakeholders are satisfied and are not willing to compromise the response time any more.

The previous technique covered the general case in which the marginal rate of substitution depends both on the levels of  $g_1$  and  $g_2$ , however, there are three special cases to be also considered:

1. The MRS does not depend on the values of  $g_1$  and  $g_2$ . That is,  $\lambda$  is always constant for any value of  $g_1$  and  $g_2$ ,
2. The MRS depends on  $g_1$  but not on  $g_2$ . That is, the amount stakeholders are willing to pay in  $g_2$  for an additional level of  $g_1$  depends on the level of  $g_1$  but not on the level of  $g_2$ ,

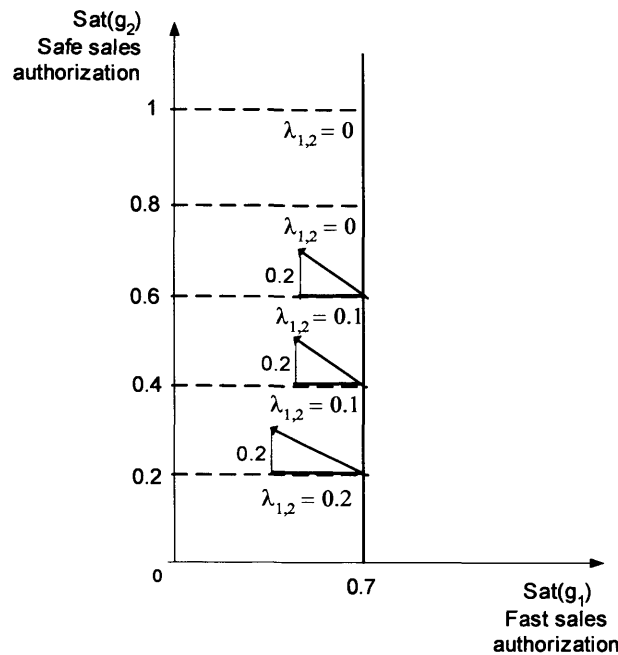


Figure 6.15: Substitution rates of sales authorization time

3. The MRS depends on  $g_2$  but not on  $g_1$ . That is, the amount stakeholders are willing to pay in  $g_1$  for additional  $g_2$  depends on the level of  $g_2$  but not on the level of  $g_1$ .

As we will see later on, a very important result can be obtained from these cases. Consider four points A  $((x_1), (y_1))$ , B  $((x_1), (y_2))$ , C  $((x_2), (y_1))$ , and D  $((x_2), (y_2))$  as shown in Figure 6.16. Suppose the following equivalences hold:

1. At A an increase of b in  $g_2$  is worth a sacrifice of a in  $g_1$ ,
2. At B an increase of c in  $g_2$  is worth a sacrifice of a in  $g_1$ ,
3. At C an increase of b in  $g_2$  is worth a sacrifice of d in  $g_1$ .

If at D an increase of c in  $g_2$  is worth a sacrifice of d in  $g_1$ , then we say that the corresponding tradeoff condition is satisfied and we have the following result:

*Theorem 1* A satisfaction function is additive if and only if the corresponding tradeoff condition is satisfied.

For a detailed demonstration of the previous theorem refer to [Keeney and Raiffa, 1993]. If theorem 1 holds, the overall satisfaction function is additive and can be aggregated by

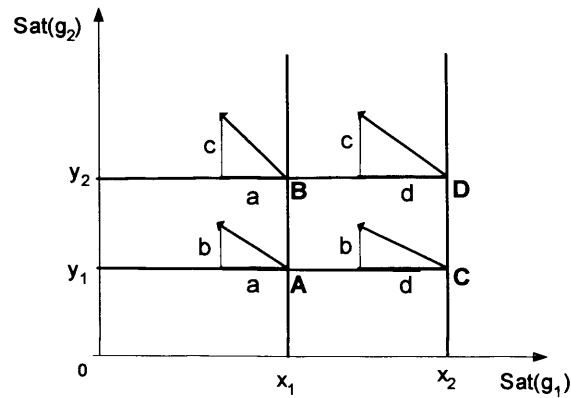


Figure 6.16: Corresponding Tradeoff between  $g_1$  and  $g_2$

using the weighted summation operator. Let us consider the global satisfaction function of all operational goals  $Sat(g_1(x_1), g_2(x_2), \dots, g_n(x_n))$ , where  $g_i(x_k)$  represents the satisfaction degree  $x_k$  of operational goal  $g_i$ . Then the additive satisfaction function is given by:

$$Sat(g_1(x_1), g_2(x_2), \dots, g_n(x_n)) = \sum_{i=1}^n w_{g_i} \times Sat_i(g_i(x_k)) \quad (6.26)$$

where  $w_{g_i}$  is the weight of operational goal  $g_i$  and  $Sat_i(\cdot)$  is the satisfaction function of a single operational goal ranging from 0 to 1.

As we will discuss in Section 6.4, this is an important result to aggregate the overall satisfaction of COTS candidates. It is important to mention that apart from the additive operator there are other aggregation operators, such as multiplicative and exponential. The additive operator is the simplest one to manipulate, however, the use of the additive operator is only valid if the corresponding tradeoff condition holds [Yen and Tiao, 1997].

In our example, let us examine how the corresponding tradeoff condition would be satisfied. Consider that we fix the value of  $g_1$  at degrees 0.5 and 0.7, and the value of  $g_2$  at levels 0.4 and 0.8. Then we have four points:

$$A(0.5, 0.4), B(0.5, 0.8), C(0.7, 0.4) \text{ and } D(0.7, 0.8)$$

Therefore, the corresponding tradeoff condition would be satisfied if we had the following situation (see Figure 6.17):

- If at level A, an increase of 0.1 of  $g_2$  satisfaction is worth a payment of 0.3 in  $g_1$ ,
- If at level B, an increase of 0.2 of  $g_2$  satisfaction is worth a payment of 0.3 in  $g_1$ ,



- If at level C, an increase of 0.1 of  $g_2$  satisfaction is worth a payment of 0.4 in  $g_1$ ,
- If at level D, an increase of 0.2 of  $g_2$  satisfaction is worth a payment of 0.4 in  $g_1$ .

We can interpret the tradeoff condition as follows:

- If we hold the values of goal “safe sales authorization” fixed, then its MRS is constant for any value of goal “fast sales authorization”,
- Similarly, if we hold the values of goal “fast sales authorization” fixed, then its MRS is constant for any value of goal “safe sales authorization”,

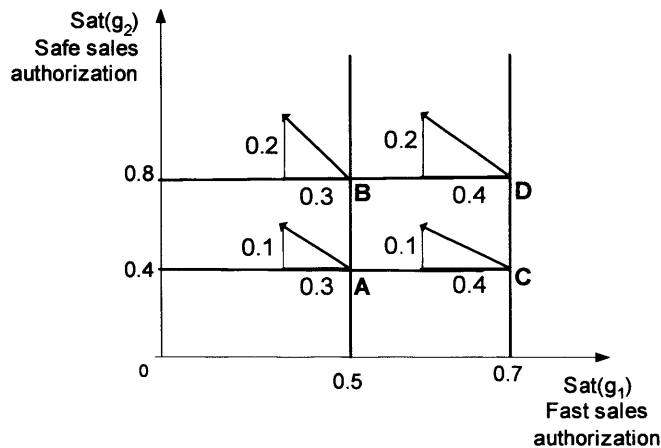
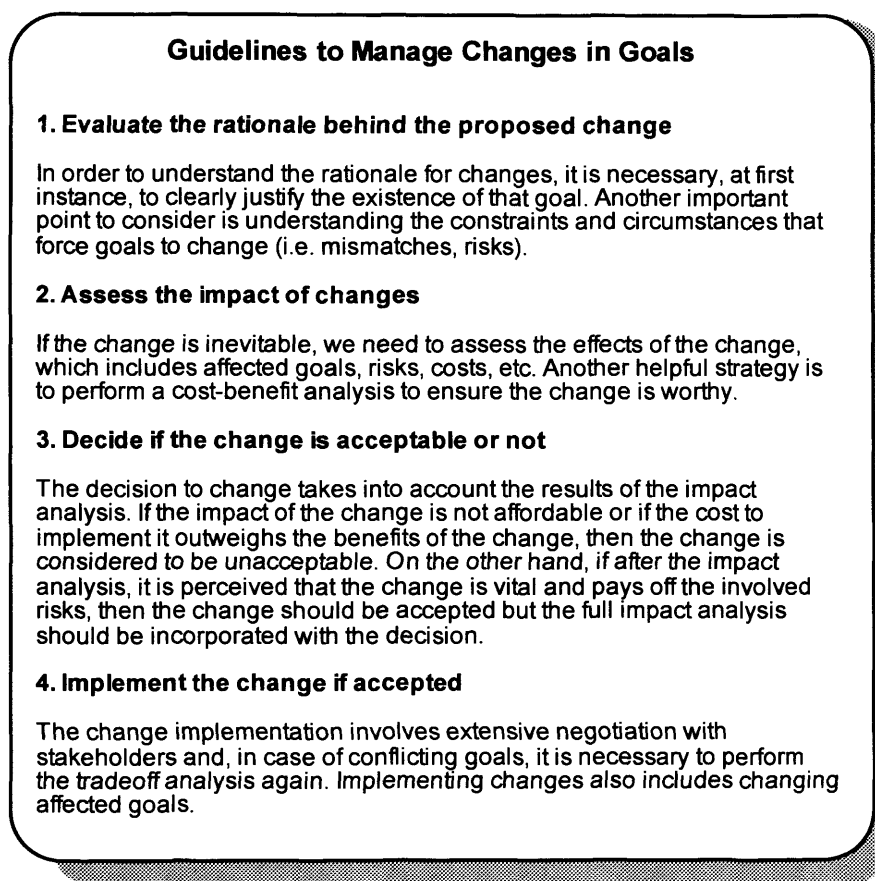


Figure 6.17: Corresponding Tradeoff between safe sales authorization and fast sales authorization

From this example, we can say that if we hold the satisfaction degree of a goal then its MRS is always constant. In the remainder of this thesis, we assume that corresponding tradeoff condition is always valid. The technique presented above examines how tradeoffs can be performed by exploring the amount stakeholders are willing to sacrifice of the satisfaction of one goal to increase one unit in the satisfaction of another goal. Given that tradeoffs are analysed by manipulating the satisfaction degree of conflicting goals and that the satisfaction degree of goals is obtained from the assessment of how well COTS products meet them, it is right to say that COTS products will influence the negotiation of conflicting goals. We address this issue in further detail during the mismatch analysis phase in Section 6.5.

### 6.2.6 Manage Goals

The final step of the goal specification phase of TAOS is the management of goals. This step will continue in parallel with other phases of the method. Given that goals for COTS-based systems need to change a lot, it is important to manage goals properly. As we have discussed in Chapter 4, the main reasons for changes in goals are the frequent evolution of COTS products and mismatches between goals and COTS products forcing goals to be negotiated and adjusted. Therefore, it is reasonable to say that the sources of change in goals are mainly imposed by COTS products since they do not match perfectly with the goals of stakeholders. The goal management process covers the whole evaluation process and continues through the lifetime of the system.



**Guidelines to Manage Changes in Goals**

- 1. Evaluate the rationale behind the proposed change**  
In order to understand the rationale for changes, it is necessary, at first instance, to clearly justify the existence of that goal. Another important point to consider is understanding the constraints and circumstances that force goals to change (i.e. mismatches, risks).
- 2. Assess the impact of changes**  
If the change is inevitable, we need to assess the effects of the change, which includes affected goals, risks, costs, etc. Another helpful strategy is to perform a cost-benefit analysis to ensure the change is worthy.
- 3. Decide if the change is acceptable or not**  
The decision to change takes into account the results of the impact analysis. If the impact of the change is not affordable or if the cost to implement it outweighs the benefits of the change, then the change is considered to be unacceptable. On the other hand, if after the impact analysis, it is perceived that the change is vital and pays off the involved risks, then the change should be accepted but the full impact analysis should be incorporated with the decision.
- 4. Implement the change if accepted**  
The change implementation involves extensive negotiation with stakeholders and, in case of conflicting goals, it is necessary to perform the tradeoff analysis again. Implementing changes also includes changing affected goals.

Figure 6.18: Guidelines to Manage Changes in Goals

The objectives of the goal management process are slightly different in each phase of the TAOS method. Initially, this activity is concerned with understanding and managing the relationships between goals, for instance, examining goal decomposition and interactions. Then, during the COTS assessment phase, new goals may emerge as stakeholders develop

a better understanding of the system from observing COTS features. Finally, during the mismatch analysis phase, a traceability mechanism is needed to reason about the impact of decisions and to assess risks posed by changes in goals. A well defined change management process ensures that the evaluation team will understand the impact of changes, as well as the rationale behind such changes and risks involved. It also provides appropriate reasoning for rejecting change proposals that are unacceptable or pose substantial risk. We provide some guidelines in Figure 6.18 to support the management of changes in goals.

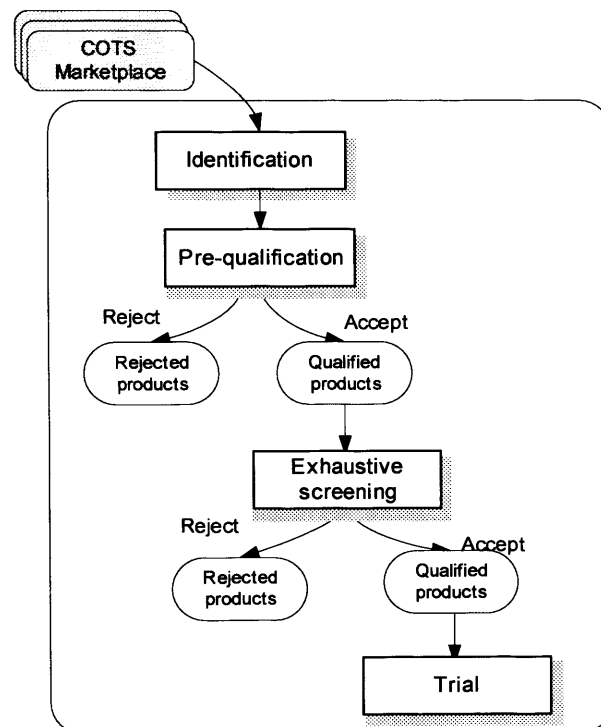


Figure 6.19: COTS Assessment Phase

## 6.3 Assess COTS Candidates

The main objective of this phase is to identify and evaluate COTS products. Given that the full understanding of large, complex COTS packages is a time consuming and intricate task and that generally there is a large number of COTS candidates available on the market, the evaluation of all potential alternatives generally demands more resources than the organization can afford. Therefore, in order to keep this process manageable, it is necessary to concentrate the assessment of the most promising products, i.e. the ones that better satisfy critical stakeholder goals. The COTS assessment phase of TAOS consists of four steps. Figure 6.19 describes how the steps are connected and highlights the qualification/rejection of COTS candidates along the assessment process.

### 6.3.1 Identification

This step involves the discovery of COTS products available in the marketplace. The search for alternatives starts as soon as the key goals for the CBD project have been defined. It is recommended to state quite generic goals so that the search is not limited by unnecessary constraints. For instance, we should start looking for e-commerce tools that satisfy the high level goals for the online bookshop system (as shown in Figure 6.4). At this stage, the evaluation team should concentrate on examining how the available tools meet the core goals, while the desirable goals should be explored in a latter stage.

Usual sources to gather information about potential COTS alternatives include: Internet, conferences, specialised literature, other organizations, consultants, etc. By the end of the identification process, the evaluation team should obtain a list of all potential candidates. This list should include commercial descriptions of the main functionality provided by the products. Generally, the identification of COTS products is a straightforward activity that requires a short time to complete. Depending on the complexity of the application domain and the scope of products, it is possible to find different COTS solutions ranging from a single, large products to several fine grained, specific packages that once integrated will provide the desired functionality. It is important to note that the more products you select, the harder is the work necessary to successfully integrate them. To develop the online bookshop, for instance, the evaluation team could opt for choosing a set of web services to compose the desired solution or select a complete e-commerce product from a single supplier. We assume that the evaluation team opted for second option because they considered it poses less risks and the system could go into operation earlier than the web service solution.

### 6.3.2 Pre-Qualification

Large COTS products are too complex to be examined in a comprehensive manner. Therefore, a pre-qualification approach is needed to narrow down the number of COTS candidates. This phase typically consists of preparing the pre-qualification questionnaire and invitation to tender (ITT) document. The objective of the questionnaire is to describe the main objectives of the tender process and the key evaluation criteria. The criteria should consist of critical high level goals and any relevant issue that can be used to distinguish COTS candidates. Critical goals are identified during the goal prioritisation step (see Section 6.2.5). Based on the weight of goals, the evaluation team can decide on the appropriate baseline to categorise critical goals. The ITT document consists of more specific explanation of the requirements the organization aims to achieve with the tender process.

Figure 6.20 describes some guidelines to compose the pre-qualification questionnaire in order to shortlist the most promising alternatives and eliminate the poor ones. Since the pre-qualification questionnaire may vary between different acquisition projects and organizations, the purpose here is to provide generic guidelines that can be modified to suit different situations. The first guideline refers to evaluating the extent to which the each product meets critical goals. It is important to note that at this stage, the evaluation team may not have sufficient evidence to confirm goal satisfaction. Some information may be only available during the trial period (see Section 6.3.4), when the suppliers have to demonstrate their products against test cases. Some of the guidelines are quite subjective, but as we have discussed before, the selection of COTS, rather than being simply a technical decision, involves a good amount of common sense and intuition from the evaluators. The last guideline is particularly important when building software applications for complex, dynamic organizations with continuously evolving business processes. For such organizations, having a committed and trustful supplier can be the decisive selection criterion.

Once the questionnaire is prepared, suppliers are invited to participate on the tender. This process basically consists of each supplier responding and returning the questionnaire. During this phase, it is not feasible to compare every bid against the others, as this might involve a significant amount of work to successfully compare all possible alternatives. Instead, each supplier is individually assessed based on how well the product satisfies the pre-qualification criteria. A major risk at this stage is that an overly long and intricate questionnaire is created that might discourage suppliers from responding it. In such cases, promising suppliers could be outside of the evaluation process because they didn't want to waste time participating on the tender process that would not necessarily result in a new customer. To avoid such situations, the questionnaire should be specified in a very simple, unambiguous and objective manner. Once suppliers are short listed, the evaluation team can examine the selected products in-depth during the exhaustive screening step.

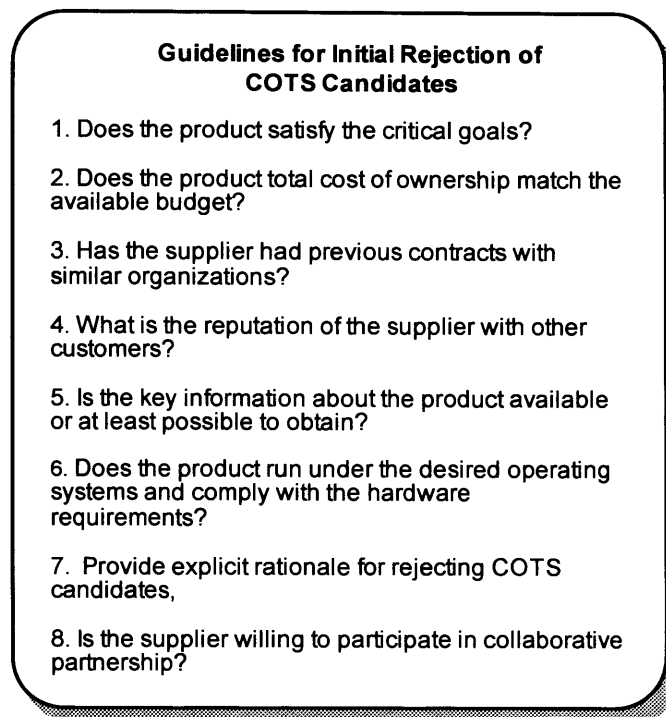


Figure 6.20: Guidelines for initial rejection of COTS Candidates

### 6.3.3 Exhaustive Screening

In the pre-qualification phase the short listed suppliers have simply stated what the main functionality offered by their products is. There is no guarantee, however, that the promised functionality is effectively implemented. The objective of the exhaustive screening is to verify vendor claims by asking suppliers more specific information about COTS products. An effective approach to do that is by arranging demonstration sessions with suppliers to better understand how their products satisfy the goal specification. The priorities given to goals (see Section 6.2.5) will guide the assessment of products, in such way that the evaluation team should concentrate the evaluation effort to verify how products satisfy critical goals. The exhaustive screening also involves the analysis of supplier responses and, in cases where a product capability is insufficiently understood to allow its adequate evaluation, a request for clarification is sent to the supplier. Request for clarification is an important mechanism to perform informed assessment.

The assessment of COTS features involves a good deal of uncertainty, where key information to assess the satisfaction of goals can be quite difficult to obtain. To ensure that the assessment process is as accurate as possible, each observed functionality must have a confidence degree associated with. The confidence degree is based on well justified arguments and evidence that a desired functionality (i.e. operational goal) is sufficiently

satisfied. We have established different levels of confidence: *verified*, *demonstrated* and *informed*. The highest confidence degree is obtained when the goal satisfaction is *verified* while the lowest one happens when the goal satisfaction is simply *informed*. Obviously, ensuring a high confidence degree implies extra costs, which means that the evaluation team needs to judge when a high confidence degree is required. In the following we explain each confidence degree:

*Verified* - perform exhaustive testing simulating the operational environment and verifying what is necessary and sufficient to achieve the operational goal,

*Demonstrated* - investigate how the product satisfies the operational goal during demonstration sessions conducted by the supplier,

*Informed* - examine commercial descriptions and product manuals as well as obtain relevant information with other organizations and third-party assessors.

#### 6.3.4 Trial

The main objective of the trial step is to perform detailed testing of products quality and functional suitability. Testing of relevant COTS features is known to be a time-consuming task. Therefore, in order to reduce the complexity of the testing activity, it is necessary to perform another rejection process to eliminate poor alternatives that failed to meet critical and non-negotiable goals. At this stage, it is recommended to qualify no more than three or four suppliers to the trial period. In the pre-qualification stage we described some guidelines to facilitate the first elimination of COTS candidates. Now, in the second round of elimination, the decision is driven by more detailed criteria. Figure 6.21 shows some guidelines to facilitate the identification of detailed rejection criteria. Based on these guidelines, the evaluation team can create a qualification checklist to determine which products are short listed and which products are rejected.

The decision to shortlist alternatives is not based simply on how well the products meet technical aspects of the qualification criteria, other more subjective factors might also influence the decision-making. Depending on the objectives of each COTS acquisition project, the rejection decision can be more influenced by economic and strategic aspects than technical issues. It could be the case that a poorer supplier in terms of technical functionality could be preferred to a technically superior supplier if the supplier brings other values, such as: cheaper product, better support, good reputation with customers. Independent of the nature of the qualification criteria (i.e. if it is mainly driven by technical or strategic factors), the rationale for supplier elimination has to be carefully recorded and justified by explicit arguments to avoid disqualifying promising candidates. Once the refined list of products is obtained, the evaluation team can now arrange the testing sessions. The testing effort can be divided into two categories:

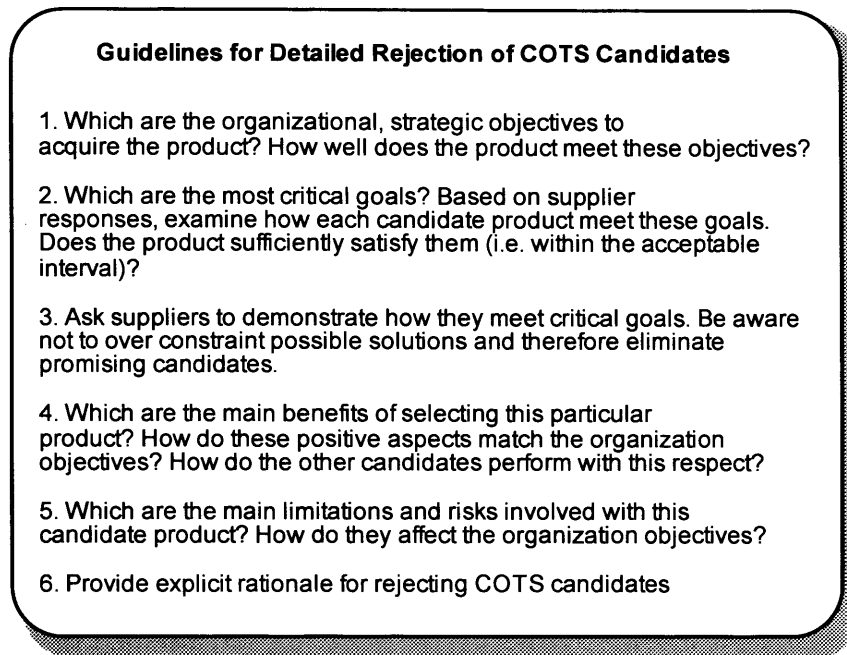


Figure 6.21: Guidelines for detailed rejection of COTS Candidates

*Quality Testing* - To verify whether the product presents undesirable system behaviour by discovering faults or defects. This type of testing reveals integration problems and consequently exposes the necessity of adapting the system,

*Acceptance testing* - To demonstrate that the product meets operational goals. This type of testing requires the participation of stakeholders to check whether the system conforms to their expectations.

COTS products are generally delivered as black-boxes, which means that the testing effort will examine the system behaviour without having access to the source code. A key principle of black box testing techniques is the definition of test case inputs and expected outputs that ultimately correspond to stakeholder requirements [Voas, 1998]. In addition, the testing effort helps identifying missing functionality (i.e. unsatisfied goals) as well as finding incompatibilities between products that need gluing or wrapping to be integrated. Generally, the buyer organization performs the testing sessions with the presence of key stakeholders, the sessions can also be assisted by consultants and specialists in the domain.

Testing sessions can be facilitated by the use of test cases to provide a systematic mechanism to perform both quality and acceptance testing. The success of these sessions highly depend on the effectiveness of the test cases to guide the examination of critical functionality and quality of the product. Figure 6.22 provides some guidelines on how to design test cases.



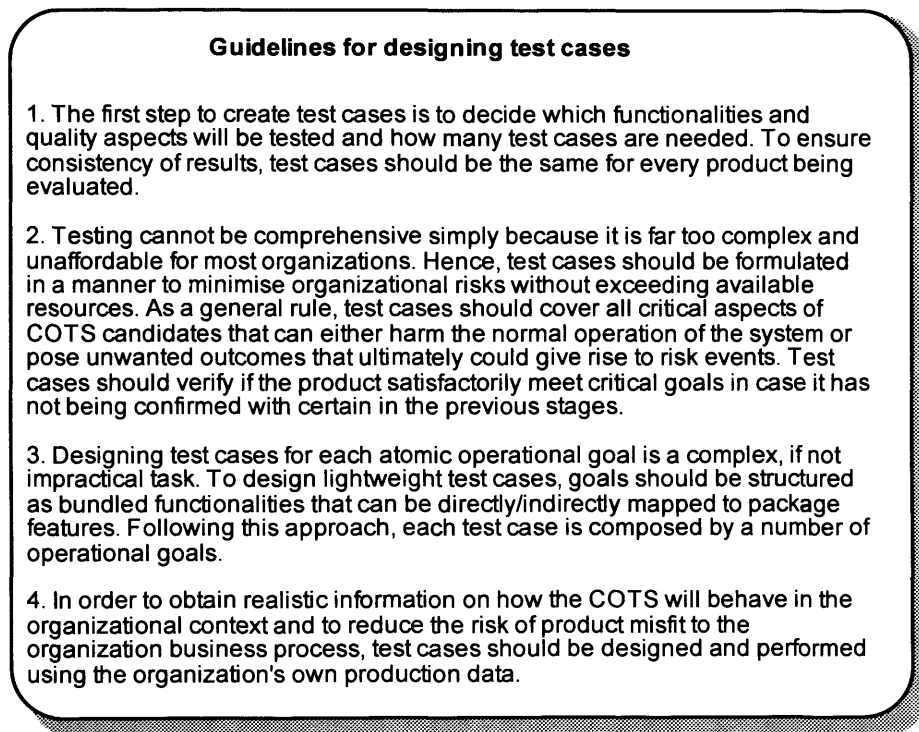


Figure 6.22: Guidelines for designing test cases

## 6.4 Perform Matching

The matching analysis is defined as the process of assessing how well each COTS being evaluated meet operational goals. This phase involves gathering sufficient information about products in order to assign satisfaction scores to operational goals. Based on the results of the matching, the evaluation team can aggregate individual satisfaction scores (i.e. how the product satisfies each operational goal) into global satisfaction scores (i.e. how the product satisfies the set of operational goals). As a result, it is possible to quantitatively compare COTS candidates and then, inform the decision making process. The matching process involves the analysis of COTS satisfaction and further discussion with involved stakeholders to determine whether a particular product sufficiently satisfies their needs or not. The matching of a particular COTS product is said to be satisfactory if the product satisfies the operational goal within its acceptable interval. The matching analysis can be performed in two ways:

*Specification Comparison* - The comparison between goal specification and product documentation is a common way to understand how features satisfy the goals of the organization. The advantage of this approach is the reduced effort needed to obtain the relevant information. Written information about product functionalities is generally abundant and easy to obtain. By simply comparing specifications, however, the confidence level is quite low that the product will truly behave as promised, since no formal verification is performed. The comparison between specifications is adequate to verify the matching of low priority goals. If, at a later stage, the evaluation team discovers that these goals are not fully satisfied, it is unlikely that this situation will pose any major risk to the selection process as a whole. On the other hand, the satisfaction of critical goals should be verified more precisely.

*Acceptance Testing* - As we described in Section 6.3.4, acceptance testing is a strategy based on the use of test cases to verify how well product features implement goals. The objective of test cases is to provide factual evidence that the product meets the desired functionality. During evaluation sessions suppliers are asked to demonstrate how their product meet a particular set of functionality. Test cases can be also designed to explore quality aspects of products, such as: performance and security testing. As an example, suppliers may demonstrate whether their products meet response time requirements by simulating a situation where 100 users are logged into the bookshop system and measuring the response time to accomplish tasks such as time to process customer account or time to show query results.

The matching analysis is a difficult step of the COTS evaluation process, which involves a considerable amount of subjectivity because the assignment of satisfaction scores is determined by different stakeholders with different views. In addition to that, the satisfaction

of operational goals can be hard to determine because the mapping between them and COTS features is not always explicit. These situations can be the result of inadequate understanding of the problem as well as vocabulary clashing. Let us examine the first situation. Operational goals are the refinement of strategic objectives the organization aims to achieve. This means that the goal specification is concerned with capturing and describing the problem (e.g. the operational goal - customers should be able to search books by author). While COTS products represent solution alternatives to solve a given problem (e.g. the feature - when a user requests a search for a book, the system will connect to a SQL database where the books inventory is stored). Given that COTS products are designed and implemented in very different ways, solutions provided by such products are generally different from the goals of the buyer organization.

Another type of inconsistency, known as semantic ambiguity [Lamsweerde et al., 1998], occurs as a result of different vocabulary used by suppliers to describe product features. For example, bread crumb navigation and navigation track control are features described by e-commerce tool suppliers. At first, they may seem distinct but after further analysis, the evaluation team observes that they refer to the very same functionality of keeping track of pages visited by customers and allowing them to easily go back to previously visited pages. Semantic ambiguity and problem understanding are usual difficulties that may prevent the direct mapping between goals and features. To handle such problems, it is vital to gain the cooperation of suppliers to clarify ambiguous and inconsistent issues.

As we discussed in Section 6.2, goals can be satisfied to a degree [Lamsweerde, 2001]. By using that motivating idea, we have defined the concept of acceptable interval in Section 6.2.3 to determine the values in which operational goals are considered to be sufficiently satisfied by the product being evaluated. The acceptable interval is represented by normalised values ranging from the *target level*, where the satisfaction degree is 1 and at the *worst level* where the satisfaction degree is 0. We say that the operational goal is fully matched if the COTS satisfies the operational goal at the target level. Similarly, the matching fails if the COTS satisfies the operational goal at the worst level, which means that the COTS being evaluated does not acceptably meets it. Now, if the degree to which the COTS satisfies the operational goal is within the acceptable interval, we say that the operational goal is partially matched. Another possible matching situation occurs when the COTS product offers more functionalities than the stakeholders have asked for. In this case we say that the COTS extends stakeholder goals. To precisely capture these situations, and therefore, facilitate the assessment of how well COTS candidates meet operational goals, we have defined a set of matching patterns [*fulfil*, *differ*, *fail*, *extend*, *uncertain*]. Figure 6.23 illustrates the meta model of the matching process. In the following we explain each matching pattern in detail.

#### Fulfil Pattern

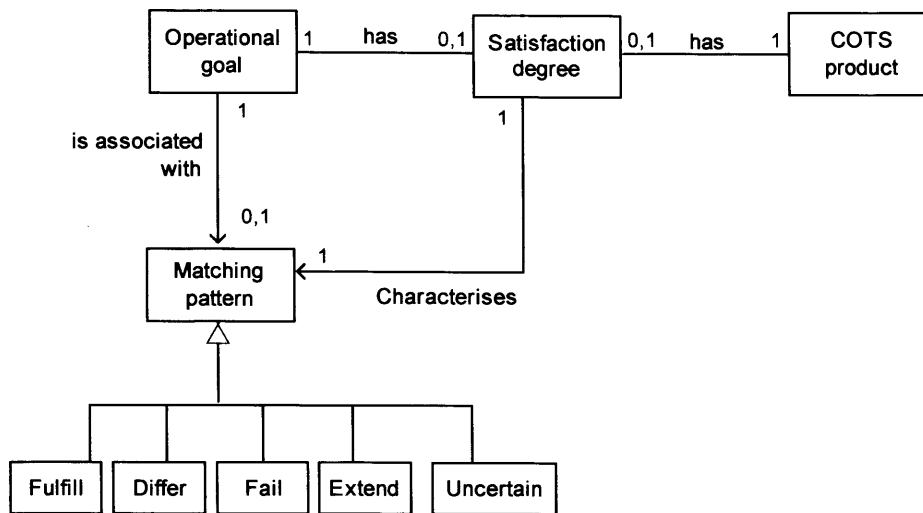


Figure 6.23: Matching Meta Model

The matching between an operational goal and COTS product is complete if the product satisfies the operational goal at the target level of the acceptable interval. Consider, for instance, the operational goal “fast sales authorization”. According to Figure 6.8, the goal is fulfilled if and only if the system authorizes sales within 10 seconds. This is an example of clear-cut matching, in which the goal satisfaction is determined explicitly and unambiguously. Consider now that we want to determine the matching of goal “the customer can opt if he wants to receive a notification e-mail when an order is placed”, the metric associated with this goal is a boolean one (i.e. the product can either support the facility to send e-mail notification or not). Therefore, a product fulfils this operational goal if the system asks the customer before sending the notification of a placed order. Let us consider now that there is a product on the market, called BuyNow that sends a message notifying customers whenever an order is placed. Note that this feature is slightly different from what Black Books shop wants. In such cases, the evaluation team has to judge to what extent the divergence between the problem and the solution is acceptable. Consider the evaluation team decides that the difference is insignificant, then we have that the product BuyNow fully satisfies the operational goal. This situation highlights the importance of having good judgement and domain expertise to assign appropriate satisfaction degrees. The formal definition of the *fulfil* pattern:

*Definition 1* - Let  $g_i$  be an operational goal and  $A$  a candidate product. We define the fulfil pattern  $\gamma$  as a relation  $\gamma(g_i, A)$  if product  $A$  satisfies operational goal  $g_i$  at target level  $x_{target}$  so that  $Sat_{g_i}((A_i(x_{target})) = 1$

Differ Pattern

The differ pattern represents cases where the operational goal is partially satisfied when the product meets the operational goal within the acceptable interval, but excluding the boundary limits (i.e. target and worst levels). For instance, according to Figure 6.8, the matching pattern of goal “safe sales authorization” should be *differ* if the product supports the security policy ranging from very high to low levels. This is the most common situation of *differ* matching, when the satisfaction of the operational goal is acceptable but not optimal. Consider now the evaluation team decides that the way the product BuyNow satisfies operational goal “the customer can opt if he wants to receive a notification e-mail when an order is placed” is not optimal but satisfactorily acceptable. In this situation, we say that BuyNow partially satisfies the desired functionality. This is a type of *differ* matching that occurs because the product does not satisfy all constraints imposed by the operational goal.

Another situation that leads to a *differ* matching happens when it is verified that the product satisfies part of the desired functionality but it is not possible or it will require substantial effort to demonstrate that the operational goal is fully satisfied. As a result, at first instance, the *differ* matching is assigned and then the evaluation team judges if it is worth carrying out further testing. As an example, consider that we want to verify if a particular candidate product meets the desired “system availability”, in which uptime ratio is the chosen metric to measure system availability. It represents the percentage of time the system is available throughout its useful life. During evaluation sessions, the availability of the e-commerce application is verified by running availability testing. These tests analyse if the system is available for users even when the hardware is replaced or its software is upgraded. In particular, the need to expand and improve e-commerce sites at a rapid pace means that the site will undergo continual change, which in turn may often result in errors that create unexpected downtime. The comprehensive testing of system availability is a difficult and time consuming process because the evaluation team has to evaluate how well the system predicts and prevents problems with the availability of data that is split among multiple database servers, application servers, etc. Therefore, if the system availability goal is not fully satisfied and demonstrated, the evaluation team may decide to assign the *differ* matching to it. To summarize, the *differ* matching typically occurs in the following situations:

- When the operational goal is satisfied within the acceptable interval but excluding the boundary limits (i.e. target and worst levels),
- When the product does not accomplish all constraints stated by the operational goal,
- When the satisfaction of the operational goal is partially demonstrated, so that it is not possible to assert for certain that the operational goal is fully satisfied.

We give the formal definition of the differ pattern as follows:

*Definition 2* - Let  $g_i$  be an operational goal and A a candidate product. We define the differ pattern  $\delta$  as a relation  $\delta(g_i, A)$  if product A satisfies operational goal  $g_i$  at level  $x_k$  within the acceptable interval so that  $0.1 \leq Sat_{g_i}((A_i(x_k))) \leq 0.9$

The differ pattern characterizes situations where the satisfaction of operational goals is partial. This means that various degrees of mismatch can occur between what is wanted from the COTS product and what the product in fact offers. As we have discussed in Section 2.2.1, when an organization decides to acquire a COTS solution it is important that stakeholders understand that available products may not fit exactly the desired functionality and level of quality. As a result, mismatches are inevitable situations that stakeholders have to tolerate. In some circumstances it may not be practicable to eliminate mismatches as it can involve substantial effort. Therefore, before trying to solve any mismatch, it is necessary to assess the risks posed by mismatches, judge if the mismatch is acceptable or not, then propose appropriate resolutions. We will discuss this issue in further detail in Section 6.5.

#### Fail Pattern

The *fail* pattern characterises extreme situations where the mismatch is complete. In other words, the operational goal is fully unsatisfied, hence the degree to which the candidate product satisfies the operational goal is zero. This generally occurs in two circumstances:

- When the degree to which the product satisfies the operational goal is below the worst level of the acceptable interval,
- When the product does not provide the desired functionality.

The formal definition for the *fail* pattern is given bellow:

*Definition 3* - Let  $g_i$  be an operational goal and A a candidate product. We define the fail pattern  $\theta$  as a relation  $\theta(g_i, A)$  if and only if product A satisfies operational goal  $g_i$  at worst level  $x_{worst}$  or below, so that  $Sat_{g_i}((A_i(x_k))) = 0$

Similarly to the *differ* matching, the acceptability of *fail* situations has to be analysed before exploring possible mismatch resolutions. *Fail* situations can indicate that a product alternative performs poorly. As a result, such situations help distinguishing products that *fail* to satisfy critical goals. *Fail* situations can also suggest unrealistic goals that cannot be achieved by any available product or even by the current technology. In such cases, the evaluation team should reexamine the feasibility of the specified goal. We will discuss how this can be done in the mismatch analysis phase of TAOS.

#### Extend Pattern

The *extend* pattern characterises situations where the COTS product provides extra functionality that are not requested by stakeholders, and hence this additional functionality cannot be mapped into the goal specification. While *fail* mismatch is caused by the lack of product functionality, *extend* mismatch occurs due to extra product functionality. As we have discussed in Chapter 2 off-the-shelf products generally offer a large number of features that are intended to meet the needs of the whole marketplace rather than the specific needs of a particular customer. It is reasonable to expect that much of this functionality will exceed the needs of many customers. The *extend* pattern represents such situations where product functionality extends stakeholder goals. To reason about the impact caused by *extend* mismatch, let us examine some possible ways the extra functionality may interact with stakeholder goals:

*Hurtful* - The additional feature has a negative impact on stakeholder goals or other systems in which the product will be integrated with. For example, let us consider that a particular e-commerce tool provides a module in which the history of every search performed, and every product purchased by a particular customer is stored and displayed to customers searching for similar books. Although this unexpected feature may help customers to easily find related books, the bookshop owners fear that this feature may hurt the privacy concerns of customers. The example described above illustrates the case where the extra feature conflicts with stakeholder needs and concerns. Another hurtful situation happens when the extra feature conflicts with the legacy system in which the product has to be integrated with. An example of this situation occurs, for instance, if the e-commerce tool does not interoperate with the database system where the bookshop inventory is stored.

*Helpful* - The *extend* mismatch is considered helpful when a particular feature, not originally requested by stakeholders, is considered a desirable functionality. As a result, stakeholders may decide that it should be included in the goal specification. It is quite common when evaluating COTS products to discover unrequested functionalities that help clarifying vague wishes and expectations of stakeholders. Moreover, stakeholders are able to discover new requirements they have not thought about before as well as to develop a better understanding of the system to be acquired.

*Neutral* - It can also happen that the extra feature does not affect stakeholder goals either positively or negatively. Unlike hurtful and helpful situations, neutral features do not need any further action to be taken.

The formal definition of the extend pattern:

*Definition 4* - Given goal specification  $G$  and candidate product  $A$ , consider feature  $f_i$  provided by  $A$ . The extend pattern  $\xi$  occurs if  $f_i$  is not mapped in specification  $G$ .

Matching pattern	Satisfaction degree
Fulfil	1
Differ	0.1...0.9
Fail	0
Extend	Not applicable
Uncertain	Not applicable

Table 6.2: Satisfaction degree associated with matching patterns

### Pattern Uncertain

Frequently, evaluators may not have sufficient information about product features to classify the matching. Therefore, further clarification is needed in order to confirm which is the right matching pattern, and consequently to check how the product being evaluated satisfies a particular operational goal. When these situations happen, we say that the matching pattern is *uncertain*. To solve such problems, the evaluation team can request clarification of unconfirmed goals as discussed in the exhaustive screening phase and try to assign the appropriate matching to the goal in a later stage of the evaluation process. In case of time constraints, preference should be given to verifying the satisfaction of unconfirmed critical goals rather than low priority ones.

The next step of the matching phase consists of aggregating the obtained matching information in a meaningful way to facilitate the decision process. To achieve that, we need to obtain the overall satisfaction score of each COTS candidate, then compare each alternative to decide which is the most suitable product to acquire. As we have discussed in Section 6.2, the evaluation of how well products meet the organization needs has to be performed at the level of operational goals because they are measurable refinement of higher level, abstract goals. The satisfaction of operational goals is measured in terms of pre-established metrics and verified during trial sessions. Once the satisfaction degree of operational goals is determined, we can assign appropriate matching patterns to each operational goal, as shown in Table 6.2.

Figure 6.24 illustrates the matching between three COTS candidates and operational goals. As soon as individual satisfaction degrees and respective matching patterns are assigned, we can compute the overall contribution each COTS gives to the satisfaction of goals. From the results obtained in equation 6.26, we can use the weighted summation operator to aggregate the overall satisfaction degree for each COTS product. We say that the overall satisfaction of COTS A is the aggregation of the satisfaction degree that COTS A meet each single operational goal, then we have that:

$$SatCOTS_A = Sat_A(g_1(x_1), g_2(x_2), \dots, g_n(x_n)) \quad (6.27)$$



Now from equation 6.4, the aggregated satisfaction score of COTS A is given by:

$$SatCOTS_A = \sum_{i=1}^n w_{g_i} \times SatCOTS_A(g_i(x_k)) \quad (6.28)$$

where  $w_{g_i}$  is the weight of operational goal  $g_i$  and  $SatCOTS_A(\cdot)$  is the satisfaction degree of a single operational goal scaled from 0 to 1.

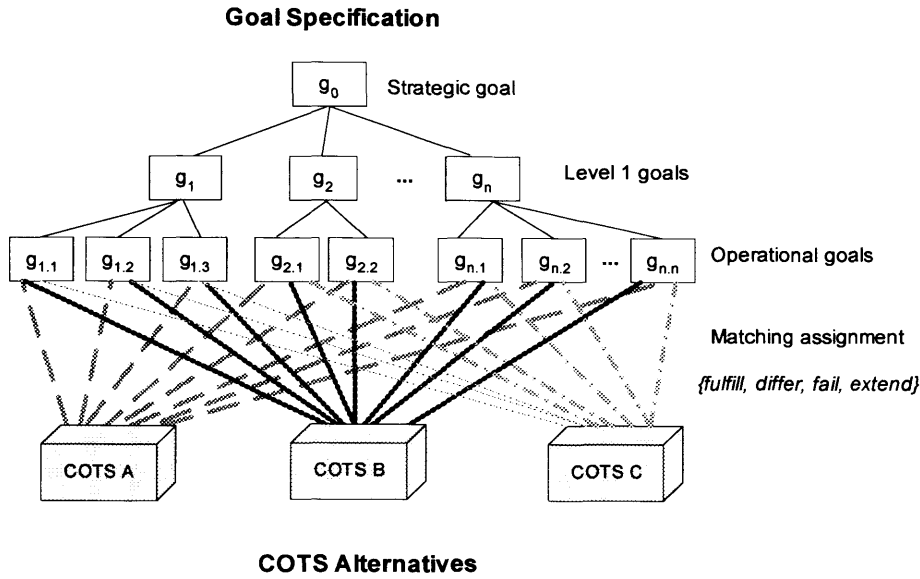


Figure 6.24: Matching between goals and COTS Products

In Section 6.2.5 we have discussed that the validity of additive aggregation depends on the tradeoff condition to be satisfied. It should be stressed that the aggregated satisfaction score is not sufficient to select a preferred COTS option, instead it helps the evaluation team to quantitatively compare different COTS alternatives. It allows a large amount of diverse information to be sorted out in a meaningful way so that the evaluation team can explore the most critical issues of the preferred options. To express the relative preference between COTS alternatives, we say that COTS A is preferred to COTS B if the aggregated satisfaction score of COTS A is higher than COTS B:

$$\text{COTS A is preferred to COTS B} \iff \left( \sum_{i=1}^n w_{g_i} \times SatCOTS_A(g_i(x_k)) \right) > \left( \sum_{j=1}^n w_{g_j} \times SatCOTS_B(g_j(x_k)) \right)$$

In other words, the preferred COTS alternative is the one that maximises the expected satisfaction of operational goals (i.e. meet operational goals at the target level). The relative ranking among COTS is defined by the aggregated numerical score of each COTS alternative. While aggregation techniques, such as the weighted summation explained above, provide an effective quantitative mechanism to compare alternatives in terms of

how well they satisfy the given criteria (i.e. operational goals), they give little support to reason about the impact of mismatches. The decision to select complex COTS systems should not rely solely on the relative ranking of COTS alternatives, it is also necessary to analyse how unsatisfied requirements may constrain the successful integration of the chosen product. In order to properly address these issues, the next phase of TAOS presents a strategy to analyse mismatches and handle risks.

## 6.5 Analyse Mismatches and Manage Risks

As we have discussed in the previous section, the matching process informs the evaluation team how well candidate products satisfy goals of the buyer organization. The relative ranking approach does not help the evaluation team to reason about the risks posed by each alternative. Analysing these risks is an essential step to make rational decisions. As discussed in Section 3.3, failing to understand and manage risks can lead to unsatisfactory solutions or even complete project failure. As a result, the evaluation team needs a systematic approach to understand the effects of mismatches, analyse conflicts between goals, explore tradeoffs and manage risks.

In the context of COTS selection, risks are defined as unacceptable outcomes, generally caused because of conflicting goals and mismatches. Given that mismatches represent non-adherence of COTS products to operational goals and conflicts arise when satisfying one goal hurts the satisfaction of another goal, we consider that risks arise when the loss caused by unsatisfied goals is intolerable. A fundamental issue in handling mismatches is the capacity to systematically structure tradeoffs. The tradeoff analysis forms the basis of our risk management strategy. The objectives of risk management strategy are to understand and handle risk events before they become threats to successfully selecting and integrating the chosen COTS product. Our risk management approach consists of three steps: risk identification, risk analysis and prioritization, and finally risk resolution.

### 6.5.1 Risk Identification

The objective of the risk identification step is to analyse potential sources of risk and judge if they originate risk events. Firstly let us examine the circumstances under which conflicts arise. Similarly to what happens in negative interaction between goals, mismatches do not always give rise to conflicts. Conflict refers to the interference between two or more parties with diverging interests [Alves and Finkelstein, 2003]. Therefore, the necessary condition for a conflict to occur is the existence of two opposing parties. The usual parties involved in COTS selection are suppliers and customers. However, suppliers are not necessarily worried of mismatches between their product and customer requirements, instead, this is primarily a customer's concern. Some mismatches can cause conflicts, when customers

decide to engage in a negotiation process to persuade the potential supplier to satisfy mismatched goals. Mismatches may give rise to a number of negative outcomes, such as, insufficient product quality and functionality or even complete failure to satisfy critical goals, unwanted product features, etc. While some of these problems may not pose any significant risk to the organization, others can originate major threats to the successful selection and further integration of the COTS product within the organization.

When evaluating COTS products, a number of risks can be originated from unsatisfactory outcomes, such as: goal mismatches, conflict between goals, uncertainty concerning product capabilities and future evolution, unrealistic schedule and budget, limited vendor support, etc. Since mismatches are the most frequent and severe cause of risk, we start the risk identification step by analysing the nature and impact of mismatches. As we discussed before, mismatches are characterised by *differ*, *fail* and *extend* patterns to characterise situations where goals are not satisfied as desired. Even though *fulfil* cases represent complete COTS adherence to goals, which means that they do not lead to mismatches, they still may pose some risks due to negative interactions and consequent conflicts between goals. In Section 6.2.4, we have discussed the importance of examining and solving conflicts between goals by means of structuring tradeoffs. The utmost objective of the tradeoff analysis is to inform the decision process. For instance, if a particular product satisfies a desired functionality in such a way that it hurts the achievement of another important goal, then the evaluation team needs to explore the severity of this negative interaction. By examining that, the evaluation team is aware of the potential risks involved with the decision to select this particular product.

A fundamental stage of the risk identification is the discovery of *risk factors*. Risk factors are unsatisfactory outcomes. Risk factors, in turn, affect the probability of *risk events* occurring. Consider, for instance, that COTS B satisfies operational goal *safe sales authorization* at degree 0.6, which indicates a *differ* mismatch. In this situation, the *differ* mismatch is a risk factor. Once the risk factor is identified, now we have to examine the effects and acceptability of the mismatch to decide if it originates a risk event or not. An important aspect to look at is the importance of achieving the goal (see section 6.2.5). Generally, when critical goals are not sufficiently satisfied, risks are originated. While low priority goals may have a more elastic range of acceptable satisfaction degree, which means that they can be more easily compromised in order to avoid risks. The optimal outcome of the matching process is when the operational goal is fully satisfied. There are situations where the satisfaction of an operational goal is inferior than optimal but still represents a satisfying solution. This means that, the satisfaction is not the “best” but “good enough”. Understanding the threshold in which the goal satisfaction is minimally acceptable is a key factor in identifying a risk event. Table 6.3 provides a template to facilitate the identification of risks.

Element	Explanation	Example
Risk factor	Describe unsatisfactory outcome	Differ mismatch between operational goal safe sales authorization and COTS B
Effects of risk factor	Investigate and explain the negative effects of the risk factor	If sales transaction is not sufficiently secure, the system can be exposed to attacks
Acceptability of risk factor	Analyse the the acceptability of not achieving the desired outcome	COTS B satisfies the goal “safe sales authorization” at level 0.6, after discussion with stakeholders they agree this is not a satisfactory security level
Risk Event	Describe the risk event triggered by the risk factor	Customer credit card details may be intercepted

Table 6.3: Risk Identification Template

### 6.5.2 Risk Analysis and Prioritization

In the previous section we discussed how to identify risk events by exploring the acceptability of risk factors and understanding their effects. By following this approach it is possible that several risks will be identified, which means that the evaluation team will need a significant amount of time to investigate all identified risks. Consequently, it is necessary to analyse and prioritise risks to concentrate on handling the most critical risks.

In order to analyse the impact of risks and hence, be able to prioritise them, we use the risk exposure technique described by [Boehm, 1991]. The risk exposure is a quantitative approach to rank risk events and determine which ones are the most important to address. Risk exposure depends on the probability of a risk event to occur combined with the utility loss caused by the risk. By utility loss, we refer to the loss caused by an unsatisfactory outcome, it represents the dissatisfaction stakeholders will feel if the COTS solution delivers undesired outcomes. Equation 6.29 shows how the risk exposure is calculated:

$$RE_i = P(R_i) \times UL(R_i) \quad (6.29)$$

where  $RE_i$  is the exposure of risk event  $R_i$ ,  $P(R_i)$  is the probability of an undesired outcome to occur and  $UL(R_i)$  is the utility loss to the involved stakeholders if the risk event occur.

Figure 6.25 shows how the risk exposure is influenced by the probability of risk occurrence and utility loss. The three identified zones [*low, medium and high*] represent increasing levels of risk exposure. The risk exposure zones are constant curves defined by the evaluation team to characterise the severity of undesired outcomes. The zones will serve as a basis to prioritise risks. This is done by assessing both probability of occurrence and utility loss caused by the undesired event on a relative scale of 0 to 10. Consider, for

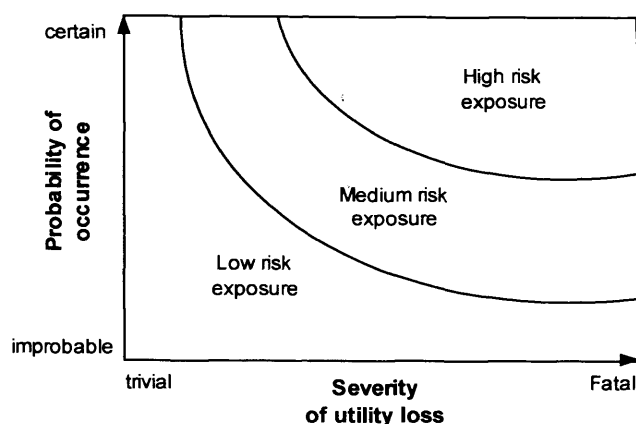


Figure 6.25: Risk factors and resulting risk exposure zones

instance, that the evaluation team estimates that the probability of risk “Customer credit card details may be intercepted” to occur is 4 and the utility loss is 8, then the resulting risk exposure is 32. Table 6.4 describes other hypothetical risks originated by COTS B and shows the ratings for  $P(R_i)$  and  $UL(R_i)$ , and the resulting risk exposure estimation. In order to assess the overall suitability of COTS B for the purposes of the decision process, it is necessary to take into consideration the aggregated satisfaction score, together with the risks posed by COTS B. By using the risk exposure zones graph (see Figure 6.25), we can plot the resulting risk exposure estimates obtained in Table 6.4. Figure 6.26 illustrates the resulting risk exposure for risks A, B and C. We observe that A has the highest risk exposure, even though it has quite low utility loss factor. While C has the highest utility loss but the resulting risk exposure is the lowest amongst the three risk events. Finally, risk event B poses medium risk exposure and has similar medium values for utility loss and probability of occurrence. Based on these observations, the risk exposure ratings can provide a basis to obtain the relative priority among these three risk events, such that:  $Prio(C) < Prio(B) < Prio(A)$ .

TAOS applies the risk estimation technique with the objective of comparing and prioritizing risks. To achieve this particular purpose, it may not be necessary to have precise values of probability and loss. Generally, an approximate estimation may be sufficient, but in cases where the estimation value is highly uncertain, the evaluation team should use strategies to reduce the uncertainty. This can be achieved by obtaining more information from suppliers about the risks and carefully examining involved effects. Once sufficient information is obtained, the evaluation team can try to estimate the risk exposure again. [Kontio, 1997] argues that the estimation of probability and loss associated with risk events is subject to biased and inaccurate assessment. Even if this technique lacks precision it suits our needs to obtain an overall notion of risks priority.

Risk event	Probability of occurrence	Utility loss	Expected risk
A. Customer credit card details may be intercepted	8	4	32
B. User interface is not sufficiently intuitive which may be difficult to find desired books	5	7	49
C. The product fails to generate store traffic reports	2	8	20

Table 6.4: Risk exposure factors for COTS B

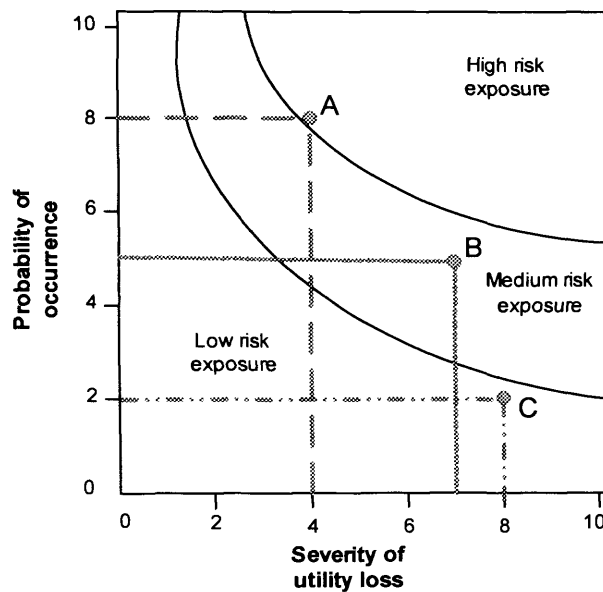


Figure 6.26: Risk events posed by COTS B with resulting risk exposure

In order to successfully explore and choose risk management strategies, we propose the use of scenario analysis to qualitatively assess the impact of decisions. In the next section we explore possible risk management strategies. Finally, in Section 6.6 we present an exploratory scenario approach to guide the evaluation team to choose appropriate risk controlling actions and to make the final selection decision.

### 6.5.3 Risk Resolution

The prioritised list of risks provides the basis for the risk resolution phase. The first step to handle risks is to propose and choose risk resolution strategies. To support that,

we have defined and refined a taxonomy of risk management strategies initially proposed in [Charette, 1990, Kontio, 1997]. Our taxonomy analyses specific risks of COTS-based development, it presents a set of options to mitigate, prevent and reduce the occurrence of risk events, it also describes situations where no risk mitigation action is needed (see Figure 6.27).

As depicted in Equation 6.29, the risk exposure directly depends on the probability of an unsatisfactory outcome happening combined with the utility loss to the affected parties. Therefore, a natural way to avoid risk events is to control the probability and loss caused by risks. In order to reduce the probability of risk events, we propose the following actions: increase the confidence degree on product satisfaction, handle uncertainty, accept irrelevant mismatches and adjust previous decisions. The first two actions refer to obtaining more information about the product being evaluated. The lack of appropriate information is a typical difficulty of COTS-based development in which some details about product functionality and quality are generally hard to obtain. The evaluation team might not attempt to fully understand every detail unless these uncertain features may pose real risks to the selection process. Some features that were not properly verified at initial stages because the assessment process was considered too time consuming, now they should be carefully examined in order to reduce the probability of risk events. For instance, a comprehensive analysis of how good is the security of transactions provided by a particular e-commerce tool can be hard to obtain. In this situation, the evaluation team may decide to perform extensive security testing if they perceive that the risk exposure of a security attack is high. By following this strategy, the evaluation team is increasing the confidence degree that the product meets the desired security level.

Another way of reducing the probability of risk events to occur is by disregarding irrelevant mismatches. Mismatches occur when the COTS solution does not fully satisfy goals. That is, when the product does not meet the operational goal at the target level of its acceptable interval. The acceptable interval represents the desirable values that stakeholders would like each operational goal to be satisfied. Initially, during the goal specification phase (Section 6.2), the acceptable interval is defined without taking into consideration the limitations of available products. Hence, adjusting the acceptable interval to tolerate imposed constraints, and consequently accepting minor mismatches, is a way to avoid risk events. For example, the evaluation team may decide to relax the desired response time to show query results if they realise that the initially defined acceptable interval is unrealistic. In a similar way, reviewing and adjusting previous decisions is an effective action to reduce the probability of risk events.

The strategies to reduce utility loss refer to limiting the impact of risk events. It consists of actions such as: issuing contractual warranties, communicating and sharing risk, performing integration tests and contingency planning. By arranging contractual warranties, the buyer organization is legally protected from the supplier violating the terms of the contract. Communicating and sharing risk among involved parties, such as users, man-

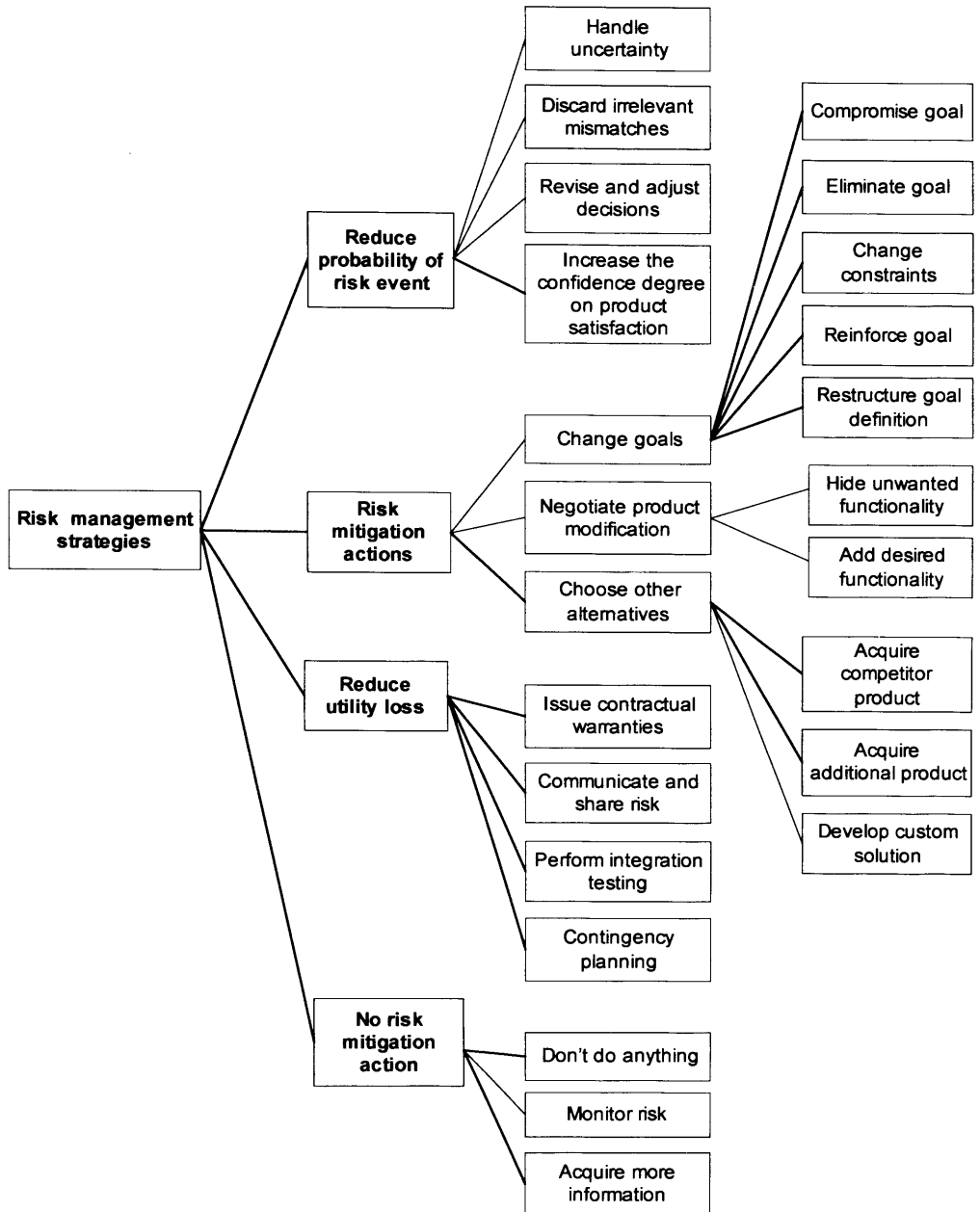


Figure 6.27: Risk Management Taxonomy

agers and suppliers is a way to reduce the damage caused by a risk event. By following this strategy we assume that it is not possible to avoid the risk. In order to control future problems caused by the risk event, the involved parties should be aware and acknowledge the



existence of the risk. Although, in many circumstances, there will always be factors outside control, this kind of participative approach is a useful risk handling strategy. Another way to control the impact of risk events is performing integration tests. The objectives of integration tests are identifying defects in how the components integrate that will require the development of additional “glue” code as well as predicting interoperability problems. Extensive integration testing is generally time-consuming, hence it is important to balance the threats posed by the product against the added value of the testing effort. This kind of decision is a typical example of tradeoff analysis in which the evaluation team needs to balance the benefits of a particular result by performing cost/value analysis. Finally, the contingency plan option involves allocating extra funds or any other strategic resolution that will help the organization to recover from the risk event.

No risk mitigation action refers to situations in which the evaluation team decides not to take any action to handle the risk event. It generally happens when the risk exposure is substantially low, such that it is more convenient not to make any effort to control the risk. This strategy includes the following options: simply don't do anything because the risk is so low that does not justify any further action to be taken, monitor the risk and acquire more information. The risk monitoring means that no immediate action is required but the risk should be observed to ensure that the impact is really low and it will not pose any additional threat. The acquire more information option refers to situations where it is necessary to better understand the risk event and its interactions with other risk events. The reason for that is because some risks can have severe and unexpected effects when combined with other risk events, that are not present when such risks occur in isolation. For example, the negative interaction between “single password strategy” and “safe sales transaction”, as shown in Figure 6.9, may not pose any major conflict, and consequently, the risk is insignificant. If besides this negative interaction, we have a product that provides unacceptable security policy, the combined risks can pose major threat to the overall security of the system. An important observation can be learned from this example, that a successful risk management strategy needs to examine the risks posed by COTS products as a whole problem of interconnected risks rather than treating each risk individually.

Finally, we propose various risk mitigation strategies in our taxonomy. Risk mitigation actions are appropriate to be applied to risk events with high and medium exposure. Risk mitigation actions cover the following options: change goals, negotiate product modification and choose other alternatives. Mitigation actions mainly deal with risks caused by conflicts originated both from mismatches and negative interactions between goals. An effective action to handle risks caused by conflicts is to change the goals involved in the conflict. A well known strategy to handle conflicting goals is by compromising them. The decision to compromise a particular goal is motivated by mismatch situations or conflicts between critical goals. This strategy involves relaxing the satisfaction of a particular goal, but in order to do that, we have to check the priority to achieve that goal. Generally,

high priority goals should not be compromised unless it is decided, after re-examining and negotiating that the initially high priority given by stakeholders is unrealistic and should be reduced. As we have discussed in the Goal Prioritisation phase (see Section 6.2.5), the final priority given to goals should reflect uncontrolled outcomes such as mismatches and conflicts.

An extreme case of changing goals is the complete elimination of the goal. This decision is appropriate in two situations: when the existence of the goal generates extremely critical conflicts or when it is absolutely impossible to satisfy the goal with the current technology, this means that the goal is unrealistic. Again, the decision to eliminate a particular goal needs to take into account the priority given to satisfy the goal. Consider, for instance, that stakeholders prefer to authenticate customers by using biometric techniques than with passwords. After examining the e-commerce solutions available, the evaluation team conclude that, at the present, biometric is not a viable implementation solution for e-commerce systems. Therefore, after examining this particular mismatch, stakeholders agree that the biometric alternative should be eliminated and they would give preference to achieving the customer authentication by means of password.

The change constraints option refers to cases where either the goal constraints are too strict and may be impossible to fully satisfy, or too soft, and may pose unwanted outcomes. As an example of increasing goal constraints, stakeholders may decide to restrict the possible solutions to allow only security certificates for online transactions from VeriSign. So far, we have examined resolutions which to a certain degree compromise goals. Now let us examine another possible risk mitigation strategy: to reinforce the importance of a goal by increasing its priority. It is very common that the importance given to some goals is underestimated in the early stages of the evaluation process due to the fact that stakeholders have incomplete understanding about the possible solutions. As a result, the priority of goals needs to be re-examined through the evaluation process, and in some cases, reinforced to reflect new circumstances. Reinforcing goals is an appropriate strategy to tradeoff a goal for another goal by increasing the preference to achieve a particular goal in order to solve a conflict. Finally, the restructure goal definition option refers to actions that handle inconsistencies in the definition of the goal in order to mitigate the risk of communication problems. The changing goal strategies are performed as part of the goal management mechanism that cover the whole selection process (see Section 6.2.6)

Another strategy to mitigate risks is to engage in a negotiation process with the supplier in order to hide or add functionalities. This strategy is appropriate in two circumstances: when the pre-selected COTS product does not sufficiently satisfy critical goals that cannot be traded off as well as when the product provides extra functionality that conflicts with stakeholder goals, and therefore the unwanted functionality needs to be hidden. In the second case, it is possible to hide the feature without direct involvement of the supplier but to accomplish the first option the supplier participation is needed. As a result, the evaluation team tries to persuade the supplier to change parts of the product that mis-

match with non-negotiable goals. A major concern of following this strategy is due to the fact that modifying the product may reduce the benefits of COTS-based systems since future versions of the product also need to be modified in order to fit the organization specific needs. Therefore, we recommend prudence when requesting product modification.

Another useful strategy to handle unacceptable mismatches is to choose other solutions. This strategy consists of actions such as acquire competitor product, acquire additional product and develop custom solution. The first option is suitable in situations where the pre-selected COTS product poses severe risks that cannot be mitigated, hence the evaluation team would prefer to reject this alternative and choose another product for which risks are more manageable. Similarly, the second option involves the acquisition of alternative products, but the difference here is that the extra product provides specific functionality that the key product does not satisfy. For example, suppose that the chosen e-commerce tool has limited and insufficient inventory management facilities. Consequently, in order to avoid the risk of implementing the online bookshop with inefficient inventory facilities, it may be necessary to acquire a specific inventory management system to be integrated with the e-commerce tool. Another possible solution to solve this mismatch is to custom build the needed inventory management functionality. This strategy allows the extra functionality to be implemented in such a way that it can successfully be integrated with the key COTS product.

It is important to note that the objective of the risk management strategies presented here is not to be an exhaustive and prescriptive classification of possible risk resolutions. Our objective here is to provide directions on how to handle mismatches and conflicts that may pose risks to the buyer organization by defining categories of possible risk management strategies. There does not exist one fits all solution, which means that the decision for the best risk resolution relies on the judgement and experience of the evaluation team. Once the potential strategies have been identified by the evaluation team, the next step is to select the best resolution or combination of resolutions.

## 6.6 Select COTS Solution

The final phase of the TAOS method consists of incorporating all information obtained during the earlier phases and using them to inform the decision process to select a suitable COTS intensive system. Note that we now refer to the product to be acquired as a COTS intensive system. The rationale for that is because a possible resolution to deal with mismatches is to select multiple products and integrate the core product with other complementary products. A COTS intensive system refers to any combination of products that successfully implements the desired functionality, such as multiple integration of COTS products, integration of COTS product with custom built system, etc.

The ultimate objective of this phase is to choose the “best solution”. Firstly, let us understand what constitutes a “best solution”. By “best” we mean that the selected COTS product(s) do not need to be optimal but satisficing. A satisficing product is the one that sufficiently satisfies the set of goals defined by stakeholders; while an optimal product aim, at any cost, to maximise the satisfaction of all goals. Here cost represents monetary investment, time, effort, complexity, or anything that may constraint the full satisfaction of goals. Given all the challenges faced by COTS-based development (as discussed in Chapter 2), it is reasonable to say that optimal solutions are unfeasible.

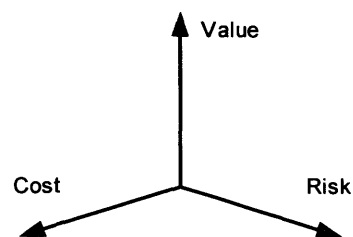


Figure 6.28: Factors to assess the worthiness of each COTS solution

In order to facilitate the identification of satisficing solutions, we measure the overall worth of each COTS intensive system(see [Robertson and Robertson, 1999]). As illustrated in Figure 6.28, the overall worth of a COTS intensive system is a combined measure of the value, cost and risk presented in each alternative. Value refers to how happy stakeholders will be if a given alternative successfully satisfies the desired goals, to a certain extent this is related to the priority stakeholders have given to goals. At this stage, however, it is clear what are the cost and risk constraining the satisfaction of the goal. As a result, the evaluation team is able to choose appropriate tradeoffs to accommodate such constraints. The suitability of each alternative is measured in terms of these three axes, where the satisficing solution will be a compromise among value, risk and cost. Instead of attaching a numerical scale to measure each of the axes as proposed by [Robertson and Robertson, 1999], we prefer to examine these interdependent perspectives using exploratory scenarios. Exploratory scenarios capture a comprehensive picture of the relevant issues involved in the decision process. We provide a set of heuristics to be used in conjunction with scenarios to assist the evaluation team in choosing adequate risk resolution strategies. In the following we describe three exploratory scenarios to guide the final COTS selection decision.

As illustrated in Table 6.5, the first proposed scenario explores some fundamental issues to be considered to judge the worth of each COTS intensive system. The value axis is assessed by the overall value and satisfaction score of the COTS solution being examined. The risk axis is assessed through goal mismatches, conflicting goals and involved risk scenario elements. Finally the cost axis is measured in terms of involved costs to deploy the solution. The in-depth examination of all these interdependent issues involved in the selection of a particular COTS solution, allows the evaluation team to make rational

Scenario Element	Explanation
Type of COTS intensive system	Describe which type of products constitute the solution: single COTS product, multiple COTS products, mixed off-the-shelf and custom built products
COTS products involved	Name all products integrated in the solution
Overall value of the solution	Describe which are the positive aspects of the proposed solution
Overall satisfaction score	Calculate the overall satisfaction degree of each COTS product using equation 6.28
Confidence Degree	Examine the confidence degree in which the proposed COTS intensive system meet the goals (see section 6.3.3)
Mismatched Goals	Describe all mismatched goals including the matching pattern and satisfaction degree (see table 6.2)
Conflicting Goals	Investigate how the proposed COTS intensive system affects conflicting goals
Involved Risks	Describe all risk events posed by the proposed COTS intensive system, use table 6.3 to guide the identification of risk events
Involved Costs	Estimate the costs to deploy the proposed COTS intensive system, this includes the cost to acquire, tailor, add glue code, integrate, etc. Use cost estimation techniques like CO-COTS [Abts et al., 2000] to perform this task

Table 6.5: Scenario 1 - Explore each COTS intensive system

decisions that are justified by well defined arguments.

A key criterion to judge the suitability of COTS solutions consists of examining the mismatches originated by the solution being evaluated. Table 6.6 describes a scenario to explore mismatches between goals and COTS solutions. To complement the usefulness of exploratory scenarios, we provide a set of heuristics to assist assessment and selection of appropriate risk resolution actions to handle mismatches.

H1. Examine the nature of the mismatch to justify its existence.

- Verify the degree of confidence in the satisfaction score assigned to the COTS product in order to judge the severity of the mismatch and decide if further information is necessary,
- Verify if the mismatch is caused because of initially high expectations then try to readjust the acceptable interval of the goal to tolerate mismatch.

- H2. Verify the priority given to mismatched goal in order to decide the relevance of the mismatch,
- If the mismatch involves low priority goals then a possible resolution is to discard the mismatch,
  - Reassess the priority given to mismatched goal and verify if it is possible to compromise or even to eliminate the goal to tolerate mismatch,
  - If the mismatched goal severely affects the overall success of the system then reinforce the importance to achieve the goal and choose other alternatives in order to successfully satisfy the goal.
- H3. If the mismatch is complete i.e. the product *fails* to meet the goal and the goal priority is high then choose other alternatives in order to successfully satisfy the goal.
- H4. If the mismatch is complete i.e. the product *fails* to meet the goal and the goal priority is low then try to compromise or eliminate the goal satisfaction in order to accept the mismatch.
- H5. If the mismatch is partial i.e. the product *differ* to meet the goal and the goal priority is high then try to negotiate product modification or if it is not possible, choose other alternatives in order to successfully satisfy the goal.
- H6. If the mismatch is partial i.e. the product *differ* to meet the goal and the goal priority is low then try to change constraints, compromise or eliminate the goal to tolerate the mismatch.
- H7. If the product *extends* the set of goals and the extra feature hurts other goals then try to hide the undesired functionality.
- H8. If the matching pattern of a goal cannot be identified then try to restructure the goal definition or try to obtain more information as means of handling uncertainty.
- H9. If the supplier has agreed to modify the product as a way to solve a particular mismatch then include specific contractual clause to certify the agreement.

The objective of exploratory scenario 3 is to facilitate the examination of risks originated by conflicting goals. As we have discussed in Section 6.2.4, a key issue to manage conflicting goals is the capacity to structure tradeoffs. To facilitate the exploration of potential conflict resolutions, we provide some heuristics to guide the tradeoff analysis of conflicting goals:

- H1. Examine the importance to achieve each conflicting goal to propose potential tradeoffs.

Scenario Element	Explanation
Goal Involved in the Mismatch	Describe mismatched goal
Goal Priority	Verify the composed weight to satisfy the goal, use equations 6.13 and 6.20
Satisfaction Degree	Verify the satisfaction degree in which the COTS solution meets the goal
Matching Pattern	Inform the matching pattern <i>differ, fail, extend</i>
Acceptance of Goal Dissatisfaction	Explore how stakeholders feel if the goal is not satisfied. The objective here is to assess if goal mismatch (i.e. its dissatisfaction) is acceptable or not
Risk Event	Describe the risk event originated by the mismatch, use table 6.3 to identify risk events
Risk Exposure	Measure the risk exposure by assessing the probability of risk occurrence and utility loss caused by the risk event, use equation 6.29 and figure 6.25
Potential Risk Resolutions	Investigate potential resolutions to manage the risk using the strategies described in figure 6.27
Involved Assumptions	Examine the necessary assumptions to make as a result of low confidence degree about product features, see section 6.3.3
Assess Risk Resolution Proposals	Assess the effectiveness of the risk resolution (i.e. its value) and cost to implement the proposed resolution, where possible costs includes: costs to change product capabilities, to add capabilities, to reflect changes to goals
Choose Risk Resolutions	Choose a suitable risk resolution or combination of resolutions to handle the risk originated by the mismatch
Justify Decisions	Provide the rationale to support the decision for a particular risk resolution proposal

Table 6.6: Scenario 2 - Explore Mismatch Situations

- If one conflicting goal has higher priority than the other goal then try to compromise the lower priority one,
- If the priority to achieve both conflicting goals is too high that stakeholders are not willing to tradeoff none of them then ensure that risks are communicated and shared among involved stakeholders,
- If the priority to achieve both conflicting goals is low and they do not interact with other goals then ignore the conflict,
- If one of the conflicting goals is decomposed through an OR link then give preference to satisfy the non-conflicting goal.

H2. Examine if the conflict between two goals depends on their satisfaction degree by

Scenario Element	Explanation
Conflict Event	Explain the conflict between goals
Conflicting Goals	Describe the goals involved in the conflict
Relative Importance between Conflicting Goals	Obtain the relative importance between conflicting goals by using the marginal rate of substitution technique described in section 6.2.5
Involved Risk	Describe the risk event originated by the conflict, use table 6.3 to identify risk events
Risk Exposure	Measure the risk exposure by assessing the probability of risk occurrence and utility loss caused by the risk event, use equation 6.29 and figure 6.25
Impact of COTS Intensive System	Assess how each COTS alternative affects the conflict, i.e. if it helps or hurts potential conflict resolutions
Involved Assumptions	Examine the necessary assumptions to make due to unverified issues such as low confidence degree about product features, see section 6.3.3
Potential Conflict Resolutions	Investigate potential resolutions to manage the conflict, where resolutions mainly consist of tradeoffs
Assess Resolution Proposals	Assess the feasibility of possible tradeoffs and analyse the impact each proposal gives to the overall goal satisfaction
Choose Conflict Resolutions	Choose a suitable resolution or combination of resolutions to manage the conflict
Justify Decisions	Provide the rationale to support the decision for a particular resolution proposal

Table 6.7: Scenario 3 - Explore Conflict between Goals

asking if the low/high satisfaction of a goal interferes with the satisfaction of another goal.

- Does the low/high satisfaction of one goal cease the conflict with the other goal?
- What is the conflicting threshold of the goal satisfaction degree?

H2. If the conflict is a type of terminology clash then restructure the definition of conflicting goals

H3. Given a severe conflict between goals, give preference to COTS solutions that resolve or minimise the conflict.

H4. If the conflict resolution is too complex or impractical to be fully addressed then discuss and share with stakeholders the impact the associated risks.

The exploratory scenarios described in Tables 6.5, 6.6, 6.7 guide the evaluation team to explore the worth of COTS alternatives by exploring their value, risk and cost. At



this stage, earlier decisions have to be reassessed and balanced to reflect results obtained from exploratory scenarios. This phase finishes when the evaluation team makes the final decision to select the COTS intensive system that brings the higher overall worth.

## 6.7 Summary

In this chapter we developed the TAOS method for addressing the requirements which a COTS selection method would need to address (as defined in chapter 5). TAOS uses a goal-oriented approach for eliciting and modelling the requirements of the system. TAOS provides guidelines and heuristics for conducting the refinement of goals and identifying their interactions. The method relies on utility theory to facilitate the prioritisation of goals and the further negotiation of conflicting goals. We have presented systematic guidance for the identification and evaluation of COTS candidates. The evaluation process results in the selection of the best COTS candidates. The shortlisted candidates are then screened to better understand their functionality and quality.

A key phase of the method is the matching process. Utility theory is used to model the satisfaction function of goals. This function measures the degree to which COTS candidates satisfy the goals of the acquirer organization. We have provided a set of matching patterns to precisely categorise the satisfaction degree of COTS products. The result of the matching phase is the identification of mismatches that may occur between COTS products and the requirements of the organization. We have provided a number of heuristics to analyse the nature and the impact of mismatches. Tradeoff analysis is performed to manage and resolve mismatches.

The analysis of mismatches can inform the existence of risks. We present a comprehensive strategy to examine and manage these risks. We use the risk exposure technique to quantify the severity of the risk based on the probability of occurrence and utility loss caused by the risk. To aggregate the results of the various steps of the method, we developed a set of templates to build exploratory scenarios. The objective of these scenarios is to facilitate the decision making and to assess the overall worth of each COTS candidate.

## Chapter 7

# Evaluation

In the previous chapter we have described the TAOS method to guide the evaluation and selection of COTS products. One of the key objectives of the method we have developed is supporting the analysis and resolution of mismatches that may occur between customer requirements and COTS products. In this chapter we evaluate the feasibility and usefulness of TAOS. Evaluating methods like TAOS is rather hard, as their effectiveness is extremely dependent on the way practitioners apply them. Practitioners generally have to tailor software engineering methods to address the needs of specific projects and suit different environments. In addition, the nature of decisions made when selecting COTS products fundamentally varies from organization to organization. This means that some selection projects may not need all the prescribed actions, artifacts and guidelines defined in TAOS, while others need to follow all TAOS activities rigorously. Different projects will use TAOS in different ways simply because their problems and goals are different. As a result, the validation of the method is subject to the context in which the method is being applied. While the results obtained from the evaluation effort presented in this chapter demonstrate the suitability of the method for the classes of problems in which the method has been examined, it is not possible to guarantee that similar results will be found on other projects.

According to [Dawson et al., 2003], conducting controlled and repeatable experiments in software engineering is a quite difficult (if not impossible) task to accomplish. This is mainly due to the fact that the way software engineering methods are applied varies in different contexts and variables involved cannot be fully controlled. The chosen strategies to evaluate TAOS method are through case studies and critical comparison with other methods. Case studies have been extensively used to empirically assess software engineering approaches [Maciaszek and Liong, 2004]. When performed in real situations, case studies provide practical and empirical evidence that a method is appropriate to solve a particular class of problems. Since case studies allow little or no replication, it is difficult to produce meaningful results that can be generalised [Yin, 1984]. Nonetheless, we consider

## Chapter 7

that case studies are the most appropriate approach to evaluate “soft” methods like TAOS. To establish the effectiveness of TAOS we have conducted three forms of evaluation:

*Industrial case study* - We have applied TAOS in a large case study involving the acquisition of a electronic document and records management system by University College London.

*Simulated case study* - We have conducted a comprehensive evaluation of TAOS method in the context of assessing mail server packages. Though this was a fictitious case study, which is a non-optimal means of evaluation, we were able to get valuable insight on the effectiveness of the method.

*Comparison with other COTS selection methods* - We have performed a critical comparison of TAOS against other COTS selection methods proposed in the literature. The chosen methods were PORE, OTSO, CARE and PECA, which are widely accepted approaches by researchers and practitioners.

The above list is stated in decreasing order of evaluation strength, evaluation through application on a real industrial project being the strongest and comparison of methods being the weakest form of evaluation. In order to control the results of the evaluation effort, we conducted the case studies following the DESMET methodology [Kitchenham et al., 1997] for guiding the evaluation of software engineering methods. The authors state that the first decision to make when undertaking a case study is to determine what the study aims to investigate and verify, in other words, to define the goals of the case study. In the context of evaluating the TAOS method, the goal we want to achieve is to demonstrate the validity of the following hypothesis:

*TAOS provides a systematic method to improve the quality of decisions made during the selection of COTS products.*

In order to test the validity of the stated hypothesis, it is necessary to precisely define the effects we expect the method to have as well as clearly identify the measurements needed to demonstrate the desired effects. In the following we explicitly define the criteria under which we will evaluate the results of the industrial case study:

1. Does TAOS provide complete, understandable and useful method to guide the evaluation of COTS products?
2. Is Goal-based requirements engineering a suitable approach to specify requirements for COTS-based systems?

3. Does the matching process provide an effective way to evaluate to which extent a COTS product satisfies the requirements of the acquirer organization?
4. Is the risk management strategy effective to identify, analyse and resolve risks that may arise during COTS-based development?
5. Do exploratory scenarios provide useful guidance to make informed selection decisions?
6. Do the heuristics help guiding mismatch and conflict analysis facilitate the tradeoff process?

In the following section we describe the mail server case study. The main objective of this study is to simulate each phase of the TAOS method in detail in order to demonstrate the method's applicability.

## 7.1 Mail Server Case Study

### 7.1.1 Background

Nowadays, organizations of any size increasingly depend on electronic mail to support their communication and coordination. According to the Radicati Group [Rad, ], worldwide revenue for the messaging software market grew by nearly 10% totalling 2.85 billion dollars in 2004. Consequently, competition in this market has intensified in the last few years, especially in the corporate messaging software market, where large corporations such as IBM and Microsoft have battled for dominance in this increasingly profitable market. Electronic messaging services continue to evolve into a critical business application which requires broad requirements for collaboration, and must satisfy growing security and privacy concerns. The electronic messaging infrastructure comprises a number of applications such as mail servers, mail clients, anti-virus and anti-spam tools, collaboration tools, backup and recovery facilities, etc. Each of these applications can be supplied individually by different vendors such that customers have to procure and integrate the desired applications or a number of these functionalities can be available in the market already integrated by a single supplier. It is important to note that even when customers adopt an integrated messaging product, they may still want to buy specific products providing functionality that is inadequately supported by the integrated product.

Electronic mail is an asynchronous messaging technology built over a distributed client-server architecture. Figure 7.1 illustrates the basic way in which messages are transmitted over the Internet. Let us suppose that Alice wants to send an email to White Rabbit. Alice composes a message using her client program, which is called mail user agent (MUA), then Alice's MUA uses the Simple Mail Transfer Protocol (SMTP) to send the message to

the local mail transfer agent (MTA). The MTA looks at the destination address, in this case White Rabbit's e-mail address. Now, in order to White Rabbit receive the message, Alice's local MTA needs to relay the message to one or more MTAs. When the message arrives to its destination (i.e. White Rabbit's server), the final MTA delivers the message to the appropriate mail message store (MS), from where White Rabbit can access it from his mail client.

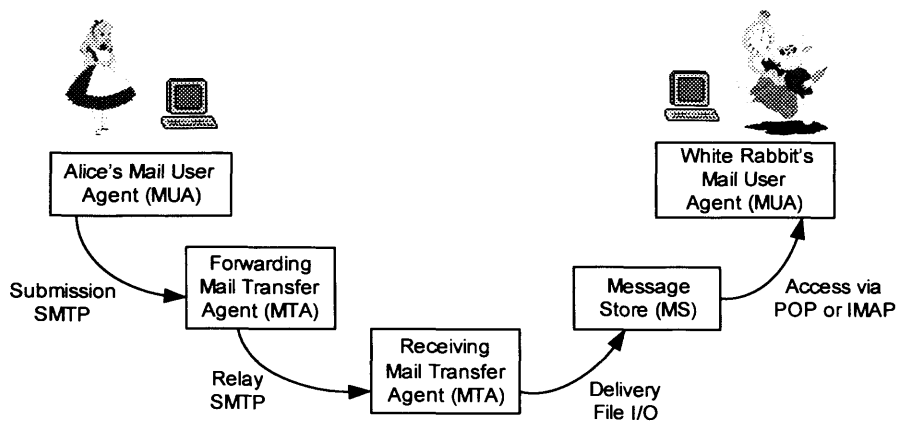


Figure 7.1: Basic electronic mail system

Electronic messages were originally designed to handle only text messages in simple 7-bit ASCII format. To overcome this limitation, the Multipurpose Internet Mail Extensions standard (MIME) has been adopted to allow the encoding of binary attachments including images, sounds and HTML attachments. The format of Internet e-mail messages is defined in RFC 2822 [RFC, ]. Email messages commonly consist of two major components:

- Headers : Message summary, sender, receiver, and other information about the e-mail
- Body : The message itself, usually containing a signature block at the end
- From : The e-mail address of the sender of the message
- To : The e-mail address of the receiver of the message
- Subject : A brief summary of the contents of the message
- Date : The local time and date when the message was originally sent
- Cc : Carbon copy
- Bcc : Blind carbon copy
- Received : Tracking information generated by mail servers that have previously handled a message
- Content-Type : Information about how the message has to be displayed, usually a MIME type

Mail clients are programs for viewing and composing emails. Mail clients usually retrieve email from MS using either the Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP). IMAP has a number of benefits over POP, for instance: support for both connected and disconnected modes of operation (i.e. in connected mode the messages are kept on the server and downloaded remotely by the email client, while the disconnected mode the messages are left on the server but a copy is available in the local machine), support for access to MIME parts of messages and partial fetch, support for multiple clients simultaneously connected to the same mailbox. Note, however, that POP has the advantage of being a much simpler and more available protocol than IMAP. There are a large availability of mail clients in the market, they can be classified in three main groups:

- Fat email clients : specialized programs such as Eudora, Outlook, Mozilla mail and Lotus notes.
- Web-based email clients : also known as Webmail, e-mail messages are accessed and sent by using Web browsers, examples include free programs Hotmail and Gmail.
- Mobile email clients: email programs such as BlackBerry, GoodLink and SnapperMail allow sending and receiving messages from mobile devices.

Electronic messaging systems face key threats caused by spamming and viruses. Spamming is unsolicited commercial e-mail. According to a large e-mail service provider called Postini [Postini, ], around 70% of corporate e-mails are spam. The large number of spam can cause email systems to experience unexpected overload in bandwidth, server storage capacity, and loss of end-user productivity. Viruses are more aggressive than spam since many viruses are designed to compromise the e-mail system security and reliability. The combination of spam and virus programs results in considerable economic and productivity losses for organizations. Therefore, the use of anti-virus and anti-spam tools is greatly important in order to protect users against threats caused by viruses and spams. Another increasingly important concern involved in email systems is the privacy of messages. Privacy risks are caused because of email messages have to go through intermediate computers before reaching their destination, meaning it is relatively easy for others to intercept and read messages. Security mechanisms such as cryptographic techniques can serve as a remedy to privacy issues.

We have explored in this section the main concepts, characteristics, and mechanisms involved in electronic messaging systems. Now, we discuss how the mail server case study was conducted with the purpose of evaluating the TAOS method.

### 7.1.2 Methodology and Case Study Artifacts

This case study was inspired on work from [Franch and Carvalho, 2002, Carvalho et al., 2003]. They proposed a methodology to build quality models based

on the ISO/IEC 9126 standard. In [Carvallo et al., 2003], they define a comprehensive quality model with characteristics and attributes related to the mail server domain. In [Alves et al., 2005], we present a joint research that investigated the suitability of quality models to facilitate the elaboration of goals for COTS-based systems. It is important to note that the work presented in this thesis is fundamentally different from earlier work because our objective now is to examine the applicability of TAOS to support the evaluation and selection of mail server products. Nevertheless, we use knowledge expressed in the mail server quality model as the basis for this case study. This has helped us in understanding quality attributes involved in the mail server domain.

The specific objective of this case study is to investigate how TAOS can be applied by organizations to guide the evaluation and selection of mail servers. Even though this case study was not conducted on a real COTS selection project, which means that the results may not be a conclusive account of the method effectiveness to guide the selection of this class of software systems. We still consider that the knowledge gained by conducting this simulation case study represents an important piece of evidence [Dawson et al., 2003] regarding the applicability of TAOS. In contrast to the way we have conducted the EDRM case study, in which we could not present a detailed description of the evaluation effort due to the project's confidentiality constraints, in the mail server case study we provide a comprehensive explanation on how TAOS is applied in practice. In order to conduct this case study as objectively as possible, we have to make a number of assumptions. Firstly, we assume that the organization wishing to acquire a new mail server is a medium size private bank, having its applications running under Microsoft Windows operating system. Secondly, the bank currently uses Microsoft Exchange version 5.5 as messaging system. Thirdly, this case study is presented from the evaluation team's point of view whose role is to evaluate candidate mail servers and choose the one that better suits the bank's requirements. The main reason the bank wants to acquire a new messaging package is because Microsoft is phasing out Exchange v5.5, which means that the software will no longer be supported by Microsoft. Hence, the bank has been forced to upgrade their messaging system to ensure complete support from the supplier. In the next sections we follow each one of TAOS phases in order to evaluate mail servers available in the market.

### 1. Specify Goals

To start applying TAOS, we had to understand the key objectives and constraints of the selection project. An obvious issue to be considered is the decision to continue using a Microsoft solution by updating the current system to a new version of the Exchange mail server or to migrate to a competitor supplier. The implications involved in this decision will deeply affect the whole evaluation process. The high level goals our fictitious bank wants to satisfy with the new mail server is shown in Table 7.1. It includes a list of functional and non-functional goals and respective subgoals. Some subgoals defined at this level are

already well-defined and measurable, such as  $g_{5.1}$ ,  $g_{5.2}$ ,  $g_{5.3}$ ,  $g_{5.4}$  where the metric associated to assess the satisfaction of these subgoals is a simple boolean [*support*, *not support*]. While other subgoals still need further refinement to allow their measurement, examples of such subgoals include  $g_{2.1}$ ,  $g_{2.2}$ ,  $g_{2.3}$  which are subgoals of  $g_2$  *ensure that messages must never get lost*. In order to properly evaluate subgoals like these ones, we need to obtain further information to better understand them. During this refinement process the evaluation team is likely to raise some of the following questions: what is the average size of users mailbox? what is the maximum size of inbound and outbound message queues? what are the usual mail clients used by bank staff? are the messages stored on the server or on the client? These questions can lead to new refinements for the given subgoals, and consequently provide a more complete description on how to achieve goal  $g_2$ . In addition, the examination of domain knowledge such as the mail server quality model described in [Carvalho et al., 2003] combined with the assessment of mail servers documentation were valuable sources of information. This effort enabled us to start exploring possible solutions to implement  $g_2$ . For instance, by examining documentation we observed that server clusters are common mechanisms mail servers use to eliminate single points of failure and provide alternative access to message stores. Hence, server clustering may be one of the possible ways to handle system failure and reduce the probability of losing messages.

According to TAOS, the refinement of goals continues until it is possible to assign metrics to lower level subgoals. From the initial list of 12 high level goals, we have refined the goal specification into 72 operational goals, each of them associated with appropriate metrics. An important comment is that it was difficult to associate precise metrics to some goals and even more difficult was to evaluate how mail server candidates satisfy them. In particular, we found that evaluating the satisfaction of  $g_{3.2}$  *message throughput rate should be less than 5 minutes per megabyte* depends on several environmental parameters such as system platform, number of users, number of concurrent access. We need, therefore, to define these parameters in order to assess the satisfaction of  $g_{3.2}$ .

Following the definition of operational goals, we have established the acceptable interval for each operational goal with respective satisfaction functions. Figure 7.2 illustrates the satisfaction functions for operational goals  $g_{3.1}$  *The average response time should not exceed 1 minute*,  $g_{5.2}$  *support IMAP4*, and  $g_{7.1.1}$  *enable directory methods*. Note that  $g_{7.1.1}$  represents a further refinement of high level goal  $g_7$  *ensure data security*. In this example, each operational goal is associated to different types of metrics: integer, boolean and fixed set, respectively. Now that satisfaction functions for these operational goals are defined, it is possible to start evaluating the extent to which mail servers satisfy them. Depending on the number of operational goals, the definition of functions can be a time consuming activity. To control the effort needed, we may opt to focus on explicitly defining satisfaction functions for those goals that are either critical, difficult to assess or help distinguishing candidate products, while disregarding the definition of satisfaction function for other less important goals. Obviously that the risks caused by poorly understanding of goals have



Goal description	Subgoals
<i>g</i> <sub>1</sub> Ensure and communicate message delivery	<i>g</i> <sub>1.1</sub> Configure number of delivery retries <i>g</i> <sub>1.2</sub> Configure time between retries <i>g</i> <sub>1.3</sub> Provide message delivery notification
<i>g</i> <sub>2</sub> Ensure that messages never get lost	<i>g</i> <sub>2.1</sub> Messages must never get lost if mailbox runs out of space <i>g</i> <sub>2.2</sub> Messages must never get lost if a failure happens <i>g</i> <sub>2.3</sub> Messages must never get lost if they cannot be delivered
<i>g</i> <sub>3</sub> Ensure fast message delivering	<i>g</i> <sub>3.1</sub> The average response time should not exceed 1 minute <i>g</i> <sub>3.2</sub> Message throughput rate should be less than 5 minutes per megabyte
<i>g</i> <sub>4</sub> Support collaborative work	<i>g</i> <sub>4.1</sub> Integrated document management <i>g</i> <sub>4.2</sub> Instant messaging <i>g</i> <sub>4.3</sub> Voice and video conferencing
<i>g</i> <sub>5</sub> Support common communication protocols	<i>g</i> <sub>5.1</sub> Support POP 3 <i>g</i> <sub>5.2</sub> Support IMAP 4 <i>g</i> <sub>5.3</sub> Support HTTP <i>g</i> <sub>5.4</sub> Support SMTP
<i>g</i> <sub>6</sub> Support for web access	<i>g</i> <sub>6.1</sub> Support Webmail <i>g</i> <sub>6.2</sub> Support Web browser
<i>g</i> <sub>7</sub> Ensure data security	<i>g</i> <sub>7.1</sub> Authentication of users <i>g</i> <sub>7.2</sub> Data integrity
<i>g</i> <sub>8</sub> Support automatic subscription to mail lists	No Subgoal
<i>g</i> <sub>9</sub> Support protection against external attacks	<i>g</i> <sub>9.1</sub> Provide anti-spam filters <i>g</i> <sub>9.2</sub> Provide anti-virus scanning
<i>g</i> <sub>10</sub> Support middleware	<i>g</i> <sub>10.1</sub> Support DCOM <i>g</i> <sub>10.2</sub> Support CORBA <i>g</i> <sub>10.3</sub> Support RMI
<i>g</i> <sub>11</sub> Verify the maturity of the product in the market	<i>g</i> <sub>11.1</sub> Version Stability <i>g</i> <sub>11.2</sub> Vendor reputation
<i>g</i> <sub>12</sub> Installation and administration facilities	<i>g</i> <sub>12.1</sub> Adm tools and wizards <i>g</i> <sub>12.2</sub> Documentation available

Table 7.1: Goal Specification for Mail Server

to be balanced against the benefits of reducing the time needed to accomplish the task of defining satisfaction functions.

The next activity proposed by TAOS is the identification of interactions between goals. Figure 7.3 shows the goal refinement tree with some positive and negative interactions between goals. For example, it was quite straightforward to identify the positive interaction between *g*<sub>11.1</sub> *version stability* and *g*<sub>12</sub> *installation and administration facilities*. The implications of positive interactions will be observed during the assessment of candidate

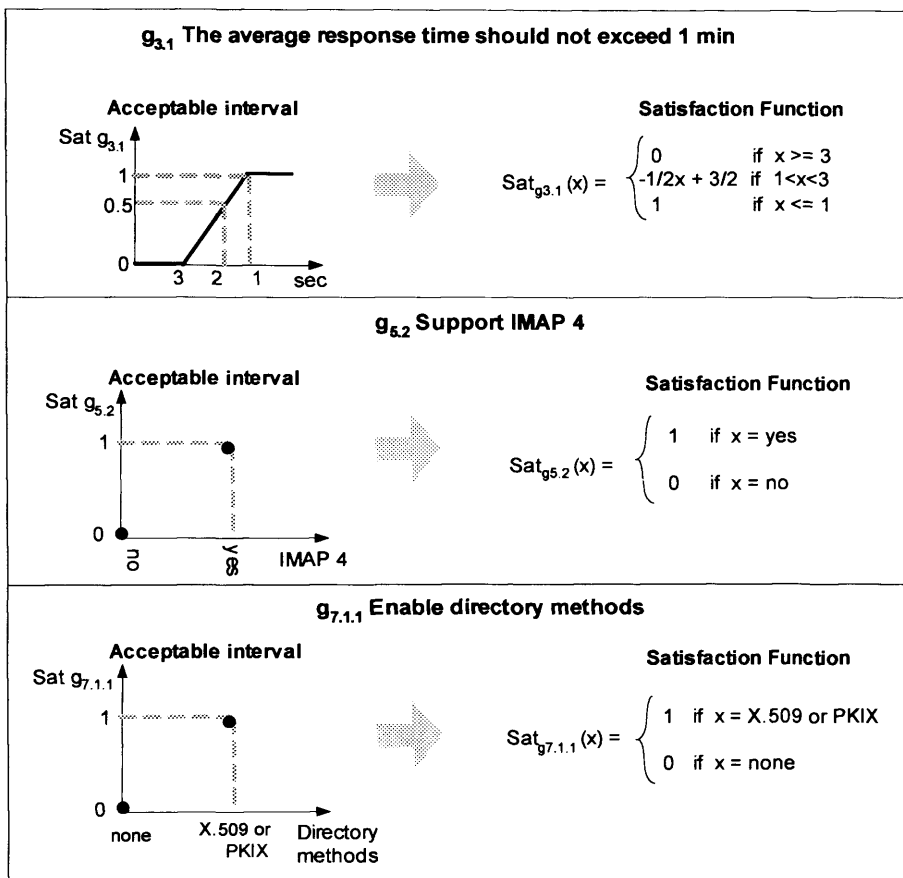


Figure 7.2: Acceptable interval and satisfaction function for operational goals  $g_{3.1}$ ,  $g_{5.2}$ ,  $g_{7.1.1}$

mail servers. In this case, it is more likely that a mail server which achieved a stable version would be easier to install and administrate than another product with an unstable version. The reason for that is because stable versions of products are less prone to failures and bugs, consequently helping system administrators to install and configure the mail server. An example of negative interaction occurs between goals  $g_{2.2.1}$  *backup and recovery strategies* and  $g_3$  *fast message delivery*. The reason for this negative interaction is because backup and recovery strategies cause system overhead, and hence delaying the time of message delivery. In order to handle this negative interaction, we need to assess the importance to satisfy each conflicting goal and tradeoff the satisfaction of the less important goal for the other. Another example of negative interaction occurs between  $g_{6.2}$  *support webmail* and  $g_7$  *ensure security*. We identified this negative interaction by reading security reports explaining how webmail applications may cause security vulnerabilities. An important observation is that the identification of interactions between goals will inform the evaluation of mail servers because dependencies and conflicts will ultimately have an impact on decisions made to satisfy interrelated goals.

Another useful strategy to help the decision process is the prioritization of goals. By applying the goal prioritization technique described by TAOS, we were able to compare the relative importance to achieve goals and obtain relative weights for them. To explain the prioritization process, let us take as example how the weights were obtained for goals  $g_{4.1}$  *integrated document management*,  $g_{4.2}$  *instant messaging*, and  $g_{4.3}$  *voice and video conferencing*. The first step described by TAOS is to obtain inequalities between goals. Given the nature of activities performed by our bank, the most important collaboration facility is to allow documents distributed across branches to be accessed by staff from other branches. The second most important collaboration goal the bank wants to achieve with the mail server solution is to allow managers from different branches to have meetings by using voice and video conferencing facility. Finally, the least important goal is instant messaging since the primary duty of bank employees is to serve customers rather than interact with other employees. From these observations, we obtain that  $\mu_{g_{4.1}} \succ \mu_{g_{4.3}} \succ \mu_{g_{4.2}}$ . The next step is to assign numerical weights to each goal, then we assume that  $\mu_{g_{4.1}} = 0.5$ ,  $\mu_{g_{4.3}} = 0.4$  and  $\mu_{g_{4.2}} = 0.1$ . Once these steps are followed to all goals, it is possible to obtain the composed weight for goals. The composed weight of a particular goal embraces the weight of the goal itself together with the weights of all parent goals. For instance the composed weight of  $g_{4.1}$  is  $\omega_{g_{4.1}} = \mu_{g_0} \times \mu_{g_4} \times \mu_{g_{4.1}}$ , then we have  $\omega_{g_{4.1}} = 1 \times 0.2 \times 0.5$ , and finally  $\omega_{g_{4.1}} = 0.1$ .

## 2. Assess Products

Given the maturity and size of the electronic messaging market, it was easy to identify mail server products. We found a large number of mail server suppliers, ranging from small companies offering simple mail systems including few features to large corporations

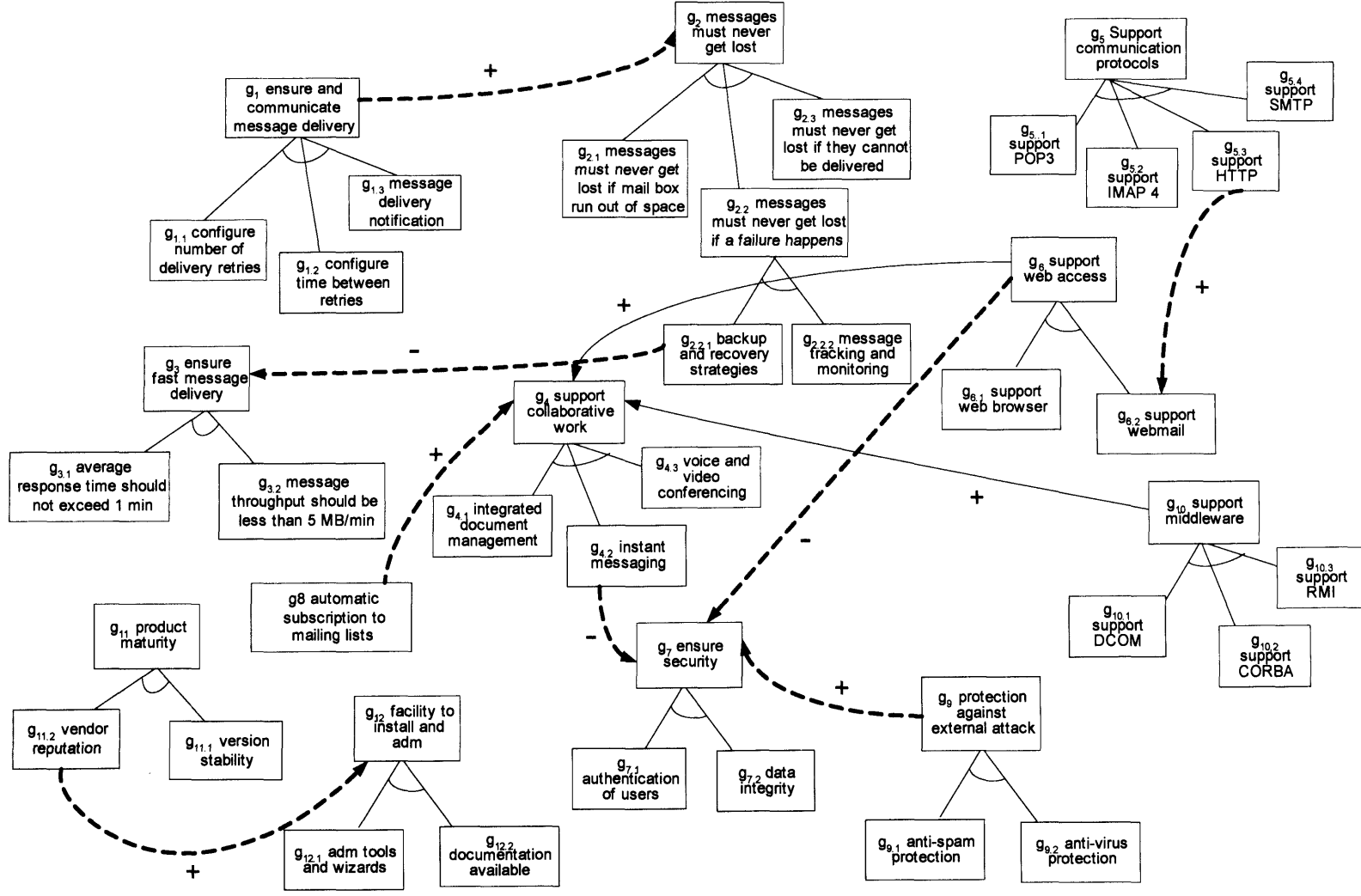


Figure 7.3: Interaction between goals for the Mail Server system

providing complete messaging and collaboration infrastructure. As we have discussed in the beginning of the case study, we assume that the bank is currently running the outdated 5.5 version of Microsoft Exchange mail server and wishes to replace it with a state-of-the-art product. A decision that will drive the selection process is the choice of upgrading Microsoft Exchange v5.5 to Exchange v2000 or migrating to a competitor supplier. According to market analysis consultants [Rad, ], the main competitor of Exchange 2000 is Lotus Domino R5. It is important to note that when this case study was performed two years ago these products were the main players in the electronic messaging market. Besides the reputation and maturity of suppliers, another critical aspect to shortlist candidate mail servers is the platform constraint for the system to run on Windows operating system. This is a non-negotiable constraint since it would be extremely costly to migrate the whole hardware platform. As a consequence, all products running only on Unix had to be eliminated from the product list.

By considering the platform constraint together with the strategic decision to select a market leading product, we concluded that the best alternatives to be shortlisted are Exchange 2000 and Lotus Domino R5. Microsoft offers Exchange as the server and Outlook as the client, and IBM Lotus offers Domino as the server and Notes as the client. Even though other mail clients are also supported by the suppliers, this is the typical client-server configuration. Exchange is primarily a messaging system that includes limited application development capability, while Domino has a reputation to be a richer collaborative system with strong application development environment but average messaging capability. It is important to note that our objective is not to perform a comparative analysis between products, this topic has been extensively covered by comparison reports available on the Internet. Instead, we concentrated the evaluation effort to verify how candidate mail servers satisfy the set of goals described in Table 7.1. A key limitation of this case study is that it was not possible to install the mail servers in order to conduct test cases. The evaluation effort solely considered commercial and technical documentation of mail servers available on the suppliers website as well as comparative reports from third-party organizations.

### 3. Perform Matching

We analysed the matching between mail server features and operational goals by examining available information about Exchange 2000 and Domino R5 and mapping their features to operational goals. For example, one of the refinements to achieve goal  $g_{1.3}$  *provide message delivery notification* is the operational goal  $g_{1.3.1}$  *message tracking and monitoring*. We obtained from Exchange 2000 documentation the following statement concerning the way it implements message tracking and monitoring: “*The Message Tracking Center windows can trace a message through its path. The log includes the point it originated, the point it was delivered, the date and time a message was sent, information about the sender, the sender’s*

*IP address and host name, the number of recipients, and their addresses, the message ID, the message subject, the message priority, and whether the message was encrypted*". Based on this information, we had to decide whether  $g_{1.3.1}$  is fully satisfied or not, and then associate the appropriate satisfaction degree to the goal. We followed the same process to check the satisfaction of other operational goals. In order to do that, we drew a table with the list of operational goals and tried to map related functionality described in the products documentation. This task demonstrated to be quite hard because of inappropriate and marketing-oriented vocabulary used by suppliers. In an ideal situation, we would request supplier representatives to clarify ambiguous aspects and demonstrate uncertain features.

An example of *extend* mismatch is caused by a feature offered by Exchange called presence information which enables users to see whether other users are logged in the system. This functionality was not requested in the goal specification and it may hurt privacy concerns of users who do not want to be seen online. Therefore, to control this unwanted outcome, users should be able to disable the functionality as they wish. Later we discovered that the presence information feature can be fully configured, and hence we concluded that the nature of the *extend* mismatch is neutral.

Given the limited sources of information, we were unable to verify how mail servers satisfy a number of operational goals. For instance, it was not possible to obtain objective information concerning the satisfaction of operational goals  $g_{3.1}$  and  $g_{3.2}$ . We considered the matching pattern is *uncertain* for both operational goals. Performance tests are quite a tedious and costly task because performance results greatly depend on platform and system configuration parameters. In order to check whether the mail server's performance is under acceptable levels, we may simply decide to rely on widespread performance benchmarks available on the Internet, instead of conducting the performance test cases themselves. By following this strategy we could save substantial time in evaluating products. On the other hand, its drawback is the limited reliability on benchmark results, especially the ones conducted by mail server suppliers. We observed that information present on product specification is often biased and contradictory. This situation is a direct effect of the aggressive battle between Microsoft and IBM over the electronic messaging market dominance, in which suppliers must convince customers of their superiority in order to increase market share.

When evaluating mail servers documentation, we observed that both suppliers offer several extra modules that can be integrated with the standard mail server product to satisfy specific functionalities. For instance, in order to satisfy  $g_{4.3}$  *voice and video conferencing* it is necessary to obtain extra software modules from both suppliers. In this case, the matching is *fail* and the rationale for the mismatch is that even though the product can satisfy the operational goal, this functionality is not available in the standard package, which means extra costs to acquire the voice and video conferencing module. Similarly, Exchange *fails* to satisfy operational goal  $g_{4.1}$  *integrated document management*, while Domino *fulfills* it. We observed that Exchange and Domino offer very limited anti-virus

and anti-spam capabilities, such that these products are commonly integrated with specific tools to fully support these functions. As a result, the matching pattern between mail server candidates and operational goals  $g_{9.1}$  and  $g_{9.2}$  is *fail*.

According to our comparison effort, Exchange and Domino obtained similar ranking in terms of goal satisfaction. Both suppliers provide equally powerful mail servers with comprehensive messaging and collaboration capabilities, hence satisfying most goals defined in table 7.1. One of the key differentiating factors between the products refers to Exchange's ability to employ native services provided by Microsoft Windows operating system, such as clustering and active directory services. This ability enables Exchange to leverage system scalability and reliability. While Domino R5 uses its own clustering and directory services that have to be synchronised with Windows services, this is a suboptimal strategy that tends to be more difficult to manage. On the other hand, Domino R5 performs much better than Exchange in satisfying goal  $g_4$  *support collaborative work*. Lotus/Domino mail server is built on top of Lotus Notes system, which is a well-established and mature workgroup tool. To consolidate these preliminary results, we needed to examine the impact of mismatches caused by both products to verify if they pose any risks and help the comparison between mail server candidates.

#### 4. Analyse Mismatches and Manage Risks

As we discussed before, the ranking results did not allow us to distinguish a superior product. Therefore, it was necessary to examine mismatches as a way to establish which mail server is the preferred option, and hence inform the decision-making process. In particular, one of the critical mismatches we have identified refers to the mail servers insufficient support to anti-virus and anti-spam capabilities. By following the risk identification template provided by TAOS (see Table 6.3), we obtained information concerning the risks caused by this mismatch as shown in table 7.2.

Given the severity of consequences caused by a security attack together with the high probability of this event to happen since mail servers fail to support anti-virus and anti-spam facilities, we consider that the exposure of this risk is high. As a result, we needed to properly handle it and ensure that the system will be effectively protected against external attacks. According to the risk management taxonomy described by TAOS (see Figure 6.27), an appropriate risk action to mitigate this type of risk is the acquisition of additional products. Following this decision, the final system will consist of an integrated solution of mail server, anti-virus and anti-spam tools. In order to achieve that, we had to define a set of goals to drive the evaluation of anti-virus and anti-spam tools. An obvious killer selection criterion is the tools' ability to successfully integrate with the chosen mail server. Note that, as shown in Figure 7.4 it may be possible to have different acceptable configurations for the final system. In all described configurations, the mail server is always

Element	Example
Risk factor	Mail server fails to support anti-virus and anti-spam facilities.
Effects of risk factor	Spam can cause unexpected overload in bandwidth, server storage capacity, compromise and loss of end-user productivity. While viruses are more harmful. They can threaten the security of the system from simply allowing the attacker to view sensitive information, whether by examining network traffic or by getting read-only access to administrator or system files to allowing attackers gain complete control of the target system and do virtually any amount of damage that a fully authorized system administrator can do.
Acceptability of risk factor	<i>g<sub>9</sub></i> Support protection against external attacks is a critical goal, this means that it cannot be compromised.
Risk Event	Mail server can be easily attacked by external threats.

Table 7.2: Identifying risk events

the core component that will drive the evaluation of complementary components. Hence, decisions made during the evaluation of mail server will affect and possibly constrain decisions regarding anti-virus and anti-spam tools. More specifically, the anti-virus and anti-spam tools will have to comply with the architecture and platform of the mail server.

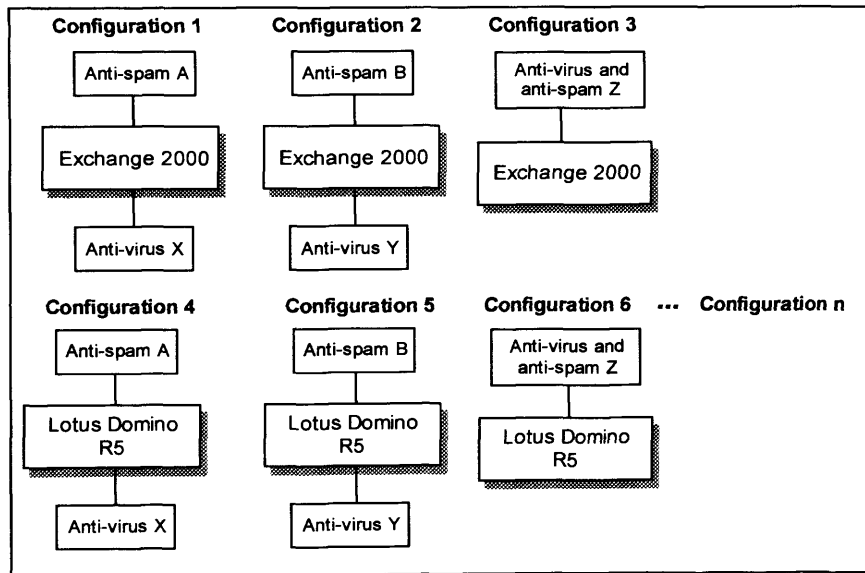


Figure 7.4: Possible configurations of integrated system composed by mail server, anti-virus and anti-spam tools

Concerning the mismatch between *g<sub>4.3</sub>* voice and video conferencing and both mail servers,



we decided to wait until the mail server is fully deployed to satisfy this feature (i.e. no risk mitigation action). Another interesting mismatch occurred between  $g_{4.1}$  *integrated document management* and Exchange 2000. The standard Exchange mail server product *fails* to support the management of integrated documents. However, it is possible to acquire an additional package from Microsoft called Workflow Designer to support this functionality and many other workflow applications which are not particularly desired by the bank. Domino, on the other hand, satisfies  $g_{4.1}$  as a built-in functionality (i.e. Lotus/Domino *fulfils*  $g_{4.1}$ ). In order to explore the mismatch caused by Exchange and establish which is the preferred solution offered by mail servers, we need to assess how damaging is the dissatisfaction of goal  $g_{4.1}$  and understand the risks involved with this outcome. During the goal specification phase we obtained that  $\omega_{g_{4.1}} = 0.1$  and  $\mu_{g_{4.1}} = 0.5$ , which are the composed and single weights respectively. These weights were necessary to measure the overall satisfaction degree mail servers satisfy goals. Now, we have to ask how stakeholders feel if  $g_{4.1}$  is not satisfied. If they say that failing to satisfy  $g_{4.1}$  is acceptable and the impact over the final system satisfaction is minor, then an appropriate risk handling action is to compromise the goal, and hence accept Exchange limitation. Otherwise, if stakeholders perceive that not satisfying  $g_{4.1}$  will cause major risks to the system acceptance then suitable actions to mitigate this risk would be either to give preference to Lotus/Domino or to acquire the Workflow Designer package to complement Exchange's functionality. Let us assume that stakeholders definitely want to satisfy  $g_{4.1}$ . In order to decide whether to prefer Lotus/Domino or Exchange, we had to consider several issues such as: Workflow Designer acquisition and integration costs, possible risks involved with the integration of Workflow Designer, and value added by Workflow Designer.

## 5. Select COTS Solution

The final step of the evaluation process consisted of recommending the mail server that brings the highest worth to the bank. According to TAOS, the overall worth of a particular COTS product consists of assessing the product's value against its risks and costs. We observed that both products obtained similar ranking regarding the satisfaction of goals (i.e. similar overall value). One of the few mismatches found was their failure to support anti-virus and anti-spam facilities. Based on rationale arguments, it was decided that extra products should be acquired to fulfil these features. We found a large variety of anti-virus and anti-spam tools available in the market that can be integrated either with Exchange or Lotus/Domino. As a result, this mismatch did not help discriminating products. Another mismatch examined was the one between  $g_{4.1}$  and Exchange, we found that one possible resolution to handle this mismatch is to acquire Workflow Designer package. After examining the risks and costs associated with the acquisition of this tool, we concluded that the preferred option would be to select Lotus/Domino instead of Exchange. The exploratory scenario described in Table 7.3 presents all the issues involved to reach this decision.

Scenario Element	Explanation
Goal Involved in the Mismatch	$g_{4.1}$ integrated document management
Goal Priority	$\omega_{g_{4.1}} = 0.1$ and $\mu_{g_{4.1}} = 0.5$
Satisfaction Degree	$Sat_{g_{4.1}}(Exchange) = 0$
Matching Pattern	Fail
Acceptance of goal dissatisfaction	Even though this goal is not critical, stakeholders do not want to compromise it.
Risk Event	The collaboration among staff from different branches will be compromised.
Risk Exposure	The probability of risk occurrence is very high because of the fail mismatch between Exchange and $g_{4.1}$ . The severity of risk event is also high because the ability to actively manage documents shared by different users is perceived to be a fundamental feature to build the bank's collaboration infrastructure.
Potential Risk Resolutions	Acquire Workflow Designer package to complement Exchange 2000 or give preference to select a competitor mail server that fulfils $g_{4.1}$ , in this case Lotus/Domino R5.
Involved Assumptions	Due to time constraints, it was not possible to fully explore the risks involved with the acquisition of Workflow Designer. We had to rely on information available on the supplier website.
Assess Risk Resolution Proposals	Given the limited budget allocated for the mail server selection, opting for a built-in solution seems to be more adequate than acquiring extra package to satisfy $g_{4.1}$ .
Choose Risk Resolutions	Prefer Lotus/Domino R5 instead of Exchange 2000.
Justify Decisions	Since Lotus/Domino R5 satisfies $g_{4.1}$ as part of its standard mail server solution, this alternative is more cost-effective than acquiring Workflow Designer package.

Table 7.3: Exploratory Scenario - Mismatch between  $g_{4.1}$  and Exchange 2000

A key discriminative factor to assess the costs and risks involved with the acquisition of each product is the strategic decision of upgrading the mail server from Microsoft Exchange v5.5 to Exchange v2000 or migrating to the competitive platform Lotus/Domino R5. In order to compare the costs involved with both alternatives, we had to consider the following cost elements: labour necessary to install, configure and migrate data, software licences, hardware, training, and potential downtime costs. On the one side, Microsoft offers attractive discounts for current customers to upgrade their Exchange system. Exchange also requires lower training costs when compared to Domino since users are already familiar with Exchange system. On the other hand, Domino typically requires less new hardware investment than Microsoft recommends for an equivalent Exchange 2000 upgrade.

In terms of risks, upgrading the messaging infrastructure to Exchange 2000 is likely to be less complex, and hence would bring fewer risks than migrating to a different platform. In addition to making major changes to the IT infrastructure, the migration project can affect the way the bank does business, as well as cause cultural changes within the organization. Cultural changes can be very difficult to be accepted by users. For that reason, upgrading the system to a platform which users are familiar is expected to bring fewer changes. In order to make the final decision to establish the preferred mail server product for the bank, we had to analyse and balance all these issues against each other. At the end, after performing a number of tradeoffs, we decided that the benefits brought by Exchange 2000 pays off the limitations of the product. An example of tradeoff that we had to perform refers to the need to acquire the Workflow Designer tool as part of the Exchange 2000 final solution. We discussed in the exploratory scenario presented in (see Table 7.3) that if we consider only the issues and risks involved with this particular mismatch, Domino should be preferred to Exchange. However, after examining other risk events caused by both candidates and more importantly after considering the strategic opportunities of continuing the Exchange platform, we concluded that Exchange 2000 is the best mail server option to satisfy the needs and constraints of the bank.

### 7.1.3 Critical Evaluation

We believe that this case study proved to be a valid evaluation effort. The guidelines and techniques present in TAOS offered appropriate support to conduct the evaluation and selection of mail servers. In particular, TAOS enabled us to effectively distinguish between mail server candidates, and hence ensure that decisions are made based on objective and clear arguments. In the case study both products obtained similar ranking regarding the way they satisfy the goals of the bank. Existing COTS selection methods such as PORE, CARE, and OTSO do not provide any guidance on how to proceed the evaluation process in situations like this one since they all assume the ultimate objective of the COTS selection is the ranking of COTS candidates, and then recommending the product with the highest ranking. TAOS method, on the other hand, was developed with the primary objective of supporting the evaluation team to identify and analyse mismatches caused by products as a way of complementing the ranking strategy and better informing the decision process. As a result, we argue that TAOS provides better support to address situations like the ones encountered in the mail server case study, in which products could not be differentiated based on their ranking, than other selection methods. In section 7.3 we further assess the novelty of TAOS method by performing a comprehensive comparison of TAOS against other COTS selection methods proposed in the literature.

#### 7.1.4 Lessons Learned

This section summarises the lessons learned from the mail server case study.

*Quality models can complement TAOS method by facilitating the understanding of the domain and refinement of goals.*

In this case study we were inspired by the mail server quality model proposed by [Carvallo et al., 2003] to facilitate the refinement of high level goals into more objective/detailed goals (i.e. operational goals). The use of quality models showed to be a useful strategy to better understand the mail server domain while helping the refinement of goals and defining appropriate metrics to goals. Note, however, that the measurement strategy described by TAOS extends the quality model by identifying appropriate acceptable interval and defining satisfaction functions to operational goals. Then, these results will be used to assess the extent to which products satisfy operational goals. We believe that quality model is a complimentary approach to TAOS method. While quality model focuses on the definition of relevant quality attributes for a particular domain and elaboration of appropriate metrics for their future evaluation, TAOS focuses on the elaboration of goals and assessment of their satisfaction in terms of COTS features. An important observation is that one of the criticisms against quality models is the cumbersome task to build the quality model for the first time. As a result, we believe that the benefit of using quality model in this case study was greatly because the mail server quality model was already available, however, the gains may not be the same when developing the quality model from scratch.

*Biased information concerning product features is a key problem of COTS evaluation.*

In highly profitable and competitive software markets like the electronic messaging market suppliers face aggressive competition, which means that they must employ heavy marketing strategies in order to increase their market share. Therefore, information we found about mail server products was generally biased and marketing-oriented. We mentioned in the beginning of the case study that the only sources of information we had were the products commercial and technical specifications together with third-party evaluation reports. Given the limitation of information, we were not able not perform test cases as recommended by TAOS to verify uncertain features and confirm supplier claims. On the one side, we believe this was a suboptimal strategy to effectively evaluate mail server products. On the other hand, this is the very situation frequently faced by small/medium enterprise (SME) when acquiring software systems from large, powerful suppliers [Brereton, 2004] SME customers may not be able to receive adequate pre-acquisition support from suppliers, and hence they have to cope with insufficient and biased information to evaluate COTS products.

*Exploratory scenarios help preventing typical limitations of MCDM techniques.*

A frequent criticism towards MCDM techniques [Kontio, 1995, Schoemaker and Waid, 1982] is the fact that the maximum aggregated score obtained in a decision problem may hide the individual importance to achieve a particular criterion. More specifically, some goals may have an importance that outweighs their overall contribution to the aggregated satisfaction score. Consequently, a particular product may obtain the highest overall score without satisfying some critical goals that cannot be compromised. It is fundamental to ensure that these critical goals are going to be satisfied by the winner product. By exploring the mismatches between goals and candidate products, we were able to identify goals such as *g<sub>4.1</sub> integrated document management*, which is considered to be a critical goal by stakeholders and is not satisfied by Exchange 2000. Since both products obtained similar rankings, this mismatch would be overlooked if we were solely using traditional MCDM techniques. Nevertheless, by using exploratory scenarios it was possible to perform an in-depth assessment of the impact caused by this mismatch and explore potential resolutions to handle it.

## 7.2 Electronic Document and Records Management System Case Study

### 7.2.1 Background

This case study followed the procurement of Electronic Document and Records Management (EDRM) system performed by University College London (UCL). The main objective of this case study is to explore the suitability of TAOS to support the selection of large and complex COTS products. We were able to observe and follow (in confidence) the whole procurement process. Although, some details about this case study could not be made available because of confidentiality issues, we have obtained valuable insights of TAOS effectiveness in improving the quality of decisions made during the selection of EDRM system.

To understand the technical details of the EDRM case study, first it is necessary to give a brief explanation about the domain in question. Document management system is a class of software systems concerned with with the creation, management, distribution, publishing and discovery of corporate information [Robertson, 2003]. It covers the entire lifecycle of documents in an organization, from their creation or capture to archiving. In the following we give a brief explanation about the objectives of the procurement process that has been published in the Official Journal of the European Communities:

“University College London (UCL) wishes to enter into a contract arrangement with a records and document management systems provider who can: form a medium to long term relationship with UCL to enable UCL to achieve its objectives in relation to records

and document management; be proactive in seeking out solutions, and in suggesting new technologies, or new uses of existing technologies; provide system installation and/or hosting services; provide data migration services; provide system integration support and consultancy; provide ongoing technical and user support. UCL requires a comprehensive records and document management software package. The solution will: provide modern graphical user interfaces and is expected to have a web interface for 'self-service' style use; be standards based and have long term support arrangements; provide an archiving facility for off-line storage, be capable of interfacing with UCL's database systems and authentication services (NIS, Active Directory or Oracle Directory Services), initially with its Oracle based student system; be capable of migrating data (images and indexes) from UCL's legacy Document Management System; be capable of accepting batches of scanned images and index information from separate bureau services; be ideally capable of being integrated with UCL Web Content Management solutions and scalable to provide workflow capability, forms based data capture, legal compliance, collaboration, enable re-use and eliminate duplication."

The short term objective of UCL is to acquire a solution to substitute the current systems of the registry and research grants departments. In the long term, UCL is seeking to extend the chosen EDRM system to other departments and deploy a standard, organization-wide document and records management solution. The registry and research grants sections will act as pilot areas for the strategic deployment of the EDRM across the university. As a result, the selected product needs to satisfy the immediate requirements of the pilot areas while allowing future extension of the system for an organization wide EDRM solution. A steering committee was formed in order to conduct the procurement process. The committee was composed by eleven people from different departments at UCL. They were the key decision-makers involved in the project.

At present, the registry division uses a legacy document management system with the main purpose of archiving documents. The admissions process is done manually and the produced documents are kept as paper documents. Only documents of accepted students are scanned and sent to student records. Even though the current legacy system is stable, it provides limited functionality and there is a risk of the system not being able to supporting the increasing workload. The registry seeks to replace the current system with an off-the-shelf solution that needs to be fully operational during the next admissions period. The EDRM system will need to interface with a recently purchased student management system called SITS. The registry plans to put both systems into operation simultaneously. The system which currently manages student records is called MIRUCL, the plan is to progressively substitute MIRUCL for SITS, then go fully live with SITS in the next 18 months.

The research administration section is responsible for the pre-award research applications to different funding bodies as well as the post-award administration of all research projects in the University. Currently, the department uses Oracle financial system integrated with

the human resources database and payroll system to manage research projects. The research administration section wishes to move from the paper-based business process to an electronic document management system. Moreover, the section is particularly interested to exploit collaborative facilities to share information among the different partners of research projects. In addition, the chosen EDRM system needs to comply with a number of legislative requirements, such as government clinical governance, data protection and freedom of information acts.

From the brief explanation of the project, we can make some initial observations of the challenges faced during the procurement process. The project involves the acquisition of requirements from stakeholders working in various departments of the university, facing different problems and therefore having different requirements in mind. These potentially conflicting requirements need to be well understood and agreed before starting the final assessment of candidate EDRM systems. The evaluation effort will be based on the requirements of the registry and research administration departments, which are the chosen areas to conduct the pilot project. Given the strategic objective to implement an organization-wide EDRM solution, it is difficult to assert that a product which suits the immediate requirements for the pilot project will in turn be adequate for other departments in UCL. This is a highly uncertain issue. Another challenge faced by the project is the variety of systems that the EDRM system has to interface, ranging from bespoke, off-the-shelf and open source systems. For all these issues, the EDRM case study offers a rich selection problem to evaluate the TAOS method. In the next section we will discuss the methodology we followed to perform this case study.

### 7.2.2 Methodology and Case Study Artifacts

A possible technique for evaluating methods like TAOS is to employ shadowing techniques. By shadowing, we aim at observing how the steps of the method could be performed by participants, executed in live. The aim is to understand the strengths and weaknesses of the method upon live execution in the presence of participants (e.g. requirements engineers, evaluation team, stakeholders, etc). Shadowing will provide feedback on the length of the evaluation, cost effectiveness of the method, inputs required, outputs expected, difficulties in executing the steps of the method, and their feasibility. Though, it is a recommended practice, we acknowledge that it is beyond the capability of a PhD project to conduct this kind of experiment. The reason for that is because shadowing is considerably time demanding. It requires high level of commitment and in depth knowledge from participants. Besides that, UCL has not allowed us to publish confidential data of the EDRM (Electronic Document and Records Management) project. For all these reasons, we have simulated the steps of the method using the data provided by UCL. we have met regularly with participants of the evaluation team to obtain input data about the project to perform the phases of TAOS as well as to report the results we obtained from applying

TAOS to the EDRM selection project.

We have conducted the EDRM case study for twelve months. We started following the procurement process in the early months of the project. This gave us the opportunity to examine the whole process of requirements acquisition and refinement. After the first 4 months, the project suffered severe delay caused by insufficient understanding of registry requirements. As a result, the committee agreed that the initially produced requirements specification should be reviewed and improved before continuing the evaluation process. In order to support this process, it was decided that a consultant, specialist in the document management domain, should be appointed to facilitate the process. This decision is justified by the complexity and costs involved in procuring and implementing an integrated EDRM solution in a large organization like UCL. Consequently, it was vital that the right package be selected.

The information used to conduct the case study came from a variety of sources, such as: regular meetings with key stakeholders, participation in committee meetings, invitation to tender document and other documents that were produced by stakeholders through the evaluation process. The meetings with individual stakeholders were primarily requirements elicitation sessions in which we played the role of requirements analyst. During committee meetings, we simply acted as observer. This proved to be a valuable source of information to observe the dynamics of the group and therefore understand how decisions were negotiated and made. In addition to information concerning the requirements for the new EDRM system, we were given an initial list of 30 potential suppliers. The list was obtained as the result of the tender notice published in the Official Journal of the European Communities. It is important to mention that the initial list included all market leading suppliers. We conducted the EDRM case study by following the phases of TAOS method systematically, as described in Chapter 6 and exemplified in the previous case study. In addition, we assessed the results of the case study against the evaluation criteria presented earlier.

### **1. Specify Goals**

We have followed the goal specification phase described in the TAOS method to specify the requirements for the EDRM system. The first step of the goal specification phase is the elicitation of high level goals. We have obtained relevant information by analysing the description of the project objectives published in the Official Journal of the European Communities. Table 7.4 shows some high level goals.

These high level goals were further refined into more specific and quantifiable operational goals. Once lower level goals were sufficiently defined, the evaluation team produced the pre-qualification questionnaire consisting of 39 atomic goals. Then the questionnaire was sent to all interested suppliers, a total of 30. At this stage, suppliers were not aware of



**High Level Goals for EDRM System**

1. Form a medium to long term relationship with UCL
2. Provide modern graphical web interface for 'self-service' style use
3. Provide an archiving facility for off-line storage
4. Be capable of interfacing with UCL's database systems and authentication services (NIS, Active Directory or Oracle Directory Services), initially with its Oracle based student system
5. Be capable of migrating data (images and indexes) from UCL's legacy document management system
6. Be capable of accepting batches of scanned images and index information from separate bureau services
7. Be ideally capable of being integrated with UCL web content management solutions
8. Be scalable to provide workflow capability
9. Allow forms based data capture
10. Meet legal compliance
11. Provide collaboration facilities
12. Enable reuse
13. Eliminate duplication

Table 7.4: High Level Goals that the EDRM product to be acquired should satisfy

the importance each requirement would play in the evaluation process, so that they did not know the exact criteria under which they were being evaluated. This demonstrated to be an interesting strategy to avoid biased claims from suppliers who would try to argue that their packages satisfy all mandatory goals in order to be shortlisted. Two members of the evaluation team with previous experience in procurement projects analysed the questionnaire responses. They have performed the evaluation separately so that one would not influence the opinion of the other. Then they met to discuss findings and decide which suppliers to eliminate, this stage was quite straightforward since they came up with similar comments. Some rejection criteria were very objective, for example, the product must be compliant with The National Archives requirements for Electronic Records Management, and the ability to support open standard image files. While other rejection criteria were based on rather subjective aspects such as the prospect of having a fruitful and long term relationship with supplier based on the quality of the answers given. In fact, some leading suppliers were eliminated for not satisfying this "fuzzy" but important criterion. To assess the satisfaction of this criterion the evaluation team had to use their knowledge in the domain. The pre-qualification process resulted in 9 suppliers being shortlisted.

The rationale for describing the early results of product assessment together with the goal specification phase is due to the fact that these processes were performed in parallel during the project. An important observation is that the pre-qualification questionnaire included generic but discriminative goals. Specifying every single goal in detail was clearly

not justified at this early stage of the evaluation process. One needs to make tradeoffs between the effort spent in evaluating alternative products against exhaustive criteria and the confidence in the rejection decisions being made. A direct consequence of making such tradeoffs is accepting the risk of rejecting promising candidates too early. With these implications in mind, the evaluation team opted for elaborating a comprehensive questionnaire. This decision was justified by the fact that UCL has strategic objectives for an organization-wide document management solution; hence it was convenient to ensure that potential suppliers fully satisfy critical goals before starting the time consuming negotiation process. It was agreed that the specific goals of the Registry and Research grants section, which are the pilot areas, should not be included in the questionnaire. Instead, more detailed analysis on the objectives and associated business processes of these pilot areas should be collected and elaborated in the next stages, after finishing the second pre-qualification process. From the discussion above, we can conclude that an iterative model of goal elaboration and COTS assessment like the one proposed in TAOS seems to be a sensible approach. Some decisions could be made early based on high-level goals, before specifying the goals in full detail; while other decisions required more detailed information about the domain and the available products.

The following step of the project consisted of elaborating more detailed business requirements for the strategic management of documents and records to be adopted in the future as well as refining and agreeing with all stakeholders in the registry and research areas their requirements for the EDRM system. We can observe from these steps taken that it was necessary to delimit the scope of the procurement and to clarify the goals for the pilot areas in order to continue the project. By doing so, it was possible to address potentially conflicting and ambiguous goals and reach early consensus among stakeholders before starting the negotiations with the preferred tenders. The invitation to tender (ITT) document was produced with more precise requirements and hence, helped the evaluation process in two ways. Firstly, by knowing exactly what are the requirements of UCL, suppliers were able to provide more accurate costing estimates. Secondly, negotiations with potential suppliers were conducted faster and could be supported by consensual arguments. Again, the guidelines provided by TAOS demonstrated to be an appropriate strategy to underline the importance and provide guidance on managing conflicting goals early in the procurement process.

Given the large number of goal statements described in the ITT document, we found tiresome the definition of acceptable interval for all operational goals. A number of operational goals had boolean metrics associated with them, while others required further examination to assign appropriate metrics and define acceptable interval. In these cases, we decided to give preference to examine in better detail critical and discriminative operational goals.

It is important to mention that some decisions concerning goal refinement and project scope had to be made in a later stage when sufficient information about the COTS candidates was available. For example, the high level goal *be ideally capable of being integrated*

with *UCL web content management solutions* is particularly important for the research grants area. They have recently obtained an open source tool to manage their web content. This solution encompasses a preliminary effort towards a collaborative infrastructure desired by the research area to facilitate interaction among research project partners. Some document management tools offer collaborative facilities as part of their software suite and such functionality may overlap with the current web content management solution. Consequently, it was necessary to clearly delimit the interfaces between both systems and define how they will interact. Besides that, another possible solution was to purchase a full suite solution from a single supplier and gradually discontinue the current web content management tool. The final decision of which design alternative to choose may be made after assessing all different possibilities offered by suppliers as well as analysing risks associated with each alternative.

## 2. Assess Products

As we have discussed previously, the assessment of products was conducted in parallel with the goal specification phase. This phase started with examining the websites of the initial list of 30 potential suppliers. The pre-qualification questionnaire served to shortlist 9 suppliers that seem to be the most promising ones, then the ITT document was sent to these shortlisted suppliers. This phase lasted several months mainly because stakeholders had problems in defining and agreeing with some requirements stated in the ITT document. At the end, the evaluation team selected 2 tenders to show their products in demonstration sessions.

## 3. Perform Matching

The assessment of the 2 shortlisted document management products consisted of comparing the responses of each supplier tender against the set of goals described in the ITT document. At first instance, we had to agree on how the satisfaction scores would be assigned. It was decided that a sub-committee would be appointed to draw up an evaluation table and conduct the scoring effort. The weighted summation scoring proposed by TAOS proved to be an effective and easy way to aggregate the scores. We assigned satisfaction scores to each supplier based on how well their responses to the ITT satisfy each atomic operational goal. This gave us an overview of product's compliance to requirements and helped us to eliminate the lowest scoring suppliers. More specifically, for each product being evaluated, we have attached one of the matching patterns *fulfill*, *differ*, *fail*, *extend* between the product and each operational goal. By detecting and classifying mismatched goals, we were able to compare candidate suppliers in terms of how much their products diverge from the desired level of goal satisfaction. This approach provided a suitable complement for the scoring strategy, which main objective is the measurement of product's

satisfaction degree.

The scoring process was particularly helpful to quantitatively compare suppliers and to objectively justify decisions made along the process. At the end of the scoring process, the 2 candidate suppliers were invited to demonstrate their products. The main objectives of the demonstration sessions were to verify how each product satisfies critical goals, clarify uncertain features and view the products interface and main functionalities. Shortlisted suppliers were also asked to provide reference customers to be visited by the committee.

#### 4. Analyse Mismatches and Manage Risks

To assess the effectiveness of the risk management strategy, we decided to choose two potentially severe risks faced by the EDRM project and follow the steps presented in the method to manage such risks. After discussion with stakeholders, it was agreed that two key risks are the possibility that the selected product may not suit the requirements of other departments at UCL (risk 1); as well as the danger that UCL may not successfully align its organizational processes with the selected system (risk 2). According to TAOS, the first step to analyse the detected risks consists of examining the risk exposure estimates. To apply the risk exposure technique, we needed to determine the probability of occurrence and severity of each risk. Let us start examining the probability of risk 1 to occur. As we have discussed before, the evaluation effort has been conducted based on the requirements of registry and research areas. At first instance, the ITT document described the requirements of these areas in detail but it covered relatively little the specific requirements the other academic departments have for a document management solution. Nevertheless some effort was spent to define strategic records management requirements in the ITT, it was not possible to guarantee that such strategies will comply with the particular needs and objectives of other departments at UCL. Consequently, it is probable that the selected product may not suit the requirements of other departments as the university has a strongly decentralised and diverse system of records management. Given that it was rather difficult to obtain an accurate estimation of the probability of risk 1 to occur, we decided that it would be sufficient to estimate that the probability of occurrence is medium (i.e.  $P(R_i)$  ranging between 4 and 6).

The fully implementation of the new EDRM system across the organization is clearly a critical project success factor. Failing to successfully adopt the selected system as a standard institution wide EDRM approach would mean to a large extent failing to meet the strategic objectives of the project. Consequently, the loss caused by risk 1 is very high (which means  $UL(R_i)$  ranging between 8 and 10). In particular, there is a significant economic loss if the strategic objectives of the project are not accomplished. This is due to the fact that the large expenditure made in evaluating and acquiring an EDRM system would benefit only the registry and Research areas, and it would not justify the substantial

investment allocated for the whole project. Based on the observations made above, we can conclude that the exposure of risk 1 is high. As a result, it was necessary to find effective actions to avoid or minimise the chances of the selected product not being adopted by other departments.

Concerning the exposure of risk 2, the probability of insufficient business process alignment to occur would be considerably low if we take into account the fact that the committee was very aware of the importance to undertake extensive organizational reengineering. While the fact of having committed sponsorship to lead the project is a necessary managerial decision, it is not a sufficient condition to ensure the success of the project. The implementation of the EDRM system involves a disruptive business process change where users will be highly affected. In particular, users will have to learn new procedures to create, store and access the documents they generate in their daily work. Possibly the new system will allow them to perform such tasks more efficiently and faster but certainly in a different way. As a result, users may resist or even reject the changes brought by the new system. Considering the overall social effects caused by the EDRM system, it is reasonable to say that the probability of UCL not aligning its business process with the new EDRM system is highly influenced by the probability of users reject the new system. Therefore, considering the combined risks we can assume that the joint probability of both risks to occur is medium rather than low. Similar to risk 1, the loss caused if risk 2 occur is very high since failing to conduct an effective business process reengineering may result in the failure of the system as a whole.

From the discussion above and according to the risk exposure zones presented in Figure 6.25, we can observe that both situations represent high risks, and therefore need to be controlled by a set of appropriate risk management actions. By examining the strategies presented in TAOS, actions that reduce the probability of risk events to occur seem to be appropriate ways to handle both risks. In particular, we have to deal with the high uncertainty involved in both risk events. According to the taxonomy of risk management 6.27, a natural way to reduce uncertainty concerning risks is to acquire information to better understand the situation, and therefore improve the quality of decisions. Therefore, as a preventive resolution to avoid those risks, the committee has decided to appoint an external consulting organization to conduct a detailed study of the strategic requirements for the EDRM system and to explore the feasibility of a standard document management system for the whole organization. One important outcome of this consultancy was the elaboration of an electronic records management policy based on best practice provided by The National Archives. This new policy will serve as basis to conduct the business reengineering process across the university. Once new policies and procedures are widely accepted and adopted, then the system could be effectively deployed. To complement the benefits of producing a standard policy to successfully handle risk 1, we consider the need of a strong partnership with the chosen supplier (this resolution refers to a type of risk mitigation action). The supplier needs to be committed with delivering a solution that

fits the needs of other departments within UCL. This is likely to result in adaption and customisation efforts, which will involve extra expenses. Hence, it is necessary to plan a comprehensive financial case supporting the acquisition and implementation of the system. In terms of actions to handle risk 2, the option *communicate and share risk* presented in the risk management taxonomy was an appropriate strategy. To ensure that organizational changes are performed, it was necessary to engage in an exercise of staff motivation and involvement with the development of the system. The project should also be accompanied by the designation of working groups with clear responsibilities and in-depth training of staff with the system.

## 5. Select COTS Solution

The final phase of the evaluation process was quite straightforward. From the results of the demonstration sessions it was clear which was the preferred supplier. The overall worthiness of the chosen product is much higher than the other alternative. Even though the chosen product is substantially more expensive, the evaluation team considered that the value offered by the product may pay off. Finally, the evaluation team negotiated with the supplier costing and legal issues.

### 7.2.3 Critical Evaluation

In this section, we present a critical evaluation of TAOS by assessing the method against the evaluation criteria defined in the beginning of this chapter. As discussed in Section 7.2.2 we have performed this case study by simulating the phases of TAOS using real data from the EDRM project. The outcomes produced by TAOS have been discussed with members of the evaluation team. During these discussion sessions we have gathered their reaction to TAOS recommendations and proposed alternatives. The following evaluation reflects this feedback. Due to the strategy we used to conduct the EDRM case study, we were not able to measure the value added by applying TAOS. The reason for that is because there was no control situation against which to test the results obtained by TAOS. We acknowledge this is a serious limitation of the evaluation effort.

Does TAOS provide complete, understandable and useful method to guide the evaluation of COTS products?

The first aspect we discuss is to what extent TAOS facilitates the evaluation of COTS products. The ease to use is a key aspect that has motivated the development of TAOS method. Several aspects of TAOS contribute to making the method we propose easy to use. For instance, the wide use of guidelines and templates to explain each phase of the method, facilitates the application of the method by users without previous experience

in COTS selection. The method is based on intuitive concepts organized in a structured manner, which contributes to its repeatability. Furthermore, users of the method do not need to understand the details of the decision theory behind TAOS. On the other hand, users must have a comprehensive understanding of organization objectives and knowledge about the domain.

Given that we have personally conducted the case studies, it is difficult to have a conclusive validation of the overall ease of use of TAOS. In particular, the guidance provided by the exploratory scenarios and risk management taxonomy seem to be beneficial and ease to apply in the context of the EDRM project. The use of goal-oriented requirements specification was considered to be useful mainly because it allowed the clear separation between strategic objectives and low level requirements, at the same time that it explicitly exhibited the relationship between them in the form of goal tree decomposition. As criticism, we consider that the prioritisation technique (presented in Section 6.2.5) was quite demanding to apply given the number of operational goals produced. The evaluation team preferred to use a qualitative prioritization scale ranging from low, medium and high priority. As a result, it is necessary further evaluation to test the suitability of the prioritisation technique in large COTS selection projects.

Concerning the completeness of TAOS, we have observed that the method covers all major steps necessary to conduct a disciplined selection process. TAOS phases provide a clear, structured process to guide the acquisition and analysis of relevant information. This information assists the evaluation team in making informed decisions, and therefore, choosing a product that suits the needs of the organization. The added value that the method we propose has given to the decision process is that it provides guidance on how to manage mismatches between organization requirements and features offered by candidate products. In addition, heuristics have been provided to support the tradeoff of mismatched requirements. One of the limitations of our approach is that it does not provide specific mechanisms to fully address the problem of architectural matching. We simply tackle architectural issues during the mismatch and risk management phase through the use of exploratory scenarios. We observed in the EDRM case study that a core requirement the system to be acquired needs to satisfy is the ability to successfully integrate with several other systems already in place at UCL. Therefore, architectural constraints were an important criteria to shape the decision process. However, they were not properly supported by TAOS.

**Is goal-based requirements engineering a suitable approach to specify requirements for COTS-based systems?**

Our interest in goal-based specifications is centred on the fact that they are suitable for eliciting and elaborating requirements in such a way that stakeholders are less likely to over-specify the potential alternatives by prematurely committing to specific implementa-

tion solutions. The notion of goal is particularly appropriate to understand to what extent satisfying one goal may hurt the satisfaction of another, and therefore facilitate the identification of possible tradeoffs to be made among conflicting goals. As we have discussed in Chapter 4, the requirements activity changes considerably in COTS-based systems compared to traditional development. Therefore, we need to investigate the suitability of goals to specify requirements for COTS-based systems.

We have started the case study by specifying high level goals that were further refined to compose the pre-qualification questionnaire. By defining a goal refinement tree, we were able to identify interactions between non-related goals. It also proved to be a natural way to acquire more information about ambiguous and abstract goals, so that they could be properly measured in a later stage of the evaluation process. Based on our observations collected during the EDRM case study, we have found that goal-based specifications provide an incremental and iterative strategy to elaborate goal refinements and reason about the impact of decisions. This strategy is particularly appropriate to deal with partial and incomplete information about the system domain, in which more information may be gradually acquired during the evaluation process; hence allowing goals to be adjusted and refined. For all these reasons, we can conclude that goal modelling is an appropriate approach to elicit and specify requirements for COTS-based systems.

Does the matching process provide an effective way to evaluate to which extent a COTS product satisfies the requirements of the acquirer organization?

TAOS suggests that more importantly than simply computing numerical satisfaction scores as a way to evaluate candidate products, the evaluation team should be able to examine the effects and risks caused by unsatisfied goals. Therefore, in order to verify the validity of this proposition, we have investigated the usefulness of the matching patterns to detect and analyse mismatches in the EDRM case study. We have applied the matching patterns to classify operational goals in terms of how well they were satisfied by EDRM products. Matching patterns provided an objective way to distinguish mismatched goals that needed to be further examined. Once mismatched goals were detected, we focused the evaluation effort on reassessing the importance given to them and elaborating test cases to investigate mismatches in more detail. However, we observed that in some situations the patterns were of limited scope. An important lesson learned is that the evaluation team should not rely solely on the pattern of a goal to guide the subsequent evaluation effort, experience and domain knowledge are always crucial for successfully conducting the evaluation process.

We observed that the activities of evaluating suppliers and analysing the matching between product features and goals were clearly intertwined, in such a way that it was not possible to distinguish a clear separation between both activities. The assignment of satisfaction scores was obtained primarily based on supplier responses to the ITT and, for shortlisted suppliers, their scores were revised after the demonstration sessions. This empirical ev-



idence reinforces our assumption that the selection process is highly interactive and the separation of phases is commonly blurred.

**Is the risk management strategy effective to identify, analyse and resolve risks that may arise during COTS-based development?**

Given the scope and strategic objectives of the EDRM procurement process, this project was prone to a number of risks. On the one hand, implementing an integrated enterprise-wide document management solution allows users to leverage originally fragmented and meaningless information into useful and accessible knowledge to improve organizational efficiencies and enhance productivity. On the other hand, integrating a standard EDRM solution across departments within the organization represents a significant endeavor. In particular, it is necessary to develop new electronic records management procedures and policies and ensure that the system to be selected complies with them. In addition, the new EDRM system needs to be integrated with a number of other systems running under heterogeneous networks and databases. Moreover, an enterprise-wide EDRM system needs to satisfy very specific requirements from different departments at UCL, which may not be necessarily the same requirements of the pilot areas that guided the evaluation effort. For all these reasons, the EDRM case study presented a rich opportunity to verify the effectiveness of the risk management strategy provided by TAOS.

The exercise of applying TAOS risk management strategy to the EDRM case study proved to be valid. Much of the risk management actions draw on established software engineering good practice and they were suitable to provide generic guidance on how to solve the classes of risks being examined here. However, we cannot guarantee that the actions will be appropriate for risks from other selection contexts. In addition, the correct interpretation and use of the proposed actions will largely depend on the skills of the evaluation team. These are one of the limitations of our approach.

**Do exploratory scenarios provide useful guidance to make informed selection decisions?**

Improving the quality of decisions made during the selection of COTS products is one of the objectives of this research. In this context, the objective of exploratory scenarios is to organise relevant information gathered through the evaluation process, expose conflicts and analyse them. Ultimately, by using the proposed scenarios, the overall worthiness of each COTS alternative can be measured in order to inform the decision making process. The rationale for using scenarios is because they are a powerful and intuitive mechanism to capture and structure different types of information. TAOS presents three templates to guide the creation of exploratory scenarios, each template addresses particular aspects of the decision process: the first template is the most generic one and explores the overall worth of COTS solutions, the second template guides the exploration of mismatch situations, finally the last template examines conflicting goals.

We have applied the exploratory scenario templates to the EDRM project. In particular, we built some scenarios to explore the worthiness of each one of the 2 products shortlisted to participate on the demonstration sessions. The scenarios gave us a broad perspective of the benefits and limitations of each alternative and served as a helpful comparison framework. In particular, it was possible to analyse the shortcomings of each alternative (i.e. its mismatches and involved risks) and explore possible ways to overcome them. In addition, the scenarios allowed us to establish the necessary issues to be included and negotiated in the pos-procurement plan to ensure that the preferred supplier delivers the agreed solution. Concerning the management of risks, an important observation is that when making a decision to handle a particular risk, it is important not only to choose an appropriate resolution but also to examine the cost and value involved in each resolution. This means that the model presented in Section 6.6 to measure the worthiness of a particular COTS solution may be extended to measure the worthiness of a risk resolution action. For example, the decision to hire an external consultant has helped the elaboration of strategic electronic records management requirements and reduced the risks of unsuccessful implementation of the EDRM system. On the other hand, the appointment of the consultant incurred in substantial extra costs to conduct the requirements analysis phase. This means that the variables risk, cost and value had to be carefully balanced before deciding which risk resolution to choose. We did not have the chance to thoroughly evaluate the template scenario 3. Therefore, further evaluation effort is needed to assess the usefulness of template 3.

From the results of the case study, we observed that scenarios were a suitable strategy to reason about the impact of decisions. They allowed in-depth examination of critical issues involved in the decision process as well as helped structuring the pos-procurement plan. By creating and analysing exploratory scenarios, it was possible to discover underlying issues that did not appear in the initial phases of the evaluation activity, hence increasing the confidence in the final decision. However, a significant limitation found was that the scenario templates were too generic and needed to be significantly customised to address the particular problems faced by the EDRM project. Consequently, we concluded that the usefulness of the exploratory scenarios highly depends on the evaluation team's ability to accurately model the relevant issues to inform the decision making process. Another limitation of the exploratory scenarios is the limited guidance to analyse architectural issues. Modelling the system architecture is particularly important when selecting multiple components. We plan to address this limitation in future work.

Do the heuristics help guiding mismatch and conflict analysis facilitate the tradeoff process?

We have developed a set of heuristics to complement the use of exploratory scenarios. The heuristics were elaborated based on our experience conducting the AFTN case study (see Chapter 5) and on software engineering best practices. They focus on two fundamental aspects of the decision process: the analysis of mismatches and conflicting goals. The

added value that the heuristics we propose has given to the elaboration of exploratory scenarios is the explicit reasoning of how tradeoffs can be performed to deal with limitations of COTS alternatives. The heuristics to analyse mismatches and conflicts may be used as guidance to generate a number of possible tradeoff resolutions, then an adequate resolution can be identified among those generated when the risk and impact of the conflict becomes acceptable with respect to the cost for resolving it. The more information the evaluation team obtains regarding the causes of the mismatches and conflicting goals, the better is the generation of adequate resolutions.

Experience from applying the heuristics to the EDRM case study has suggested that they should be used as generic guidelines on how possible tradeoffs can be made. The presented heuristics should not be viewed as definitive solutions, instead they should be used to inspire and guide the negotiation process. Hence, they should not be considered “silver bullets”. Much creative thinking and domain knowledge is required to understand the impact of mismatches and conflicts; and choose appropriate tradeoff resolutions. Put in another way, the heuristics provide a way to trigger such creative thinking.

#### 7.2.4 Lessons Learned

This section summarises the lessons learned from the EDRM case study.

*The selection of COTS products largely depends on the expertise of the evaluation team*

From our experience in conducting the EDRM case study, we observed a number of benefits of having a disciplined method to guide the evaluation and selection of COTS product. By following the phases presented in TAOS it was possible to provide a systematic and repeatable process of gathering relevant information, analysing alternatives and making decisions. However, we also recognize that no matter how hard we try to be objective, there are still underlying values and subjectiveness in the ways that people perceive, judge and ultimately make decisions. Consequently, we consider that to a large extent the success of the selection process depends on the experience and domain knowledge of the evaluation team.

*A method to guide the selection of COTS needs to be customisable*

TAOS method provided useful guidance to conduct the EDRM project. Some parts of the method offered valuable support, such as, in acquiring customer requirements, evaluating product alternatives, and analysing and managing risks. On the other hand, approaches such as the goal prioritization technique was not very beneficial to the project, as their use was considered to be quite complex and time demanding. Even not using these approaches, the evaluation exercise proved to be valid. In addition to that, it is obvious that there is no one fits all solution to developing software systems. Based on these concerns, we consider

that TAOS should provide alternative paths to allow tailoring the method to better suit the needs of different procurement circumstances. We plan to address this issue in future work.

*Good understanding of organizational requirements is vital for the success of selection process*

Given the committee's decision to appoint an external consultant with experience in the domain to help defining the strategic requirements for the EDRM system as well as clarifying the requirements of stakeholders from the pilot areas, we could observe the importance of having fully understood and agreed requirements during the evaluation of COTS products. The procurement process had to be slightly delayed to conclude that task. In fact, the elaboration of requirements to issue the ITT document was the most time consuming task. This was due to the fact that the new system will affect several areas of the organization, and consequently it was difficult to reach a consensus about the scope of the system. An important comment is that the fact of having a comprehensive knowledge about UCL requirements for the new EDRM system does not contradict our claim that requirements for COTS-based systems should be flexible. Instead, by ensuring that stakeholders have a good understanding of their own requirements, the negotiations with the chosen supplier can speed up since stakeholders are able to clearly distinguish which are the core functionalities they want to be satisfied by the new system from the negotiable ones.

### 7.3 Comparative Study

The objective of this section is to compare the novelty and effectiveness of TAOS method against existing selection methods. Several approaches have been developed to support the evaluation and selection of COTS products. However, the majority of the approaches are fragmented as they aim to address specific issues of COTS evaluation rather than supporting the whole process. Given that TAOS intends to cover the entire evaluation process, we considered that it would make sense to compare our proposal against other approaches with similar scope and purpose. We chose four well-established methods to perform the comparison analysis: PORE, OTSO, PECA and CARE. This decision was driven by the fact that these approaches are accepted both by practitioners and academics, and therefore they provide a good comparison baseline. We gathered information about each method from a variety of papers and reports published in the academic literature. The sources of information were the following references: PORE [Ncube and Maiden, 1999, Ncube, 2000, Ncube and Maiden, 2000], OTSO [Kontio, 1996, Kontio, 1995], PECA [Santiago et al., 2002], CARE [Chung and Cooper, 2003]. These methods have been thoroughly described in chapter 3. We now compare each method against the set of requirements we have established in chapter 5 in which a COTS selection method

would need to fulfil. It is important to note that the set of requirements is not intended to be exhaustive, instead, it aims to focus on the challenges faced by the requirements process for COTS-based systems.

### 7.3.1 Comparison with other COTS selection methods

The following tables present a comparison of how well COTS selection methods address the set of requirements defined in Table 5.1. From the results of the comparison exercise, we observed that all studied methods provide a systematic process to guide the evaluation of COTS products. Most methods apart from CARE present specific guidelines to facilitate the use of the method. CARE provides limited guidance on how to apply the techniques and processes proposed which may affect the usability of the approach. All methods emphasise the importance to be customisable in order to suit the needs of different organizations and selection contexts. There is a consensus among some methods regarding the importance of the requirement process towards the successful selection of COTS. In particular, TAOS, PORE and CARE place the requirements process as a core activity of the evaluation effort. While PECA and OTSO pay less attention on the requirements activity, in both methods the main purpose of acquiring stakeholder requirements is to construct the evaluation criteria under which the COTS candidates will be evaluated against. Consequently, these methods disregard an important aspect of the development of any software system that is the effective understanding and modelling of stakeholder requirements. In PECA and OTSO, the elaboration of requirements is supported by the GQM method [Basili et al., 1994]. GQM allows the refinement of goals into quantifiable evaluation criteria. However, both PECA and OTSO assume that goals already exist, and therefore, they do not support the elicitation phase. TAOS, PORE and CARE use goal-oriented approaches to elicit and model requirements. This fact reinforces our claim that goal-based requirements engineering is suitable to specify requirements for COTS-based systems.

Concerning the support for the negotiation of requirements, none method except TAOS fully support this activity. PORE and CARE provide a preliminary attempt in that direction by supporting the identification of conflicting requirements. However, they do not provide any strategy to handle identified conflicts. In addition, both methods do not address the negotiation of requirements that cannot be satisfied by any COTS alternative (i.e. negotiation of mismatched requirements). It is an interesting fact that none method effectively deals with the negotiation of requirements, even though they all seem to agree on the necessity to modify and balance requirements against the features provided by COTS products. In TAOS, the negotiation of requirements is supported by a number of heuristics that helps the evaluation team to analyse possible tradeoffs to be made in order to manage conflicting requirements as well as unsatisfied requirements.

<b>Requirements for a COTS selection method</b>	<b>TAOS</b>	<b>PORE</b>	<b>PECA</b>	<b>OTSO</b>	<b>CARE</b>
Be systematic and easy to use	The method consists of five well defined and repeatable phases. In each phase a number of guidelines and templates are provided to facilitate the use of the method.	The method includes a set of well defined templates and guidelines. A rule-based tool provides advice for choosing and using appropriate techniques to guide the evaluation process.	The process consists of four iterative steps. It is highly flexible to fit different organisations and selection situations.	The method provides a systematic and generic process model that can be customized for each organization and selection case as necessary.	The method is described as a set of intertwined activities. Given the lack of clear guidelines, the use of some techniques proposed in the method may be difficult to use by non-experts.
Support the requirements process	A detailed goal-oriented requirements approach is described to guide the whole requirements process. A number of guidelines and techniques are presented to guide the identification, refinement, prioritization, and management of requirements. The method also allows the identification of interactions and conflicts among goals and presents a strategy based on utility theory to analyse tradeoffs among conflicting goals.	The requirements process is a core phase of PORE method. An iterative process model is described to elaborate and refine requirements at the same time that the number of COTS candidates is reduced. The method follows a goal-oriented approach to model requirements and presents a number of well-known techniques to assist the requirements process.	The requirements process is not well defined. Only few observations are provided concerning the importance to acquire functional and non-functional requirements as well as other issues related to architecture/interface constraints, operational environment, and programmatic constraints. Similarly to PORE, in PECA the requirements process is performed iteratively with the identification of products.	OTSO presents a simplified view of the requirements process in the sense that it assumes that requirements already exist to form what the method calls reuse goals and evaluation criteria.	The requirements process is considered to be a key phase of CARE. The method is based on well-established goal-oriented requirements engineering techniques.
Requirements elicitation and modelling	A number of guidelines are provided to support the elicitation and refinement of high level, abstract goals into lower level, quantifiable operational goals. The method provides a measurement strategy to facilitate the evaluation of goal satisfaction.	PORE provides templates for acquiring information concerning user requirements and product capabilities. Each template describes various techniques to support these tasks.	GQM method is used to support the refinement of requirements.	GQM method is used to support the decomposition and quantification of requirements.	i* and NFR frameworks are used to guide the elaboration of requirements. The former is an agent-driven approach, while the later is goal-oriented approach to model non-functional requirements.

Table 7.5: Comparison of selection methods - Part 1

<b>Requirements for a COTS selection method</b>	<b>TAOS</b>	<b>PORE</b>	<b>PECA</b>	<b>OTSO</b>	<b>CARE</b>
Requirements negotiation	The method provides a number of heuristics to guide the negotiation of conflicting requirements as well as the tradeoff analysis of unsatisfied requirements. Exploratory scenarios allow the evaluation team to explicitly examine the impact of choosing different tradeoffs.	Partially supported. In PORE, requirements are prioritised in order to facilitate the evaluation of COTS alternatives. The goal-oriented approach allows the identification of conflicting goals. However, the method does not explicitly support the negotiation of mismatched requirements.	Not supported. PECA simply mentions the need to change requirements in order to accept COTS limitations. But no strategy is described on how to negotiate unsatisfied requirements.	Not supported. OTSO includes in its model process the possibility to change requirements based on the results of the product evaluation effort. However, the method does not explicitly support the negotiation of requirements as a way to accommodate COTS limitations.	Partially supported. Conflicting goals are identified from the goal decomposition graph using mechanisms available in the NFR framework. However, management of conflicting goals is rather limited. In addition, the method does not support the negotiation of requirements with respect to COTS limitations.
Support decision making process	Utility theory is applied to measure how well COTS products satisfy requirements. TAOS provides a systematic approach to define acceptable interval of requirements and satisfaction functions, which will be the basis to quantitatively assess the satisfaction degree of requirements.	AHP, outranking and WSM techniques are used to guide the ranking of products and consequent elimination of non-compliant alternatives.	PECA provides limited support to the decision process. It mentions the possible use of weighted aggregation methods like but no further explanation on how to conduct this task is given.	AHP technique is employed for consolidating the evaluation effort. In simple decision situations, weighted scoring method is recommended to use. OTSO also describes two simple techniques to estimate the costs involved with each COTS alternative.	Not supported. The method does not use any decision-making technique to compare and rank COTS alternatives.

<b>Requirements for a COTS selection method</b>	<b>TAOS</b>	<b>PORE</b>	<b>PECA</b>	<b>OTSO</b>	<b>CARE</b>
Identify and handle mismatches between requirements and products	Analysing and managing mismatched requirements is one of the key objectives of TAOS. The method provides matching patterns to help the identification of mismatches and heuristics to explore possible resolutions.	Not supported. The method simply indicates the importance to change unsatisfied requirements in order to accommodate COTS features. But no explanation on how to conduct such changes is provided.	Partially supported. Three techniques are presented as possible approaches to perform this task: sensitivity analysis, gap analysis and analysis of the cost of repair. The limitation here is the lack of a well-defined process to guide the use of such techniques in order to identify and handle mismatches.	Not supported. The method assumes that the ultimate objective of the evaluation process is the selection of the highest-ranking product.	Not supported. The method simply describes the matching and the bridging the gap (i.e. mismatch handling) processes. However, CARE does not offer any systematic strategy in order to effectively support these tasks.
Analyse and manage risks involved with the selection of COTS	A detailed risk management approach is described by TAOS. It consists of identifying risks involved in mismatches and conflicting requirements, prioritising risks and choosing appropriate risk resolution actions. Risk management taxonomy is presented to help the selection of resolution actions.	Not supported. PORE does not state the necessity to handle risks.	Partially Supported. PECA argues that the level of rigour of the evaluation process depends on the severity of risks caused in case a wrong product is selected. However, it is not clear how risks are measured and how to choose appropriate risk resolution actions.	Not supported. OTSO does not support the risk management activity.	Not supported. CARE does not include any support to handle risks.



<b>Requirements for a COTS selection method</b>	<b>TAOS</b>	<b>PORE</b>	<b>PECA</b>	<b>OTSO</b>	<b>CARE</b>
Analyse impact of decisions	Exploratory scenarios offer a broad understanding of the worthiness of each candidate product. In particular, the impact analysis is driven by three variables: cost, value and risk.	Not supported. PORE just supports the evaluation and ranking of products in terms of their compliance with user requirements. The method does not allow further investigation of the impact caused by unsatisfied requirements or other risks.	Partially supported. PECA superficially supports the impact analysis. The techniques described as possible tactics to manage mismatches can also be employed to reason about the impact of decisions made through the selection process.	Partially supported. OTSO provides a cost-benefit analysis to compare the value of each COTS alternative. It also emphasises the importance of meetings with stakeholders to discuss strategic decision issues. However, the method does not provide a systematic approach to conduct the impact analysis.	Not supported. The method does not offer mechanisms neither to guide the decision process nor to reason about the impact of decisions.

Table 7.8: Comparison of selection methods - Part 4

Another similarity among the studied methods is that all approaches, excepting CARE which does not explicitly support the decision process, employ multi-criteria decision making techniques to compare and rank alternative products. The common techniques used are WSM and AHP. We consider the way PORE and OTSO use WSM is inappropriate to compare COTS alternatives because these methods suggest that the evaluation team should assign numerical scores ranging from 1 to 5 to measure both the weight of requirements as well the satisfaction score between requirements and COTS products. The main drawback of this approach for the evaluation of products is because numerical scores are not a meaningful way to compare alternatives. For instance, a product that obtained twice the score of another product with respect with satisfying a particular requirement cannot be interpreted as if it is twice better than the other product. Instead of using numerical scores, one should prefer to use normalised scores to measure the satisfaction degree of each product. Another problem with assigning numerical scores to prioritise requirements is because it does not suggest how the tradeoff of conflicting requirements can be performed. According to PORE and OTSO a better technique to support the decision-making process is AHP. This technique allows pair-wise comparison among products, which is considered to be a better strategy than the numerical scoring of WSM. AHP also ensures the consistency of ranking results by employing eigenvalue matrices. However, similarly to WSM, AHP provides poor support to reason about conflicting requirements. Another limitation of AHP is the way preferences are assigned using the 1-9 ratio scale. From our experience, it is quite difficult to compare how products satisfy different classes of requirements in terms of this comparison scale. We consider that the measurement strategy of TAOS provides better support to the decision making process than the multiple criteria decision making techniques employed by other selection methods. The measurement strategy is based on utility theory to allow the comparison of products in terms of the normalised degree in which they satisfy each requirement. The satisfaction degree is obtained by means of satisfaction functions that were generated from objective metrics associated to each requirement. We believe that the comparison of products with respect to the degree they satisfy requirements by using commonly agreed metrics is more effective than the comparison by means of ratio or numerical scales.

The main contribution of TAOS over the studied selection methods is the method's ability to properly address mismatches and risks that may arise in the selection of COTS-based systems. TAOS presents a number of matching patterns to formally classify the degree in which requirements are satisfied by COTS alternatives. These patterns will serve as the basis to systematically explore the potential risks caused by mismatches. In contrast, other methods assume that the ultimate objective of the evaluation process is the ranking of COTS alternatives obtained from the decision-making techniques described earlier. We consider that as important as comparing and ranking COTS against the set of requirements is investigating how mismatched requirements affect the overall worthiness of the final system. In this way, the studied methods fail to support the analysis and management of mismatched requirements. It is important to note that PECA mentions the possibility to

use sensitivity analysis, gap analysis and analysis of the cost of repair techniques as a way to address this problem. However, the applicability of such techniques to the selection of COTS is limited without a systematic process to guide their use. Finally, in terms of supporting the management of risks, TAOS is the only method presenting a detailed approach to identify, analyse and handle risks. The method has adapted and extended well established risk management techniques [Boehm, 1991, Charette, 1990, Kontio, 1997, Vitharana, 2003] to address risks particular of COTS-based systems.

### 7.3.2 Critical Evaluation

From the results of this comparison exercise, we could observe that TAOS provides better support to satisfy the defined requirements for a COTS selection method than other studied methods. We consider that such results establish the novelty of TAOS regarding other methods. TAOS provides guidance in aspects of the evaluation process where other methods are weak or even fail to address. Nevertheless, we are aware of the limitations of the chosen comparison procedure. A more effective means of comparison could be achieved by asking subjects to apply each method to conduct a particular selection problem, and then comparing the results obtained with each method. The main benefit of this comparison procedure is the opportunity to empirically assess the effectiveness of different selection methods in solving the same problem. As a result, the evaluation exercise would allow the comparison of the methods against the same evaluation baseline. Moreover, it might increase the confidence that the results of the evaluation are correct [Kitchenham et al., 1997]. On the other hand, substantial effort is needed to conduct this type of experiment, since it is necessary to plan the evaluation, design experiments, monitor subjects, analyse results. Moreover, it is arguable whether this type of carefully controlled experiment is appropriate to evaluate software engineering methods [Dawson et al., 2003]. Finally, given the time-scale of the doctoral programme, we believe that the results obtained from the content management and mail server case studies together with the comparison with other methods provide sufficient evidence that TAOS is a valid and capable method for guiding the selection of COTS products.

## 7.4 Summary

This chapter has presented three evaluation efforts to assess the effectiveness of TAOS method to support the selection of COTS products. We have performed two case studies - the selection of mail servers and electronic document and records management systems. We have also conducted a comparative study to examine the adequacy of TAOS method to the set of requirements defined in chapter 5 compared to other existing selection methods.

The objective of the mail server case study is to exemplify the use of TAOS. We have

demonstrated how to perform each phase of TAOS in details. While in the EDRM case study, we have followed a real COTS selection project. This case study allowed us to establish the suitability of TAOS to support the selection of large, complex products. Although there are recognised limitations in the TAOS method, the principal benefit of TAOS is the systematic support for identifying and managing mismatches between requirements and COTS products while other approaches provide very limited support to address these problems.

## Chapter 8

# Conclusions and Future Work

The goal of the work presented in this thesis has been the development of new processes, strategies and mechanisms to improve the quality of decisions made during the selection of COTS products. The starting point for this research has been the investigation of new challenges faced by the requirements engineering process for COTS-based development. We have also explored the risks customer organizations have to handle when selecting COTS products. These studies have motivated the main objective of this thesis: to effectively support the identification, analysis and management of mismatches that may occur between customer requirements and COTS products. This research has resulted in the TAOS method, a systematic and iterative method that provides effective support to guide organizations in selecting COTS products that best suit their needs and constraints. This final chapter reexamines the contributions of this thesis and provides an overview of future work.

### 8.1 Summary of Contributions

The contribution of this thesis to the evaluation and selection of COTS products is the TAOS method. More specifically, our method employs a goal-driven approach to support the elicitation and modelling of requirements. By following a goal-driven requirements modelling approach, the positive and negative interactions that may occur among goals are easily identified through the goal refinement process. These interactions among goals will influence the decisions made through the evaluation of products. Another important step of the requirements process is the prioritization of goals. TAOS supports the prioritization of goals with a technique to determine the relative weight of goals. To obtain the weights of conflicting goals, TAOS proposes another technique based on utility theory to facilitate the analysis of how possible tradeoff can be performed, and hence aid customers to reprioritise their goals. The method also presents a number of guidelines to refine goals.

The refinement process continues until the satisfaction of goals can be assessed by means of metrics. In order to objectively evaluate how candidate products satisfy customer goals, TAOS presents a measurement strategy that helps the evaluation team obtaining the interval of acceptable values to satisfy a particular goal and the satisfaction function of goals, these results will serve as the basis to determine the degree in which each product satisfies a particular goal.

TAOS provides a straightforward strategy to identify, understand and evaluate COTS products. The core of the evaluation activity is the analysis of the matching between products and goals. In order to support that step, the method presents a set of matching patterns to precisely categorise the satisfaction degree of goals. Products are then ranked in terms of the degree in which they satisfy goals. The ranking of products is obtained by using the weighted summation aggregation operator. By analysing the matching pattern of goals, the evaluation team is able to identify mismatches. A number of heuristics are provided to facilitate the analysis of mismatches, and hence identify risks originated from mismatches. TAOS then provides a detailed strategy to examine and manage risks. The evaluation team can choose appropriate actions to handle different classes of risk by using the taxonomy of risk management described by TAOS. The taxonomy contains actions to mitigate, reduce the damage and probability of risk events. Finally, TAOS guides the evaluation team to decide which is the best product for the acquirer organization by measuring the worth of each candidate, this consists of trading off the value against the costs and risks of each product. Exploratory scenarios enable the assessment of relevant issues that may influence the overall worth of products.

## 8.2 Future Work

The method presented in this thesis can be improved and extended in several directions, in this section we describe possible areas for future work.

### Organizational modelling

A possible extension to the goal-driven approach proposed by TAOS is to use the  $i^*$  technique. This technique has been originally developed to model the goals of heterogeneous actors in business and social-technical systems.  $i^*$  has been extensively applied in different application domains, from web applications and integration of organizational modelling with object-oriented modelling to air traffic management [Bolchini and Paolini, 2004, Castro et al., 2001, Maiden et al., 2004]. In particular, in the context of COTS-based systems, the technique has been applied to model architecture dependencies of multiple COTS selection [Franch and Maiden, 2003] and to support the goal modelling phase of CARE selection method [Chung and Cooper, 2003]. These

preliminary results suggest the potential suitability of  $i^*$  for modelling the goals and architecture of COTS-based systems. We intend to investigate how  $i^*$  can be used to facilitate the simultaneous tradeoff analysis among goals, architecture and COTS by guiding the evaluation team to reason about the impact of conflicting goals, support the modelling of dependencies among products and guide the decision to choose appropriate architectures to integrate COTS products.

### Requirements Evolution

In COTS-based systems, developers have to handle two major sources of change: the frequent modifications of the product and the evolution of the stakeholder requirements. In this thesis we focused on the evolution of requirements solely during the evaluation of products. During this phase, stakeholder requirements change constantly as they get better understanding about available product features as well as a result of the negotiation process. This means that the final requirements are inevitably a compromise among them. In the future, we aim to investigate how the evolution of requirements takes place through the lifecycle of COTS-based systems. Many research questions remain open in this direction. For example, developers have to deal with issues such as: What is the impact of the frequent COTS evolution cycles on the requirements stakeholders have agreed prior the product has been selected? Given that it is expected the business and technologic environment will change after the product has been integrated into the organization, how can these changes be reflected in the COTS systems?

### Multiple selection of COTS

In this thesis we have superficially covered the problem of multiple COTS selection. TAOS risk management strategy and exploratory scenarios provide basic support to handle the risks and mismatches that may arise when selecting several products. We assumed that the class of problems covered by TAOS involves the selection of a core product which will drive the decisions to select other complimentary products. However, there may be a number of situations where the selection of a multiple COTS-based system is far more complex. When the control of power is shared among products and critical dependencies and conflicts may occur among them. For instance, requirements defined to select a specific product may conflict with requirements for other product(s) that will be integrated together. This is an issue that needs to be better addressed in future extensions of TAOS.

### Architectural matching

Another relevant direction of research is the architectural matching problem. This problem is closely related to the selection of multiple COTS since products have to be integrated together within the same system architecture. The development of COTS-based systems

is known to require simultaneous tradeoffs among COTS, requirements and architectural constraints [Ncube and Maiden, 2000]. The need to perform tradeoffs is due to mismatches among these aspects. In this thesis we focused on the requirements view of the mismatch problem. In terms of architectural mismatch, early work in this direction can be found in [Garlan et al., 1995]. They describe that the sources of architectural mismatch are due to a set of assumptions concerning: the nature of the components, the nature of the connectors, the global architectural structure, and the construction process. We believe that architectural mismatch handling is an important extension to be addressed in our method. For instance, future developments of TAOS should explicitly analyse risks such as non-interoperability between products and describe suitable resolutions for such risks in the risk management taxonomy.

### Cost analysis

A key benefit organizations aim to achieve with the COTS-based paradigm is reducing overall system development costs. Despite the importance of economic factors, this issue has been quite neglected in the COTS literature. One of the few research done in the cost analysis area is the COCOTS framework [Abts et al., 2000]. This framework provides preliminary results towards a cost estimation model for COTS-based development. We believe that further research effort is still necessary to develop an effective model to estimate the real costs involved with the use of COTS products, and hence allow organizations to objectively measure the economic benefits of using COTS.



# Bibliography

- [Rad, ] Radicati group, IBM lotus and microsoft - corporate messaging market analysis. <http://www.radicati.com>.
- [RFC, ] Rfc 2822 - internet message format. <http://www.ietf.org/rfc/rfc2822.txt>.
- [ERP, 2005] (2005). PMP Research. [www.evaluationcentre.com](http://www.evaluationcentre.com).
- [Ten, 2005] (2005). Tender Management. <http://www.telelogic.com>.
- [Abts et al., 2000] Abts, C., Boehm, B. W., and Clark, E. B. (2000). COCOTS: a COTS software integration cost model - model overview and preliminary data findings. In *Proceedings of the European Software Control and Metrics Conference*.
- [Albert and Brownsword, 2002] Albert, C. and Brownsword, L. (2002). Evolutionary process for integrating COTS-based systems EPIC: An overview. Technical report, COTS-Based Systems Initiative - Carnegie Mellon University.
- [Alves et al., 2005] Alves, C., , Franch, X., Carvallo, J., and Finkelstein, A. (2005). Using goals and quality models to support the matching analysis during COTS selection. In *Proceedings of the Fourth International Conference on COTS-Based Software Systems*.
- [Alves, 2003] Alves, C. (2003). COTS-based requirements engineering. In Cechich A., Piattini M., V. A., editor, *Component-Based Software Quality: Methods and Techniques*, volume 2693 of *Lecture Notes in Computer Science*, pages 21–39. Springer Verlag.
- [Alves and Castro, 2001] Alves, C. and Castro, J. (2001). CRE: A systematic method for cots selection. In *Proceedings of the 15th Brazilian Symposium on Software Engineering*.
- [Alves and Finkelstein, 2002] Alves, C. and Finkelstein, A. (2002). Negotiating requirements for COTS selection. In *Proceedings of the RE Workshop on Requirements Engineering: Foundation for Software Quality*.
- [Alves and Finkelstein, 2003] Alves, C. and Finkelstein, A. (2003). Investigating conflicts in COTS decision making. In Chang, S., editor, *International Journal of Software Engineering and Knowledge Engineering*, volume 13, pages 473–493. World Scientific Publishing Company.

## BIBLIOGRAPHY

- [Anton, 1997] Anton, A. (1997). *Goal Identification and Refinement in the Specification of Software-Based Information Systems*. PhD thesis, Georgia Institute of Technology.
- [Bahsoon and Emmerich, 2004] Bahsoon, R. and Emmerich, W. (2004). Evaluating the stability of software architectures with real options theory. In *20th IEEE International Conference on Software Maintenance*. IEEE Press.
- [Basili and Boehm, 2001] Basili, V. and Boehm, B. (2001). COTS-based systems top 10 list. *IEEE Computer*, 34(5):91–93.
- [Basili et al., 1994] Basili, V., Caldiera, G., and Rombach, D. (1994). Goal/question/metric paradigm. In *Encyclopedia of Software Engineering*. Vol. 1.
- [Bertoa and Vallecillo, 2002] Bertoa, M. and Vallecillo, A. (2002). Quality attributes for COTS components. In *6th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering*, pages 54–66.
- [Beus-Dukic, 2000] Beus-Dukic, L. (2000). Non-functional requirements for COTS software components. In *Continuing Collaborations for Successful COTS Development Workshop in ICSE2000*.
- [Boehm, 1988] Boehm, B. (1988). A spiral model of software development and enhancement. *IEEE Computer*, 21(5):61–72.
- [Boehm, 1991] Boehm, B. (1991). Software risk management: Principles and practices. *IEEE Software*, 8(1):32–41.
- [Boehm, 2000] Boehm, B. (2000). Requirements that handle IKIWISI, COTS, and rapid change. *IEEE Computer*, 33(7):99–102.
- [Boehm and Abts, 1999] Boehm, B. and Abts, C. (1999). COTS integration: Plug and pray? *IEEE Computer*, 32(1):135–138.
- [Boehm and Port, 2004] Boehm, B. and Port, D. (2004). Risk-based strategic software design: How much COTS evaluation is enough? In *Third International Workshop on Economics-Driven Software Engineering Research*, pages 183–198.
- [Boehm et al., 2003] Boehm, B., Port, D., Yang, Y., and Bhuta, J. (2003). Not all CBS are created equally: COTS-intensive project types. In *Second International Conference on COTS-Based Software Systems*, pages 36–50. Springer-Verlag.
- [Bolchini and Paolini, 2004] Bolchini, D. and Paolini, P. (2004). Goal-driven requirements analysis for hypermedia-intensive web applications. *Requirements Engineering Journal*, 9(2):85–103.
- [Brereton, 2004] Brereton, P. (2004). The software customer/supplier relationship. *Communications of ACM*, 47(2):77–81.

## BIBLIOGRAPHY

- [Brereton and Budgen, 2000] Brereton, P. and Budgen, D. (2000). Component-based systems: A classification of issues. *IEEE Computer*, 33(11):54–62.
- [Brereton et al., 2002] Brereton, P., Linkman, S., Thomas, N., Boegh, J., and Panfilis, S. (2002). Software components - enabling a mass market. In *10th International Workshop on Software Technology and Engineering Practice*.
- [Brinkkemper, 1996] Brinkkemper, S. (1996). Method engineering: engineering of information systems development methods and tools. In *Information and Software Technology 38*.
- [Brown and Wallnau, 1998] Brown, A. and Wallnau, K. (1998). The current state of CBSE. *IEEE Software*, 15(5):37–46.
- [Brownsword et al., 2000] Brownsword, L., Oberndorf, T., and Sledge, C. (2000). Developing new processes for COTS-based systems. *IEEE Software*, 17(4):48–55.
- [Bush, 2003] Bush, D. (2003). The AFTN message switch case study. In *Requirements Engineering Specialist Group Event: COTS Integration: why you need requirements*.
- [Carney, 1998] Carney, D. (1998). COTS evaluation in the real world. SEI Interactive.
- [Carney, 2002] Carney, D. (2002). *Requirements and Components Chapter, In Building Systems from Commercial Components*. Addison-Wesley Longman Publishing.
- [Carney and Long, 2000] Carney, D. and Long, F. (2000). What do you mean by COTS? finally, a useful answer. *IEEE Software*, 17(2).
- [Carvallo et al., 2003] Carvallo, J., Franch, X., and Quer, C. (2003). Defining a quality model for mail servers. In *2nd International Conference on COTS-Based Software Systems*, pages 51–61.
- [Castro et al., 2001] Castro, J., Mylopoulos, J., Alencar, F., and Cysneiros, G. (2001). Integrating organizational requirements and object oriented modeling. In *5th IEEE International Symposium on Requirements Engineering*, pages 146–153.
- [Cechich and Piattini, 2004] Cechich, A. and Piattini, M. (2004). On the measurement of COTS functional suitability. In *Proceedings of the Third International Conference on COTS-Based Software Systems*.
- [Charette, 1990] Charette, R. (1990). *Software Engineering Risk Analysis and Management*. McGraw-Hill, New York.
- [Chung and Cooper, 2003] Chung, L. and Cooper, K. (2003). A knowledge-based cots-aware requirements engineering approach. In *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering*.

## BIBLIOGRAPHY

- [Chung et al., 1999] Chung, L., Nixon, B. A., Yu, E., and Mylopoulos, J. (1999). *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishing.
- [Clark and Torchiano, 2004] Clark, B. and Torchiano, M. (2004). COTS terminology and categories: Can we reach a consensus? In *Proceedings of the Third International Conference on COTS-Based Software Systems*, pages 4–5.
- [Cooper and Chung, 2002] Cooper, K. and Chung, L. (2002). A COTS-aware requirements engineering and architecting approach: Defining system level agents, goals, requirements and architecture. Technical report, Department of Computer Science, University of Texas at Dallas.
- [Dardenne et al., 1993] Dardenne, A., Lamsweerde, A., and Fickas, S. (1993). Goal-directed requirements acquisition. In *Science of Computer Programming*, volume 20, pages 3–50.
- [Dawson et al., 2003] Dawson, R., Bones, P., Oates, B., Brereton, P., Azuma, M., and Jackson, M. (2003). Empirical methodologies in software engineering. In *Eleventh Annual International Workshop on Software Technology and Engineering Practice*, pages 52 – 58.
- [Dean and Vigder, 2000] Dean, J. and Vigder, M. (2000). COTS software evaluation techniques. In *NATO Information Systems Technology Panel Symposium on Commercial Off-the-Shelf Products in Defence Applications*.
- [Deifel, 1998] Deifel, B. (1998). Requirements engineering for complex COTS. In *4th International Workshop on Requirements Engineering: Foundation for Software Quality*.
- [DeLine, 1999] DeLine, R. (1999). A catalog of techniques for resolving packaging mismatch. In *Proceedings of the 1999 symposium on Software reusability*, pages 44–53. ACM Press.
- [Dix et al., 1998] Dix, A., Finley, J., Abowd, G., and Beale, R. (1998). *Human-Computer Interaction*. Prentice-Hall.
- [Dyer and Sarin, 1979] Dyer, J. and Sarin, R. (1979). Measurable multiattribute utility functions. *Operations Research*, 27(4):810–822.
- [Easterbrook, ] Easterbrook, S. Resolving requirements conflicts with computer-supported negotiation. In *Social and Technological Issues in Requirements Engineering*.
- [Easterbrook, 1991] Easterbrook, S. (1991). *Elicitation of Requirements from Multiple Perspectives*. PhD thesis, Imperial College of Science Technology and Medicine, University of London.
- [Easterbrook et al., 1993] Easterbrook, S., Beck, E., , Goodlet, J., and Plowman, L. (1993). A survey of empirical studies of conflict. In *In CSCW: Cooperation or Conflict?* Springer-Verlag.

## BIBLIOGRAPHY

- [Egyed and Boehm, 1999] Egyed, A. and Boehm, B. (1999). Comparing software system requirements negotiation patterns. *Journal for Systems Engineering*, 6(1).
- [Egyed et al., ] Egyed, A., Medvidovic, N., and Gacek, C. A component-based perspective of software mismatch detection and resolution. In *IEE Software Engineering*, volume 147.
- [Engert and Clapp, 2001] Engert, P. and Clapp, J. (2001). Common risks and risk mitigation actions for management of a COTS-based system. In *The Edge Perspectives - MITRE publications*, volume 2.
- [Farbey and Finkelstein, 2001] Farbey, B. and Finkelstein, A. (2001). Software acquisition: A business strategy analysis. In *International Symposium on Requirements Engineering*, pages 76–83.
- [Fenton, 1994] Fenton, N. (1994). *Software Metrics - A Rigorous Approach*. Chapman and Hall.
- [Finkelstein, 2001] Finkelstein, A. (2001). CAPSA and its implementation report to the audit committee and the board of scrutiny.
- [Finkelstein et al., 1994] Finkelstein, A., Gabbay, D., Hunter, A., Kramer, J., and Nu-seibeh, B. (1994). Inconsistency handling in multiperspective specifications. *IEEE Transactions of Software Engineering*, 20(8):569–578.
- [Finkelstein and Spanoudakis, 1996] Finkelstein, A. Ryan, M. and Spanoudakis, G. (1996). Software package requirements and procurement. In *8th International Workshop on Software Specification Design*.
- [Fisher and Ury, 1992] Fisher, R. and Ury, W. (1992). *Getting to Yes: Negotiating Agreement Without Giving In*. Boston: Houghton Mifflin Co.
- [Frair, 1995] Frair, L. (1995). Student peer evaluations using the analytic hierarchy process method. Foundation Coalition, Department of Industrial Engineering, University of Alabama.
- [Franch and Carvallo, 2002] Franch, X. and Carvallo, J. (2002). A quality model based approach for describing and evaluating software packages. In *Proceedings of the 2002 IEEE Joint International Requirements Engineering Conference*.
- [Franch and Maiden, 2003] Franch, X. and Maiden, N. (2003). Modelling component dependencies to inform their selection. In *Proceedings of the Second International Conference on COTS-Based Software Systems*, pages 81–91. Springer-Verlag.
- [Garlan et al., 1995] Garlan, D., Allen, R., and Ockerbloom, J. (1995). Architectural mismatch, or, why it's hard to build systems out of existing parts. In *Proceedings of the 17th International Conference on Software Engineering*, pages 179–185.

## BIBLIOGRAPHY

- [Giesen and Volker, 2003] Giesen, J. and Volker, A. (2003). Requirements interdependencies and stakeholders preferences. In *Proceedings of the 2003 IEEE Joint International Requirements Engineering Conference*.
- [Glass and Vessey, 1995] Glass, R. and Vessey, I. (1995). Contemporary application-domain taxonomies. *IEEE Software*, 12(4):63–76.
- [Handfield and Nichols, 1998] Handfield, R. and Nichols, E. (1998). *Introduction to Supply Chain Management*. Prentice Hall, Upper Saddle River, NJ.
- [Hausen, 2003] Hausen, H. (2003). Standardized acceptance testing. In *International Workshop on COTS and Product Software: Why Requirements are so Important. In Conjunction with 11th RE*.
- [Hooks and Farry, 2001] Hooks, I. and Farry, K. (2001). *Customer-Centred Products*. American Management Association.
- [Hull et al., 2005] Hull, E., Jackson, K., and Dick, J. (2005). *Requirements Engineering*. Springer, London, UK.
- [Jackson, 2001] Jackson, M. (2001). *Problem frames: analyzing and structuring software development problems*. Addison-Wesley.
- [Jacobson et al., 1999] Jacobson, I., Rumbaugh, J., and Booch, G. (1999). *The Unified Software Development Process*. Addison-Wesley, Harlow.
- [Jeanrenaud and Romanazzi, 1904] Jeanrenaud, J. and Romanazzi, P. (1904). Software product evaluation: A methodological approach. In *Software Quality Management II: Building Software into Quality*, pages 59–69.
- [Karlsson and Ryan, 1997] Karlsson, J. and Ryan, K. (1997). A cost-value approach for prioritizing requirements. *IEEE Software*, 14(5):67–74.
- [Kato et al., 2003] Kato, J., Nagata, M., Yamamoto, S., Saeki, M., Kaiya, H., Horai, H., Ohnishi, A., Komiya, S., and Watahiki, K. (2003). PAORE: Package oriented requirements elicitation. In *Tenth Asia-Pacific Software Engineering Conference Software Engineering Conference*, pages 17–27.
- [Keeney and Raiffa, 1993] Keeney, R. and Raiffa, H. (1993). *Decision with Multiple Objective: Preferences and Value Tradeoffs*. Wiley, New York.
- [Kitchenham et al., 1997] Kitchenham, B., Linkman, S., and Law, D. (1997). DESMET: A methodology for evaluating software engineering methods and tools. In *IEE Computing and Control Journal*.
- [Koch, 2002] Koch, C. (2002). The ABCs of ERP. <http://www.cio.com/research/erp/edit/erpbasics.html>.

## BIBLIOGRAPHY

- [Kontio, 1995] Kontio, J. (1995). OTSO: A systematic process for reusable software component selection. Technical report, University of Maryland.
- [Kontio, 1996] Kontio, J. (1996). A case study in applying a systematic method for COTS selection. In *18th International Conference on Software Engineering*, pages 201–209.
- [Kontio, 1997] Kontio, J. (1997). The riskit method for software risk management, version 1.00. Technical report.
- [Kreitner and Kinicki, 2002] Kreitner, R. and Kinicki, A. (2002). *Organizational Behaviour*. McGrawHill.
- [Kumar, 2001] Kumar, K. (2001). Technology for supporting supply chain management. volume 44, pages 58–61.
- [Lamsweerde, 2001] Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour. In *5th IEEE International Symposium on Requirements Engineering*, pages 249–261. IEEE Computer Society.
- [Lamsweerde, 2004] Lamsweerde, A. (2004). Goal-oriented requirements engineering: A roundtrip from research to practice. In *12th IEEE International Conference on Requirements Engineering*, pages 4–7.
- [Lamsweerde and Letier, 2000] Lamsweerde, A. and Letier, E. (2000). Handling obstacles in goal-oriented requirements engineering. *Software Engineering*, 26(10):978–1005.
- [Lamsweerde et al., 1998] Lamsweerde, A., Letier, E., and Darimont, R. (1998). Managing conflicts in goal-driven requirements engineering. *IEEE Transactions of Software Engineering*, 24(11):908–926.
- [Lauesen, 2004] Lauesen, S. (2004). COTS tenders and integration requirements. In *Proceedings of the 12th IEEE International Conference on Requirements Engineering*, pages 166–175.
- [Lawlis et al., 2001] Lawlis, P., Mark, K., Thomas, D., and Courtheyn, T. (2001). A formal process for evaluating COTS software products. *IEEE Computer*, 34(5):58–63.
- [Letier and Lamsweerde, 2002] Letier, E. and Lamsweerde, A. (2002). Deriving operational software specifications from system goals. In *10th ACM SIGSOFT Symposium on the Foundations of Software Engineering, Charleston*.
- [Lewis and Morris, 2004] Lewis, G. and Morris, E. (2004). From system requirements to COTS evaluation criteria. In *2nd International Conference on COTS-Based Software Systems*, pages 159–168.
- [Li et al., 2004a] Li, J., Conradi, R., Slyngstad1, O., Torchiano, M., Morisio, M., and Bunse, C. (2004a). A state-of-practice survey on risk management in off-the-shelf

## BIBLIOGRAPHY

- component-based development. In *10th IEEE International Metrics Symposium*, pages 72–83.
- [Li et al., 2004b] Li, J., Finn, B., Conradi, R., and Kampenes, V. (2004b). An empirical study of variations in COTS-based software development processes in norwegian it industry. In *10th IEEE International Metrics Symposium*, pages 72–83.
- [Louis, 2003] Louis, R. (2003). Risk management of COTS based system development. In *Component-Based Software Quality - Methods and Techniques, LNCS Vol. 2693*, pages 352–373. Springer.
- [Maciaszek, 2001] Maciaszek, L. (2001). *Requirements Analysis and System Design*. Pearson Education Limited.
- [Maciaszek and Liong, 2004] Maciaszek, L. and Liong, B. (2004). *Practical Software Engineering - A Case-Study Approach*. Addison-Wesley.
- [Maiden, 1998] Maiden, N. (1998). CREWS-SAVRE: Scenarios for acquiring and validating requirements. *Automated Software Engineering*, 5(4):419–446.
- [Maiden et al., 2004] Maiden, N., Jones, S., Manning, S., Greenwood, J., and Renou, L. (2004). Model-driven requirements engineering: Synchronising models in an air traffic management case study. In *16th Conference on Advanced Information Systems Engineering*, pages 368–383.
- [Maiden and Ncube, 1998a] Maiden, N. and Ncube, C. (1998a). Acquiring COTS software selection requirements. *IEEE Software*, 15(2):46–56.
- [Maiden and Ncube, 1998b] Maiden, N. and Ncube, C. (1998b). Acquiring COTS software selection requirements. In *IEEE Software*, volume 15, pages 46–56.
- [Maiden and Rugg, 1996] Maiden, N. and Rugg, G. (1996). ACRE: Selecting methods for requirements acquisition. *Software Engineering Journal*, 11(3):183–192.
- [Marvin et al., 1993] Marvin, J., Carr, M., Konda, S., Monarch, I., Ulrich, F., and Walker, C. (1993). Taxonomy-based risk identification. Technical report.
- [McIlroy, 1968] McIlroy, M. (1968). Mass production software components. In *Report on a conference sponsored by the NATO Science Committee*.
- [Meyers and Oberndorf, 2001] Meyers, B. and Oberndorf, P. (2001). *Managing Software Adquisition: Open Systems and COTS Products*. Addison Wesley.
- [Morisio et al., 2000] Morisio, M., Seaman, C., Basili, V., Parra, A., Kraft, S., and Condon, S. (2000). Investigating and improving a COTS-based software development. In *22nd International Conference on Software Engineering*, pages 32–41.



## BIBLIOGRAPHY

- [Morisio et al., 2002] Morisio, M., Seaman, C., Basili, V., Parra, A., Kraft, S., and Condon, S. (2002). COTS-based software development: processes and open issues. *Journal of Systems and Software*, 61(3):189–189.
- [Morisio and Tsoukiys, 1997] Morisio, M. and Tsoukiys, A. (1997). IUSWARE: A formal methodology for software evaluation and selection. *IEE Proceedings on Software Engineering*, 144(4):162–174.
- [Ncube, 2000] Ncube, C. (2000). *A Requirements Engineering Method for COTS-Based Systems Development*. PhD thesis, City University.
- [Ncube and Dean, 2002] Ncube, C. and Dean, J. (2002). The limitations of current decision-making techniques in the procurement of COTS software components. In *Proceedings of the 1st International Conference on COTS-Based Software System*, volume 2255, pages 176–187.
- [Ncube and Maiden, 1999] Ncube, C. and Maiden, N. (1999). PORE: Procurement-oriented requirements engineering method for the component based systems engineering development paradigm. In *Proceedings of 1999 International Workshop on Component Based Software Engineering*.
- [Ncube and Maiden, 2000] Ncube, C. and Maiden, N. (2000). COTS software selection: The need to make tradeoffs between system requirements, architectures and cots components. In *Workshop on Continuing Collaborations for Successful COTS Development*.
- [Neumann and Morgenstern, 1947] Neumann, V. and Morgenstern, O. (1947). *Theory of Games and Economics Behavior*. Princeton University Press, Princeton.
- [Ning, 1999] Ning, Q. (1999). A component model proposal. In *1999 International Workshop on Component-Based Software Engineering*.
- [Nuseibeh and Easterbrook, 2000] Nuseibeh, B. and Easterbrook, S. (2000). Requirements engineering: a roadmap. In *The Future of Software Engineering*, pages 35–46. ACM Press.
- [Nuseibeh, 1994] Nuseibeh, C. (1994). *A Multi-Perspective Framework for Method Integration*. PhD thesis, Imperial College of Science Technology and Medicine, University of London.
- [Petrosian and Zenkevich, 1996] Petrosian, L. and Zenkevich, N. (1996). *Game Theory (Series on Optimization)*. World Scientific.
- [Pfleeger, 1998] Pfleeger, S. (1998). *Software Engineering: Theory and Practice*. Prentice-Hall.
- [Port and Chen, 2004] Port, D. and Chen, S. (2004). Assessing COTS assessment: How much is enough? In *2nd International Conference on COTS-Based Software Systems*, pages 183–198.

## BIBLIOGRAPHY

- [Postini, ] Postini. E-mail service provider. <http://www.postini.com>.
- [Rahim, 1985] Rahim, M. (1985). A strategy for managing conflict in complex organizations. page 84.
- [Raiffa, 1982] Raiffa, H. (1982). *The Art and Science of Negotiation*. Harvard University Press, Cambridge, MA.
- [Rashid and Kotonya, 2001] Rashid, A. and Kotonya, G. (2001). Risk management in component-based development: A separation of concerns perspective. In *27th Euromicro Conference*.
- [Robertson, 2003] Robertson, J. (2003). So, what is a content management system? <http://www.steptwo.com.au/>.
- [Robertson and Robertson, 1999] Robertson, S. and Robertson, J. (1999). *Mastering the requirements process*. ACM Press/Addison-Wesley Publishing Co.
- [Robinson, 1990] Robinson, W. (1990). Negotiation behaviour during requirements specification. In *Proceedings of the 12th IEEE International Conference on Software Engineering*. ACM Press.
- [Robinson and Volkov, 1997] Robinson, W. and Volkov, S. (1997). A meta-model for restructuring stakeholder requirements. In *Proceedings of the 19th international Conference on Software Engineering*, pages 140–149. ACM Press.
- [Robinson and Volkov, 1999] Robinson, W. and Volkov, V. (1999). Requirement conflict restructuring. Technical report, GSU CIS Working Paper 99-5, Georgia State University.
- [Rolland, 1999] Rolland, C. (1999). A goal driven approach to modelling COTS acquisition impacts. In *International Workshop on Enterprise Management and Resource Planning Systems: Methods, Tools and Architectures*.
- [Rolland and Prakash, 2001] Rolland, C. and Prakash, N. (2001). Matching ERP system functionality to customer requirements. In *Proceedings of the 2001 IEEE Joint International Requirements Engineering Conference*.
- [Russell and Norvig, 1994] Russell, S. and Norvig, P. (1994). *Artificial Intelligence: A Modern Approach*. Prentice Hall, New Jersey.
- [Saaty, 1980] Saaty, T. (1980). *The Analytic Hierarchy Process, Planning, Priority Setting, Resource Allocation*. McGraw-Hill, New York.
- [Salo and Hamalainen, 1997] Salo, A. and Hamalainen, R. (1997). On the measurement of preferences in the analytic hierarchy process. *Journal of Multi-Criteria Decision Analysis*, 6(6):309–319.

## BIBLIOGRAPHY

- [Santiago et al., 2002] Santiago, C., Dean, J., Morris, E., and Oberndorf, P. (2002). A process for COTS software product evaluation. In *Proceedings of the 1st International Conference on COTS-Based Software System*, pages 86–96.
- [Schoemaker and Waid, 1982] Schoemaker, P. and Waid, C. (1982). An experimental comparison of different approaches to determining weights in additive utility models. In *Management Science*, volume 28, pages 182–196.
- [Shaffer and McPherson, 2002] Shaffer, G. and McPherson, G. (2002). FAA COTS risk mitigation guide: Practical methods for effective COTS acquisition and life cycle support.
- [Simon, 1997] Simon, H. (1997). *Administrative Behaviour*. The Free Press.
- [Sommerville, 2004] Sommerville, I. (2004). *Software Engineering*. Pearson Education Limited.
- [Sommerville and Kontonya, 1998] Sommerville, I. and Kontonya, G. (1998). *Requirements Engineering: Processes and Techniques*. John Wiley & Sons, Inc.
- [Szyperski, 1998] Szyperski, C. (1998). *Component Software: Beyond Object-oriented Programming*. Addison-Wesley.
- [Torchiano et al., 2002] Torchiano, M., Jaccheri, L., Srensen, C., and Wang, A. (2002). COTS products characterization. In *14th International Conference on Software Engineering and Knowledge Engineering*, pages 335–338.
- [Torchiano and Morisio, 2004] Torchiano, M. and Morisio, M. (2004). Overlooked aspects of COTS-based development. *IEEE Software*, 21(2):88–93.
- [Tran and Liu, 1997] Tran, V. and Liu, T. (1997). A procurement-centric model for engineering component-based software systems. In *Proceedings of the Fifth International Symposium on Assessment of Software Tools*.
- [Vidger and Dean, 1997] Vidger, M. and Dean, J. (1997). An architectural approach to building systems from COTS software components. In *1997 Center for Advanced Studies Conference*.
- [Vitharana, 2003] Vitharana, P. (2003). Risks and challenges of component-based software development. *Communications of ACM*, 46(8):67–72.
- [Voas, 1998] Voas, J. (1998). Certifying off-the-shelf software components. *IEEE Computer*, 31(6).
- [Wallnau et al., 1998] Wallnau, K., Carney, D., and Pollack, B. (1998). How COTS software affects the design of COTS-intensive systems. SEI Interactive.

## BIBLIOGRAPHY

- [Wallnau et al., 2002] Wallnau, K., Hissam, S., and Seacord, R. (2002). *Building Systems from Commercial Components*. Addison-Wesley Longman Publishing.
- [Yen and Tiao, 1997] Yen, J. and Tiao, W. (1997). A systematic tradeoff analysis for conflicting imprecise requirements. In *Proceedings of the Third IEEE International Symposium on Requirements Engineering*, pages 87–97.
- [Yin, 1984] Yin, K. (1984). *Case Study Research: Design and Methods*. Sage, Beverley Hills, CA.
- [Yu, 1997] Yu, E. (1997). Towards modeling and reasoning support for early-phase requirements engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering*. IEEE Computer Society.
- [Zave, 1997] Zave, P. (1997). Classification of research efforts in requirements engineering. *ACM Computing Surveys*, 29(4):315–321.
- [Zave and Jackson, 1997] Zave, P. and Jackson, M. (1997). Four dark corners of requirements engineering. *ACM Transactions on Software Engineering and Methodology*, 6(1):1–30.
- [Zlotkin and Rosenschein, 1996] Zlotkin, G. and Rosenschein, J. (1996). Mechanisms for automated negotiation in state oriented domains. *Journal of Artificial Intelligence Research*, 5:163–238.