# TRACEABILITY APPROACH FOR CONFLICT DISSOLUTION IN HANDLING REQUIREMENTS CROSSCUTTING

JAMALUDDIN HJ JASMIS

UNIVERSITI TEKNOLOGI MALAYSIA

# TRACEABILITY APPROACH FOR CONFLICT DISSOLUTION IN HANDLING REQUIREMENTS CROSSCUTTING

JAMALUDDIN HJ JASMIS

A thesis submitted in fulfilment of the
requirements for the award of the degree of
Doctor of Philosophy (Computer Science)

Faculty of Computing
Universiti Teknologi Malaysia

FEBRUARY 2017

To my wife Pn. Zalina Zulkifli,
daughters and son
Aymeen Jessenia, Aleesya Nur Jannah
and Muhammad Aqeel Jawhar

# ACKNOWLEDGEMENT

I would like to express my deepest appreciation to my principle supervisor, Prof. Dr. Suhaimi Ibrahim and co-supervisor, Assoc. Prof. Dr. Zuraini Ismail for their continued support, patience, vivacious discussions, invaluable constructive comments and friendship that have kept my focus, and helped me in completing this research. I am also indebted to Allahyarham En. Saharbudin Naim Tahir Shah for his opinions back in the year 2009 which has led me in selecting this research area.

I also thank to my good friend, En. Ahmad Efendi Mohd Hassan for his willingness to share his thoughts and experiences in system development projects; special colleague Dr. Hazlifah Mohd Rusli and my head of department, Assoc. Prof. Aishah Ahmad @ Abu Mutalib for their utmost support, understanding, patience in hearing my sad stories and heartiest replacement on my lecturing works whenever I was struggling to complete my thesis; my two good friends, husband and wife – Dr. Sugumaran s/o Nallusamy and Dr. Kanmani d/o Munusamy for their heartiest support and experience-sharing during the process of completing this thesis. I am also thankful to University Technology MARA and Ministry of Higher Education for funding my studies for this PhD programme. My special gratitude goes to my parents, parents-in-law, sisters, brothers and friends, Dr. Mohamad Yusof Darus and Dr. Shamsul Jamel Elias for their encouragement and cheerful stories at all times. My special thanks also go to all the childhood and undergraduate friends, neighbours, staff at AIS, co-researchers at UTM and colleagues at FSKM, University Technology MARA who have helped me in my ways to complete this research.

At last and most importantly, this programme would not have been possible without the support of my wife, Pn. Zalina Zulkifli. I am thankful for her thesis editing contribution, endless patience and loves, consistent support and motivation for me to be where I am at present.

# ABSTRACT

Requirements crosscutting in software development and maintenance has gradually become an important issue in software engineering. There are growing needs of traceability support to achieve some possible understanding in requirements crosscutting throughout phases in software lifecycle. It is aimed to manage practical process in addressing requirements crosscutting at various phases in order to comply with industrial standard. However, due to its distinct nature, many recent works are focusing on identification, modularization, composition and conflict dissolution of requirements crosscutting which are mostly saturated at requirements level. These works fail to practically specify crosscutting properties for functional and non-functional requirements at requirements, analysis and design phases. Therefore, this situation leads to inability to provide sufficient support for software engineers to manage requirements crosscutting across the remaining development phases. This thesis proposes a new approach called the Identification, Modularization, Design Composition Rules and Conflict Dissolutions (IM-DeCRuD) that provides a special traceability to facilitate better understanding and reasoning for engineering tasks towards requirements crosscutting during software development and evolution. This study also promotes a simple but significant way to support pragmatic changes of crosscutting properties at requirements, analysis and design phases for medium sizes of software development and maintenance projects. A tool was developed based on the proposed approach to support four main perspectives namely requirements specification definition, requirements specification modification, requirements prioritization setting and graphics visualizing representation. Software design components are generated using Generic Modeling Environment (GME) with Java language interpreter to incorporate all these features. The proposed IM-DeCRuD was applied to an industrial strength case study of medium-scaled system called myPolicy. The tool was evaluated and the results were verified by some experts for validation and opinion. The feedbacks were then gathered and analyzed using DESMET qualitative method. The outcomes show that the IM-DeCRuD is applicable to address some tedious job of engineering process in handling crosscutting properties at requirements, analysis and design phases for system development and evolution.

## ABSTRAK

Keratan rentas keperluan dalam pembangunan perisian dan penyelenggaraan telah menjadi isu yang semakin penting dalam bidang kejuruteraan perisian. Terdapat permintaan yang semakin bertambah terhadap sokongan jejakan untuk memahami keratan rentas keperluan sepanjang fasa dalam kitar hayat perisian. Ianya bertujuan untuk mengurus proses yang praktikal dalam menangani keratan rentas keperluan di pelbagai fasa dalam usaha memenuhi piawaian indusri. Walaubagaimanapun, disebabkan tabiinya yang khusus, banyak kerja yang dijalankan pada masa kini menumpukan kepada pengenalan, modularisasi, komposisi dan penyelesaian konflik terhadap keratan rentas keperluan yang mana kebanyakannya tertumpu pada aras keperluan. Kerja-kerja ini gagal menentukan ciri-ciri keratan rentas secara praktikal bagi keperluan kefungsian dan bukan kefungsian pada fasa keperluan, analisis dan reka bentuk. Lantaran itu, situasi ini membawa kepada ketidakmampuan untuk menyediakan sokongan yang secukupnya untuk jurutera perisian mengendalikan keratan rentas keperluan merentasi baki fasa pembangunan. Tesis ini mencadangkan pendekatan baru yang dipanggil Pengenalan, Modularisasi, Peraturan komposisi reka bentuk dan penyelesaian konflik (IM-DeCRuD) yang menyediakan keupayaan mengesan yang khusus untuk membantu kefahaman dan pertimbangan lebih baik untuk aktiviti kejuruteraan ke arah keratan rentas keperluan semasa pembangunan dan evolusi perisian. Kajian ini juga menggalakkan cara yang mudah tetapi signifikan dalam menangani kesan perubahan secara pragmatik terhadap keratan rentas keperluan pada fasa keperluan, analisa dan reka bentuk untuk pembangunan perisian berukuran sederhana dan projek-projek penyelenggaraan. Alatan dibangunkan berdasarkan pendekatan yang dicadangkan untuk membantu empat perspektif utama iaitu pentakrifan spesifikasi keperluan, pengubahsuaian spesifikasi keperluan, aturan keutamaan keperluan dan perwakilan visualisasi grafik. Komponen-komponen reka bentuk perisian dihasil menggunakan persekitaran model generik (PMG) bersama penterjemah bahasa Java untuk merangkumi semua ciri-ciri ini. IM-DeCRuD yang dicadangkan telah dilaksanakan terhadap satu kes ujian industri yang berukuran sederhana yang dinamakan myPolicy. Alatan ini telah dinilai dan hasilnya telah disahkan oleh beberapa pakar untuk pengesahsahihan dan pandangan. Maklumbalas kemudian dikumpul dan dianalisa dengan menggunakan kaedah kualitatif DESMET. Dapatan kajian menunjukkan bahawa IM-DeCRuD boleh digunapakai untuk menangani proses kejuruteraan yang remeh dalam mengendalikan ciri-ciri keratan rentas keperluan pada fasa keperluan, analisis dan reka bentuk untuk pembangunan dan evolusi sistem.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

## LIST OF ABBREVATIONS

| | | |
|---|---|---|
| A-BT | - | Agent-Based Tactics |
| AGORA | - | Attributed Goal-Oriented Requirements Analysis |
| AIM | - | Aspect Identification Method |
| AOM | - | Analysis Object Models |
| AORE | - | Aspect Oriented Requirements Engineering |
| AOSD | - | Aspect Oriented Software Development |
| API | - | Application Programming Interface |
| APSs | - | Activity Pattern Specifications |
| ARCADE | - | Web Ontology Language |
| CMMI | - | Capability Maturity Model Integration |
| CVS | - | Concurrent Versions System |
| DISCERN | - | Dealing Separately With Crosscutting Concerns |
| DOM | - | Document Object Modeling |
| DOORS | - | Dynamic Object Oriented Requirements System |
| DOSS | - | Deriving Operational Software Specifications |
| DTEBS | - | Deriving Tabular Event-Based Specifications |
| EA | - | Early Aspect |
| EBT | - | Event Based Traceability |
| EBT-DP | - | Event Based Traceability with Design Patterns |
| FUR | - | Functional User Requirements |
| GBRAM | - | Goal-Based Requirement Analysis Method |
| GCT | - | Goal – Centric Traceability |
| GOIG | - | Goal-Oriented Idea Generation |
| GORE | - | Goal Oriented Requirement Engineering |
| GSTH | - | General System Thinking Heuristics |
| HB | - | History Based |
| IDE | - | Integrated Development Environment |

| | | |
|---|---|---|
| IM- DeCRuD | - | Identification, Modularization, Design Composition Rules And Conflict Dissolutions |
| IR | - | Information Retrieval |
| IREQ | - | Inter-Requirement Traceability Rule |
| ISO | - | International Organization for Standardization |
| KDE | - | K Desktop Environment |
| LSI | - | Latent Semantic Indexing |
| MDE | - | Model Driven Engineering |
| MOOR2M | - | Model-based Object-Oriented Requirement Metamodel |
| MOORM | - | Model-based OO Requirement Model |
| MORE | - | Model-based Object-oriented approach to Requirement Engineering |
| NFUR | - | Non-Functional User Requirements |
| NLP | - | Natural Language Processing |
| OMG | - | Object Management Group |
| RB | - | Rule Based |
| RE | - | Requirements Engineering |
| RMRT | - | Reference Models for Requirements Traceability |
| RSD | - | Requirements Statement Documents |
| RT | - | Requirements Traceability |
| RTM | - | Requirements Traceability Matrix |
| RTM | - | Requirements Traceability Matrix |
| RTOM | - | Requirement-To-Object-Model |
| RTTC | - | Requirements Traceability and Transformation Conformance |
| SAX | - | Simple API For XML |
| SBT | - | Scenario Based Traceability |
| SRS | - | Software Requirements Specification |
| UCD | - | Use Case Documents |
| UMLNFR/AUC | - | Unified Markup Language Non-Functional Requirements/Aspectual Use Case |

| | | |
|---|---|---|
| UNFR | - | Unified Software Development Process With Non-Functional Requirements |
| USDP | - | Unified Software Development Process |
| VSM | - | Vector Space Model |
| VVA | - | Visual Variability Analysis |
| XML | - | Extensible Markup Language |

# LIST OF APPENDICES

# CHAPTER 1

# INTRODUCTION

## 1.1 Overview

This chapter discusses the introduction to this research. First of all, brief introduction of requirements crosscutting is described. Consequently, background of the problem to be solved, problem statement, objective, scope, and also significance of the study are also described respectively.

## 1.2 Background of the Problem

Model is a representation that encapsulates details of a system pertaining to system structure or its processes. Model has been used to describe various angles of a system facilitated by Object-Oriented methods [1]. Besides, models are the main artifacts in Model Driven Engineering (MDE); they can potentially be included on several levels of abstraction as well as transformations to different code or models [1, 2]. Generally, the maintenance and development of a full-scale software system are associated with a great deal of software models that include requirements, designs, implementations, testing suites and maintenance records.

Branching from MDE, Requirements Engineering (RE) which deals with requirements model is involved with requirements eliciting and analysis [2]. This is known as requirements engineering (RE). RE is referred to as systematic requirement analysis as it involves systematic requirements captured on the specification made by the stakeholders [2]. RE is a multi-disciplinary activity that implements various stages of development techniques and tools for application domains of various types [3]. In software development process as well as in the management of software change, RE is the front-end activity to be regarded.

As a result of requirements analysis, particular unique requirements are extracted and segregated in systematic form referred to as concerns that is of interest to one or more stakeholders. These concerns are in the forms of functional (system capabilities) or non-functional (system properties) that may affect one or more concerns. In other word, a system capability may be described by one or more related properties. For example, a stakeholder's functional requirement with a capability of handling a user on-line transaction might be described by some properties i.e. the non-functional requirements such as user's response time within acceptable limit, with appropriate security features and affordable workload. This type of scenario is called tangling.

In another situation, a concern of non-functional requirement may describe properties for several other functional requirements in order for the functional requirements to remain useful. For example, the performance as a property of a system would be applied to several other functional requirements i.e. concerns with similar or different specifications. This type of scenario is called scattering. Thus, RE which deals with requirements model is involved with concerns that may be scattered amongst other concerns as well as tangled within a concern.

In the context of RE, the above perspectives of tangling and scattering are also known as requirements crosscutting. Crosscutting concerns are related to each other within artifact as well as correlated artifacts across multiple phases [4]. Consequently, any changes to crosscutting concerns may yield direct or indirect impact to other artifacts. As such, it is necessary to have an approach supported by tool to store relationship dependencies since traceability is highly considered among artifacts in MDE to support understanding and maintenance of software systems [4]. Furthermore, design quality is difficult to be assured if obscure relationships exist, involving requirements and design for comparison. Without clear relationships of these artifacts evaluation of the quality of design is almost impossible to be done [5].

Nevertheless, new RE research domain, Aspects Oriented Requirements Engineering (AORE) dedicatedly deals with crosscutting concerns in term of processes like identification, modularization, composition and analysis of their effect on other concerns in documentation [6]. AORE capabilities is being supported further by Aspect Oriented Software Development (AOSD) in which crosscutting concerns in requirement can be consistently addressed across stages of software development lifecycle [7].

This research is inspired by research efforts in Requirements Traceability (RT) taking into consideration the crosscutting concept and design. RT is a sub-discipline under Requirement Engineering (RE) which is based on the capability to describe and follow the flow of a requirement in both forward and backward directions [8]. Forward traceability is related to the mapping of among the requirements or to the work products that implements them. Meanwhile, backward traceability supports mapping from the work product right back to its correspondences as well as tracing each requirement back to its source.

## 1.3 Statement of the Problem

Current object oriented analysis failed to identify and modularize crosscutting concerns [6]. Their characteristics are difficult to be identified as they may be obvious or subtle. In addition, crosscutting concerns identification involves in a tedious tracing process towards large amount of specification documents. Worse case, interview transcript documents are generally lack of accuracy and vague. In addition, crosscutting concerns are usually scattered across documents that complicate their identification [9, 10]. It is apparent that common requirement may occur in different segments of the documents and represent in other word.

The traceability of crosscutting associated properties on artifacts (or models) at upcoming stages of development, particularly design has not been properly identified [11]. Still, there are approaches that provide resolution pertaining to crosscutting concerns in initial stages of the software development process. Different approaches are being used to represent crosscutting properties for all stages [7]. This is in line with the situation where different properties of crosscutting concerns need to be specified when created. However, it seems that there is no approach that can support seamless and significant transformation of different level of crosscutting concerns artifacts in software development and **evolution** processes. Software engineers might not be provided with sufficient guides to deal with crosscutting issues throughout development stages. [7].

On the other hand, many researches address the crosscutting concerns conflict analysis at the requirement level due to its potential issues in which documentations are always related to high-level non-functional requirements. This is due to scattering and tangling properties of crosscutting requirements have direct impact on conflicts. However, there is lack of traceability research that is directed towards handling conflicts that may arise during crosscutting concerns composition at later stage. Furthermore, providing solutions to conflicts is crucial due to the problematic that issues contribute undesirable impacts on the full system and its composition. Poor in conflicts resolution will result in producing poor architecture [12]. It is also reported that consistencies and constraints of global scoped requirements is still largely unsupported [11].

Existing approaches have made some contribution to various aspects of traceability but lack for the purpose of handling crosscutting concerns at various stages. The aim of this research is to produce an improved software traceability approach to provide support for the crosscutting concern-driven evolution procedure among requirements, analysis and its association at the design level. With this, the general question that this research attempt to answer is:

"How to support evolution procedure including identification, modularization, propagation as well as conflict analysis of crosscutting concerns components between requirements, analysis and its correspondence in design phase via an improved traceability approach?"

To properly provide a solution this question, a number of research questions which address this issue are formulated, which are the following:

(a) **RQ1:** What are the requirements crosscutting approaches?
  (i) What are the state-of-arts of the approaches?
  (ii) What are the suitability of these approach – when, where to use?
  (iii) What are the advantages and disadvantages of these approaches?

(b) **RQ2:** Why engineers' tasks are still not able to be accommodated by the existing approaches?

(c) **RQ3:** How to provide an improved traceability approach to support engineering tasks for requirements, analysis and design crosscutting?

(d) **RQ4:** How the proposed approach can be used by the engineers?

(e) **RQ5:** How to evaluate the proposed approach to ensure its defined criteria?
  (i) In order to identify the applicability of the proposed approach, what is the most suitable evaluation method?
  (ii) How to conduct it?
  (iii) How these obtained results can be analyzed?

## 1.4 Objective of Study

The research has the following objectives:

(a) To analyze and emphasize on crosscutting criteria applied to requirements, analysis and design phases.

(b) To formulate and construct a traceability approach for requirements, analysis and design crosscutting.

(c) To develop a tool that supports the proposed approach.

(d) To evaluate the applicability of the approach proposed by applying it on a medium-scaled, standard industrial-strength application.

International Software Benchmarking Standards (ISBSG) defines the term applicability as process conformance. It is one form of quality management audit to benchmark the proposed approach against some evaluation criteria [13].

## 1.5 Scope of the Study

In order to produce an improved traceability approach, five research directions were inspired. They are the researches in Model Driven Engineering (MDE), Requirements Engineering, Requirements Traceability, Crosscutting Concerns and Aspect Oriented Requirements Engineering (AORE)/Aspect Oriented Software Development (AOSD). Those directions are presented here as the scope of the research subject in this research.

**(a) Model Driven Engineering**

Firstly, subject of this research is basically based on Model-Driven Engineering (MDE). MDE is the term used for development processes that are based on model (or artifacts) which opposites to code-centric [1]. In MDE, models are the prime artifacts and they may exist on multiple levels of abstractions and undergo transformations to other models and/or code. MDE enables fast system development, improved system quality, short time to market and software or hardware components reusability [14]. More explanation can be obtained in Section 2.2.

**(b) Requirements Engineering**

Secondly, this research is inspired by research efforts in Requirements Engineering (RE). Branching from MDE, the term of RE is also known as systematic requirement analysis as it involves systematic requirements gathering captured upon specification made by stakeholders. RE is said to be multi-disciplinary activity which implements several different stages of development techniques and tools for application domains of various types [3]. In software development process as well as in the management of software change, RE is the front-end activity to be regarded. Subsection 2.3 describes further on RE.

**(c) Requirements Traceability**

Thirdly, this research is inspired by research efforts in Requirements Traceability (RT). RT is a sub-discipline under Requirement Engineering (RE) which is based on the ability to describe and follow the life of a requirement in both ways of forward and backward direction [8]. Forward traceability is related to the mapping of among the requirements or to the work products that implements them. Meanwhile, backward traceability supports mapping from the work product right back to its correspondences as well as tracing each requirement back to its source. More explanation can be obtained in Section 2.4.

**(d) Crosscutting Concerns**

Fourthly, this research is also inspired by research efforts in Crosscutting Concerns. A requirement is a special kind of concern [15]. Concern can be defined as "anything that involved in a software system". It could be associated to system functionalities (functional) as well as properties (non-functional) [9]. There are two types of concerns, which are core (or base) concerns and crosscutting concerns [12]. Crosscutting concern is related to a scenario when a concern crosscuts or influence one or more of other concerns. More explanation can be obtained in Section 2.5.

**(e) Aspect Oriented Requirements Engineering/Aspect Oriented Software Development**

Lastly, this research is finally branching to the research efforts in Aspect Oriented Requirements Engineering/Aspect Oriented Software Development. Aspect Oriented Requirements Engineering (AORE) is relatively new area under RE domain [9]. AORE is directed to support crosscutting concerns by means of identification, modularization, composition and analysis of their influence on other requirements in the specification documents. Meanwhile, Aspect Oriented Software Development (AOSD) broadens the capability of AORE at each development stage in which it changes and expands available constructs and decision support among software engineer at each stage of software development life cycle [7]. Subsection 2.5.1 presents a discussion on the topic of AORE/AOSD and its state-of-the-art approaches can be found on the subsection 2.5.2.

This research will focus on object-oriented system to address the issue of crosscutting concerns handling for requirements, analysis and design artifacts. The outcome of this research will be evaluated to a medium-scaled, standard industrial-strength application to ensure its applicability.

## 1.5 Significance of the Study

Traceability feature is mainly applied in software development and evolution where its control and support is important in the context of crosscutting concerns at the requirement stage [16]. However, since crosscutting concerns rarely occur in isolation in such that they are related to other artifacts within a phase or across multiple phases, providing a traceability approach that can support explicit composition for crosscutting requirements in term of its mapping and influence on succeeding development stages [11, 17] is non-trivial. With this, changes to crosscutting concerns can have consequences for other artifacts, which are directly or indirectly related to it.

AOSD has been gradually accepted to be technique in software development and maintenance. As such, several aspect-oriented approaches have been proposed to specify crosscutting concerns at different phases in the software life cycle [4]. Since visibility of crosscutting concerns is an important traceability issue that needs to be appropriately addressed, there are numerous efforts conducted towards crosscutting concerns visibility at various stages [18, 19]. However visualizing crosscutting concern without underlying formal semantic and syntax are not amendable to automated tool support. As such, this research supports the visibility of user requirements at the high level abstracts with appropriate schemas and syntax. In addition, AOSD also accommodates visualization to be weaved together with conflict analysis in order to increasingly support requirements engineers' tasks in dealing with conflicting crosscutting concerns [18].

This research will contribute in providing a traceability approach to overcome the above challenges and opportunities for requirements, analysis and design phases. With the rising amount of support for crosscutting concerns at the particular design level, manipulation of crosscutting concerns at particularly the requirements level and recognition of their associated mappings will help to implement homogeneity within mainly aspect-oriented software development and maintenance processes.

## 1.6 Thesis Outline

This chapter covers some particular issues of requirements crosscutting in traceability approach. It also focuses on the limitations of the conventional approaches in dealing requirements crosscutting with maintenance process. It expresses a proposed approach of requirements crosscutting that able to improve traceability and maintenance process. The remaining of chapters will be organized as follow:

**Chapter 2**: This chapter discusses on the background information on requirements crosscutting. It starts with some preliminary studies on Model Driven

Engineering (MDE). This is followed by Requirement Engineering (RE) and Requirement Traceability (RT). In this chapter also several identified state-of-the-art requirements crosscutting approaches that related to Aspects Oriented Requirements Engineering (AORE)/Aspects Oriented Software Development (AOSD) are discussed. This chapter also highlights seven evaluation criteria that are used to compare the selected AORE/AOSD approaches. The results of this evaluation as well as the need to solve the current limitations for further research are presented.

**Chapter 3:** This chapter is used to describe on the research procedure, operational framework, assumption and limitations of the research and the schedule of this research. It also includes a brief description of a medium-scaled, industrial-strength case study and its significance that will be applied in this research. It also covers an overview of data gathering and analysis.

**Chapter 4**: It presents requirements crosscutting-driven on traceability approach to deal with maintenance process. This is followed by a comprehensive description on the proposed approach which describes the 3 main components in the said approach and the expected findings at the end.

**Chapter 5**: It explains the design and implementation of the approach's prototype tool that functions as a proof-of-concept. The prototype components are discussed in detail by describing the three important processes, namely as requirements boilerplates entries population, management and extend the saved requirements components to design elements.

**Chapter 6**: This chapter aims to furnish an in-depth example on the application of the approach on a medium-scaled, standard industrial-strength application. It begins with an outline of the chosen application. This is complimented by an explanation of identifying and analyzing requirements specifications obtained from the stakeholders. The description of the chosen application's high level software design and a discussion on linking requirements specifications to the software design are presented in the next section. After that, an explanation on the method of implementation of both the simple and the complex changes depending on the chosen cases is shown in the succeeding section. Next, the prototype tool based on the proposed approach is assessed for its practicability. The assessment criteria and methods that are explained and carried out on the approach are features modeling validation, the case study's results and briefing as well as demonstration sessions. This research provides assessment on the basis of qualitative findings. Qualitative outcomes are collected based on customer perception towards the demonstrated prototype tool. Lastly, a summary and presentation is given on the benefit of the application.

**Chapter 7**: This is a conclusion chapter that describes the research achievements and contributions. This is followed by the research summary and suggestions for research future works.

## REFERENCES


1. M. Parastoo and A. Jan, "Evaluating Quality in Model-Driven Engineering," in *Proceedings of the International Workshop on Modeling in Software Engineering*: IEEE Computer Society, 2007.
2. I. Galvão and A. Goknil, "Survey of Traceability Approaches in Model-Driven Engineering," Annapolis, MD, United states, 2007, pp. 313-324.
3. N. Bashar and E. Steve, "Requirements Engineering: A Roadmap," in *Proceedings of the Conference on The Future of Software Engineering* Limerick, Ireland: ACM, 2000.
4. B. Tekinerdogan, C. Hofmann, M. Aksit, and J. Bakker, "Metamodel for Tracing Concerns Across the Life Cycle," Vancouver, Canada, 2007, pp. 175-194.
5. S. T. Frezza, S. P. Levitan, and P. K. Chrysanthis, "Requirements-Based Design Evaluation," in *Annual ACM IEEE Design Automation Conference: Proceedings of the 32nd ACM/IEEE Conference on Design Automation*, 1995.
6. B. S. Ali and Z. M. Kasirun, "A Review on Approaches for Identifying Crosscutting Concerns," Phuket, Thailand, 2008, pp. 855-859.
7. P. Sanchez, L. Fuentes, A. Jackson, and S. Clarke, "Aspects at the Right Time," Tiergartenstrasse 17, Heidelberg, D-69121, Germany, 2007, pp. 54-113.
8. P. Valderas and V. Pelechano, "Introducing Requirements Traceability Support in Model-Driven Development of Web Applications," *Information and Software Technology*, vol. 51, pp. 749-768, 2009.
9. B. S. Ali and Z. M. Kasirun, "Developing Tool for Crosscutting Concern Identification Using NLP," Kuala Lumpur, Malaysia, 2008, p. IEEE.

10. A. Sampaio, A. Rashid, and P. Rayson, "Early-AIM: An Approach for Identifying Aspects in Requirements," in 13th IEEE International Conference on Requirements Engineering., 2005, pp. 487-488.

11. R. Awais, M. Ana, and A. João, "Modularisation and Composition of Aspectual Requirements," in Proceedings of the 2nd International Conference on Aspect-Oriented Software Development Boston, Massachusetts: ACM, 2003.

12. S. S. Khan and M. J.-u. Rehman, "A Survey on Early Separation of Concerns," in Proceedings of the 12th Asia-Pacific Software Engineering Conference: IEEE Computer Society, 2005.

13. L. Rollo, P. Morris, E. Wasylkowski, C. Dekkers, and P. Forselius, "International Software Benchmarking Standards Group (ISBSG) ver 1.0," 2008.

14. J. Dekeyser, P. Boulet, P. Marquet, and S. Meftali, "Model Driven Engineering for SoC co - Design," in The 3rd International IEEE-NEWCAS Conference, 2005, 2005, pp. 21-25.

15. L. Rosenhainer, "Identifying Crosscutting Concerns in Requirements Specifications," in Proceedings of OOPSLA Early Aspects, Vancouver, Canada, 2004.

16. S. T. Marta, M. Ana, A. Raquel, A. Fernando, and A. Joao, "A Traceability Method for Crosscutting Concerns with Transformation Rules," in Proceedings of the Early Aspects at ICSE: Workshops in Aspect-Oriented Requirements Engineering and Architecture Design: IEEE Computer Society, 2007.

17. L. K. Kit, C. K. Man, and E. Baniassad, "Isolating and Relating Concerns in Requirements Using Latent Semantic Analysis," ACM SIGPLAN Notices, vol. 41, pp. 383-396, 2006.

18. J. Hannemann, R. Chitchyan, and A. Rashid, "Analysis of Aspect-Oriented Software," in Lecture Notes in Computer Science: Springer Berlin / Heidelberg, 2004, pp. 154-164.

19. Y. Han, G. Kniesel, and A. B. Cremers, "A Meta Model and Modeling Notation for AspectJ," in Proceedings of the AOM workshop at AOSD, 2004, 2004.

20. Thomas Leveque, Jacky Estublier, and German Vega, "Extensibility and Modularity for Model Driven Engineering Environments," in IEEE International Conference and Workshop on the Engineering of Computer-Based Systems, 2009.

21. F. Jouault, J. Bézivin, and M. Barbero, "Towards an Advanced Model-Driven Engineering Toolbox," Innovations in Systems and Software Engineering, vol. 5, pp. 5-12, 2009.

22. D. C. Schmidt, "Model-Driven Engineering," IEEE Computer, vol. 39 No. 2, pp. 41-47, 2006.

23. Y. Yang, F. Xia, W. Zhang, X. Xiao, Y. Li, and X. Li, "Towards Semantic Requirement Engineering," Huangshan, China, 2008, pp. 67-71.

24. W. T. Tsai, Z. Jin, P. Wang, and B. Wu, "Requirement Engineering in Service-Oriented System Engineering," Hong Kong, China, 2007, pp. 661-668.

25. C.-W. Lu, W. C. Chu, C.-H. Chang, and C. H. Wang, "A Model-based Object-oriented Approach to Requirement Engineering (MORE)," Beijing, China, 2007, pp. 153-156.

26. H. Matthias, R. Franz, B. Stefan, and E. Alex, "Value-Based Selection of Requirements Engineering Tool Support," in Proceedings of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications: IEEE Computer Society, 2006.

27. S. Anwer and N. Ikram, "Goal Oriented Requirement Engineering: A Critical Study of Techniques," Bangalore, India, 2006, pp. 121-127.

28. J. Matthias and B. Stefan, "A Case Study on Value-Based Requirements Tracing," in Proceedings Of The 10th European Software Engineering Conference Held Jointly With 13th ACM SIGSOFT International Symposium On Foundations Of Software Engineering Lisbon, Portugal: ACM, 2005.

29. P. Lago, H. Muccini, and H. van Vliet, "A Scoped Approach to Traceability Management," Journal of Systems and Software, vol. 82, pp. 168-182, 2009.

30. Y. Yu, J. Jurjens, and J. Mylopoulos, "Traceability for the Maintenance of Secure Software," Beijing, China, 2008, pp. 297-306.

31. Y. Yu, J. Jurjens, and J. Schreck, "Tools for Traceability in Secure Software Development," Piscataway, NJ 08855-1331, United States, 2008, pp. 503-504.

32. A. Egyed and P. Grünbacher, "Identifying Requirements Conflicts and Cooperation: How Quality Attributes and Automated Traceability Can Help," *IEEE Software*, vol. 21, pp. 50-58, 2004.

33. H. El Ghazi, "MV - TMM: A Multi View Traceability Management Method," Turku, Finland, 2008, pp. 247-254.

34. F. Blaauboer, K. Sikkel, and M. N. Aydin, "Deciding to Adopt Requirements Traceability in Practice," Trondheim, Norway, 2007, pp. 294-308.

35. V. Kirova, N. Kirby, D. Kothari, and G. Childress, "Effective Requirements Traceability: Models, Tools, and Practices," *Bell Labs Technical Journal*, vol. 12, pp. 143-158, 2008.

36. A. Egyed, P. Grünbacher, M. Heindl, and S. Biffl, "Value-Based Requirements Traceability: Lessons Learned," New Delhi, India, 2007, pp. 115-120.

37. J. H. Hayes, A. Dekhtyar, S. K. Sundaram, E. A. Holbrook, S. Vadlamudi, and A. April, "REquirements TRacing On target (RETRO): Improving Software Maintenance Through Traceability Recovery," *Innovations in Systems and Software Engineering*, vol. 3, pp. 193-202, 2007.

38. A. De Lucia, R. Oliveto, and G. Tortora, "Assessing IR-Based Traceability Recovery Tools Through Controlled Experiments," *Empirical Software Engineering*, vol. 14, pp. 57-92, 2009.

39. P. Mäder, O. Gotel, and I. Philippow, "Rule-Based Maintenance of Post-Requirements Traceability Relations," Piscataway, NJ 08855-1331, United States, 2008, pp. 23-32.

40. I. Exman, "Software Component Completeness by Block-Diagonalized Traceability Matrices," Piscataway, NJ 08855-1331, United States, 2004, pp. 118-121.

41. J. H. Hayes, A. Dekhtyar, A. Holbrook, O. Dekhtyar, and S. Sundaram, "Will Johnny/Joanie Make A Good Software Engineer? Are Course Grades Showing The Whole Picture?," Turtle Bay, HI, United states, 2006, pp. 175-182.

42. C.-H. Jane, "Just Enough Requirements Traceability," in *Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC'06)*. vol. 01: IEEE Computer Society, 2006.

43. Y. Zhang, R. Witte, J. Rilling, and V. Haarslev, "Ontological Approach for the Semantic Recovery of Traceability Links Between Software Artefacts," *IET Software*, vol. 2, pp. 185-203, 2008.

44. R. Oliveto, "Traceability Management Meets Information Retrieval Methods - Strengths and Limitations," Athens, Greece, 2008, pp. 302-305.

45. S. Rochimah, W. M. N. W. Kadir, and A. H. Abdullah, "An Evaluation of Traceability Approaches to Support Software Evolution," Piscataway, NJ 08855-1331, United States, 2007, p. 4299902.

46. K. Inoue and M. Barker, "Accountability and Traceability in Global Software Engineering (ATGSE2007)," Los Alamitos, CA 90720-1314, United States, 2007, pp. 550-554.

47. P. Mäder, O. Gotel, and I. Philippow, "Enabling Automated Traceability Maintenance by Recognizing Development Activities Applied to Models," Piscataway, NJ 08855-1331, United States, 2008, pp. 49-58.

48. P. der, O. Gotel, and I. Philippow, "Enabling Automated Traceability Maintenance by Recognizing Development Activities Applied to Models," L'Aquila, Italy, 2008, pp. 49-58.

49. G. Cysneiros and A. Zisman, "Traceability and Completeness Checking for Agent-Oriented Systems," Fortaleza, Ceara, Brazil, 2008, pp. 71-77.

50. G. Zemont, "Towards Value-Based Requirements Traceability," in *Department of Computer Science* Chicago Illinois: DePaul University, 2005, p. 80.

51. A. De Lucia, F. Fasano, R. Oliveto, and G. Tortora, "ADAMS Re-Trace: A Traceability Recovery Tool," Manchester, United kingdom, 2005, pp. 32-41.

52. G. Ankit and R. Abhik, "Synthesis and Traceability of Scenario-Based Executable Models," in *Proceedings of the Second International Symposium on Leveraging Applications of Formal Methods, Verification and Validation*: IEEE Computer Society, 2006.

53. N. Aizenbud-Reshef, B. T. Nolan, J. Rubin, and Y. Shaham-Gafni, "Model Traceability," *IBM Systems Journal*, vol. 45, pp. 515-526, 2006.

54. N. Kececi, J. Garbajosa, and P. Bourque, "Modeling Functional Requirements to Support Traceability Analysis," Montreal, QC, Canada, 2006, pp. 3305-3310.

55. J. Grady, *System Requirements Analysis*: Academic Press, 2006.

56. A. Ahmad and M. A. Ghazali, "Documenting Requirements Traceability Information for Small Projects," Piscataway, NJ 08855-1331, United States, 2007, p. 4557711.

57. R. D. McDowall, "Validation of Spectrometry Software: The Proactive Use of a Traceability Matrix in Spectrometry Software Validation, Part I: Principles," Spectroscopy, vol. 23, 2008.

58. M. Kassab, O. Ormandjieva, and M. Daneva, "A Traceability Metamodel for Change Management of Non-Functional Requirements," Prague, Czech Republic, 2008, pp. 245-254.

59. N. B. A. Chung L., Yu E., Mylopoulos J. , Non-Functional Requirements in Software Engineering. Boston, MA: Kluwer Academic Publishers., 2000.

60. S. Ibrahim, "A Document-Based Software Traceability To Support Change Impact Analysis Of Object-Oriented Software," Ph.D Thesis, Universiti Teknologi Malaysia, June 2006, p. 151.

61. N. B. I. Suhaimi Ibrahim, Aziz Deraman, Malcolm Munro, "A Requirements Traceability to Support Change Impact Analysis," Asean Journal of Information Technology, vol. 4(4), pp. 345-355, April 2005 2005.

62. N. B. I. Suhaimi Ibrahim, Aziz Deraman, Malcolm Munro, "Implementing A Document-Based Requirements Traceability: A Case Study," in Proceedings of IASTED on Software Engineering, Austria, 2005, pp. 124-131.

63. N. B. I. Suhaimi Ibrahim, Aziz Deraman, Malcolm Munro, "A Software Traceability Validation for Change Impact Analysis of Object Oriented Software " in International Conference on SERP, Las Vegas, 2006, pp. 460-465.

64. J. Cleland-Huang, W. Marrero, and B. Berenbach, "Goal-Centric Traceability: Using Virtual Plumblines to Maintain Critical Systemic Qualities," IEEE Transactions On Software Engineering, vol. 34, pp. 685-699, 2008.

65. J. Cleland-Huang, R. Settimi, O. BenKhadra, E. Berezhanskaya, and S. Christina, "Goal-Centric Traceability for Managing Non-Functional Requirements," Saint Louis, MO, United states, 2005, pp. 362-371.

66. J. Fletcher and J. Cleland-Huang, "Softgoal Traceability Patterns," Raleigh, NC, United States, 2006, pp. 363-372.

67. P. G. Alexander Egyed, "Automating Requirements Traceability: Beyond The Record & Replay Paradigm," in 17th IEEE International Conference on Automated Software Engineering (ASE'02), 2002, pp. 163-171.

68. A. D. Lucia, F. Fasano, R. Oliveto, and G. Tortora, "Recovering Traceability Links in Software Artifact Management Systems Using Information Retrieval Methods," ACM Transactions on Software Engineering and Methodology, vol. 16, p. 1276934, 2007.

69. A. De Lucia, R. Oliveto, and P. Sgueglia, "Incremental Approach and User Feedbacks: A Silver Bullet for Traceability Recovery?," Philadelphia, PA, United states, 2006, pp. 299-308.

70. A. De Lucia, F. Fasano, R. Oliveto, and G. Tortora, "Can Information Retrieval Techniques Effectively Support Traceability Link Recovery?," Athens, Greece, 2006, pp. 307-316.

71. A. De Lucia, M. Di Penta, R. Oliveto, and F. Zurolo, "Improving Comprehensibility of Source Code via Traceability Information: A Controlled Experiment," Athens, Greece, 2006, pp. 317-326.

72. A. Marcus, J. I. Maletic, and A. Sergeyev, "Recovery of Traceability Links Between Software Documentation and Source Code," International Journal of Software Engineering and Knowledge Engineering, vol. 15, pp. 811-836, 2005.

73. A. De Lucia, F. Fasano, R. Oliveto, and G. Tortora, "Enhancing an Artefact Management System with Traceability Recovery Features," Chicago, IL, United states, 2004, pp. 306-315.

74. H.-Y. Jiang, T. N. Nguyen, I.-X. Chen, H. Jaygarl, and C. K. Chang, "Incremental Latent Semantic Indexing for Automatic Traceability Link Evolution Management," L'Aquila, Italy, 2008, pp. 59-68.

75. A. de Lucia, R. Oliveto, and G. Tortora, "ADAMS Re-Trace: Traceability Link Recovery via Latent Semantic Indexing," Leipzig, Germany, 2008, pp. 839-842.

76. H.-Y. Jiang, T. N. Nguyen, C. K. Chang, and F. Dong, "Traceability Link Evolution Management with Incremental Latent Semantic Indexing," Beijing, China, 2007, pp. 309-316.

77. M. Grechanik, K. S. McKinley, and D. E. Perry, "Recovering and Using Use-Case-Diagram-To-Source-Code Traceability Links," New York, NY 10036-5701, United States, 2007, pp. 95-104.

78. M. Lormans and A. Van Deursen, "Can LSI Help Reconstructing Requirements Traceability In Design And Test?," Bari, Italy, 2006, pp. 47-56.

79. A. Tang, Y. Jin, and J. Han, "A Rationale-Based Architecture Model for Design Traceability and Reasoning," Journal of Systems and Software, vol. 80, pp. 918-934, 2007.

80. T. K. Satyananda, D. Lee, S. Kang, and S. I. Hashmi, "Identifying Traceability Between Feature Model and Software Architecture in Software Product Line Using Formal Concept Analysis," Piscataway, NJ 08855-1331, United States, 2007, pp. 380-386.

81. J. Cleland-Huang, R. Settimi, C. Duan, and X. Zou, "Utilizing Supporting Evidence to Improve Dynamic Requirements Traceability," Paris, France, 2005, pp. 135-144.

82. P. Mäder, I. Philippow, and M. Riebisch, "Customizing Traceability Links for the Unified Process," Heidelberg, D-69121, Germany, 2007, pp. 53-71.

83. G. Spanoudakis, A. Zisman, M. rez, E. ana, and P. Krause, "Rule-Based Generation of Requirements Traceability Relations," Journal of Systems and Software, vol. 72, pp. 105-127, 2004.

84. E. Alexander, "Resolving Uncertainties During Trace Analysis," in Proceedings Of The 12th ACM SIGSOFT Twelfth International Symposium on Foundations of Software Engineering Newport Beach, CA, USA: ACM, 2004.

85. P. G. Alexander Egyed, "A Scenario-Driven Approach to Trace Dependency Analysis," IEEE Transactions On Software Engineering, vol. 29, pp. 116-132, 2003.

86. E. Alexander, B. Stefan, H. Matthias, and G. Paul, "Determining The Cost-Quality Trade-Off For Automated Software Traceability," in Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering Long Beach, CA, USA: ACM, 2005.

87. A. Egyed and P. Grünbacher, "Supporting Software Understanding with Automated Requirements Traceability," International Journal of Software Engineering and Knowledge Engineering, vol. 15, pp. 783-810, 2005.

88. H. Kagdi, J. I. Maletic, and B. Sharif, "Mining Software Repositories for Traceability Links," Piscataway, NJ 08855-1331, United States, 2007, pp. 145-154.

89. M. Eaddy, T. Zimmermann, K. D. Sherwood, V. Garg, G. C. Murphy, N. Nagappan, and A. V. Aho, "Do Crosscutting Concerns Cause Defects?," IEEE Transactions On Software Engineering, vol. 34, pp. 497-515, 2008.

90. J. M. Conejero, J. Hernandez, A. Moreira, and J. Araujo, "Discovering Volatile and Aspectual Requirements Using a Crosscutting Pattern," New Delhi, India, 2007, pp. 391-392.

91. S. Americo, G. Phil, F. G. Alessandro, and R. Awais, "A Comparative Study of Aspect-Oriented Requirements Engineering Approaches," in Proceedings of the First International Symposium on Empirical Software Engineering and Measurement: IEEE Computer Society, 2007.

92. M. Silvio, R. Tobias, S. Reinhard, and G. Martin, "Modeling and Evolving Crosscutting Concerns in ADORA," in Proceedings of the Early Aspects at ICSE: Workshops in Aspect-Oriented Requirements Engineering and Architecture Design: IEEE Computer Society, 2007.

93. R. Chitchyan, A. Rashid, and P. Sawyer, "Comparing Requirements Engineering Approaches for Handling Crosscutting Concerns," in 8th Workshop on Requirements Engineering, 2005.

94. M. C. José and H. Juan, "Analysis of Crosscutting Features in Software Product Lines," in Proceedings of the 13th international workshop on Software Architectures and Mobility Leipzig, Germany: ACM, 2008.

95. J. M. Conejero and J. Hernandez, "Analysis of Crosscutting Features in Software Product Lines," Leipzig, Germany, 2008, pp. 3-9.

96. K. v. d. Berg, J. M. Conejero, and H. Juan, "Analysis of Crosscutting Across Software Development Phases Based On Traceability," in Proceedings of the 2006 international workshop on Early aspects at ICSE Shanghai, China: ACM, 2006.

97. H. Sharif, S. Rehan, and G. A. Farrukh, "Aspect-Oriented Requirements Engineering: A Use Case Based Approach," Karachi, Pakistan, 2005.

98. L. Rosenhainer, "The DISCERN Method: Dealing Separately with Crosscutting Concerns," in Early Aspects 2005: Aspect-Oriented

Requirements Engineering and Architecture Design Workshop, San Diego, CA, USA, 2005.

99. L. Rosenhainer, "A Method for Handling Requirements-Level Crosscutting Concerns."

100. A. Moreira, J. Araujo, and J. Whittle, "Modeling Volatile Concerns as Aspects," Luxembourg, 2006, pp. 544-558.

101. C. Ruzanna, S. Américo, R. Awais, and R. Paul, "A Tool Suite for Aspect-Oriented Requirements Engineering," in Proceedings of the 2006 International Workshop on Early Aspects at ICSE Shanghai, China: ACM, 2006.

102. J. A. a. A. Moreira, "An Aspectual Use-Case Driven Approach," VIII Jornadas Ingeniería del Software y Bases deDatos (JISBD), pp. 463--468, 2003.

103. E. Navarro, P. Letelier, and I. Ramos, "Requirements and Scenarios: Running Aspect-Oriented Software Architectures," Mumbai, India, 2007.

104. I. Brito, "Aspect-Oriented Requirements Engineering," in 7th International Conference on Unified Modelling Language (UML) Lisbon, Portugal, 2004.

105. J. Araújo, A. Moreira, I. Brito, and A. Rashid, "Aspect-Oriented Requirements with UML," in Workshop on Aspect-Oriented Modelling with UML, 2002.

106. L. S. Liu Xiaomei, Zheng Xiaojuan, "Adapting the NFR Framework to Aspectual Use-Case Driven Approach," in 2009 Seventh ACIS International Conference on Software Engineering Research, Management and Applications, 2009, pp. 209-214.

107. L. X. Zheng Xiaojuan, Liu Shulin, "Use Case and Non-Functional Scenario Template-Based Approach to Identify Aspects," in 2010 Second International Conference on Computer Engineering and Applications, 2010.

108. S. Américo, C. Ruzanna, R. Awais, and R. Paul, "EA-Miner: A Tool for Automating Aspect-Oriented Requirements Identification," in Proceedings of the 20th IEEE/ACM international Conference on Automated Software Engineering Long Beach, CA, USA: ACM, 2005.

109. S. Pete, R. Paul, and G. Roger, "REVERE: Support for Requirements Synthesis from Documents," Information Systems Frontiers, vol. 4, pp. 343-353, 2002.

110. A. Rashid, P. Sawyer, A. Moreira, and J. Araújo, "Early Aspects: A Model for Aspect-Oriented Requirements Engineering," in International Conference on Requirements Engineering (RE) Essen, Germany: Society Press, 2002, pp. 199--202.

111. S. Katz and A. Rashid, "From Aspectual Requirements to Proof Obligations for Aspect-Oriented Systems," in 12th IEEE International Conference on Requirements Engineering (RE), Kyoto, Japan, 2004.

112. S. S. Geórgia Sousa, Paulo Borba and Jaelson Castro, "Separation of Crosscutting Concerns from Requirements to Design: Adapting an Use Case Driven Approach," in Early Aspects 2004:Aspect-Oriented Requirements Engineering and Architecture Design Workshop of the 3rd International Conference on Aspect-Oriented Software Development, , Lancaster, UK., 2004.

113. I. Jacobson, "Use Case and Aspects - Working Seemlessly Together," Journal of Object Technology, vol. 2 no. 4, pp. 7-28, July - August 2003.

114. M. Lázaro and E. Marcos, "Research in Software Engineering: Paradigms and Methods," in 17th International Conference on Advanced Information System (CAiSE'05), Porto, Portugal, June 2005.

115. A. F. George Spanoudakis, "Reconciling Requirements: A Method for Managing Interference, Inconsistency and Conflict," Annals of Software Engineering, Special Issue on Software Requirements Engineering, vol. 3, pp. 433--457, 1996.

116. B. Kitchenham, S. Linkman, and D. Law, "DESMET: A Methodology for Evaluating Software Engineering Methods and Tools," Computing & Control Engineering Journal, vol. 8, pp. 120-126, 1997.

117. B. Farbay and A. Finkelstein, "Evaluation in Software Engineering: ROI, but More than ROI," in 3rd International Workshop on Economics-Driven Software Engineering Research (EDSER-3) 2001.

118. W. Ossadnik and O. Lange, "AHP-based Evaluation of AHP-Software," European Journal of Operational Research, vol. 118, pp. 578-588, 1999.

119. N. M. Nayan, H. B. Zaman, and T. M. T. Sembuk, "Defining Information System Failure in Malaysia: Results from Delphi Technique," in 2010 International Symposium on Information Technology, 2010, pp. 1616-1621.

120. D. Morera, "COTS Evaluation Using DESMET Methodology & Analytic Hierarchy Process (AHP)," in International Conference on Product Focused Software Process Improvement, 2002, pp. 485-493.

121. M. Shahid, S. Ibrahim, and M. N. r. Mahrin, "An Evaluation of Requirements Management and Traceability Tools," World Academy of Science, Engineering and Technology, WASET, 2011.

122. G. Antoniol, G. La Gommare, G. Giraudo, and P. Tonella, "Effective Feature Analysis for Tool Selection," in VTT SYMPOSIUM, 1999, pp. 103-117.

123. H. Hedberg and J. Lappalainen, "A Preliminary Evaluation of Software Inspection Tools, with the DESMET Method," in Quality Software, 2005.(QSIC 2005). Fifth International Conference on, 2005, pp. 45-52.

124. S. L. B. Kitchenham, D. Law, "DESMET: A Method for Evaluating Software Engineering Methods and Tools." vol. 21(1): SIGSOFT Notes, 1996, pp. 11-14.

125. M. S. Tabares, A. Moreira, R. Anaya, F. Arango, and J. Araujo, "A Traceability Method for Crosscutting Concerns with Transformation Rules," Minneapolis, United states, 2007, p. IEEE Computer Society Technical Council on Software Engineering; ACM Special Interest Group on Software Engineering (SIGSOFT).

126. I. S. B. Elisabete Soeiro, Ana Moreira, "An XML-Based Language for Specification and Composition of Aspectual Concerns," in Proceedings of the 8th International Conference on Enterprise Information Systems (ICEIS), Paphos, Cyprus, 2006, pp. 410–419.

127. B. S. Ali and Z. M. Kasirun, "Crosscutting Concern Identification at Requirements Level," Malaysian Journal of Computer Science, vol. 21(2), pp. 78-87, 2008.

128. W. M. N. W. K. Noraini Ibrahim, Safaai Deris, "Propagating Requirement Change into Software High Level Designs towards Resilient Software Evolution," in 2009 16th Asia-Pacific Software Engineering Conference, Penang, 2009, pp. 347-354.

129. R. Sebesta, Concepts of Programming Language, 7 ed.: Pearson Education, 2006.

130. A. B. Marco Ajmone Marsan, Luigi Ciminiera, Riccardo Sisto, Adriano Valenzano, "A LOTOS Extension for the Performance Analysis of

Distributed Systems," *IEEE/ACM Transactions on Networking*, vol. 2, pp. 151 - 165, 1994.

131. I. f. S. I. S. (ISIS), "GME: Generic Modeling Environment." vol. 2008: Institute for Software Integrated Systems (ISIS), Schoole of Engineering, Vanderbilt University, 2008.

132. K. Barbara Ann, "Evaluating Software Engineering Methods and Tool Part 1: The Evaluation Context and Evaluation Methods," *SIGSOFT Softw. Eng. Notes*, vol. 21, pp. 11-14, 1996.

133. K. E. Kendall and J. E. Kendall, *Systems Analysis and Design*, 4th ed.: Prentice Hall, 1998.