Traceability Matrix for Effort Estimation in Change Request

Authors:
Mazidah Mat Rejab[1], Nurulhuda Firdaus Mohd Azmi[2],
Suriayati Chuprat[3], Hairulnizam Mahdin[4]
Email:
mazidah@uthm.edu.my[1], huda@utm.my[2], suriayati.kl@utm.my[3],
hairuln@uthm.edu.my[4]

Abstract: In the last decade, the management of IT projects has become a challenging task. The latest published figures on the status of IT projects indicate a large failure rate, which has created a crucial challenge for project managers. In maintenance phase, the impact of changes is an important aspect due to the evolving environment of the software development life cycle. The aim of this book is to investigate the need and significant use of the traceability matrix for effort estimation to accommodate changes request in maintenance tasks. Change request is necessary to keep task current and reusable. Software evolves over time due to specific changes during development and maintenance at every point, the management aspect of modifications can become more complicated and potentially risky. Many of the current traceability approaches and tools are devoted to and restricted to high-level objects such as specifications and fewer capabilities made available to handle lower-level artefacts such as classes and codes While effort estimates have been in place for decades, it remains a major challenge for project management to make accurate estimates and, ultimately, to successfully complete the IT project.

Keywords: Software, changes, maintenance, SLOC

Traceability Matrix

for Effort Estimation in Change Request

MAZIDAH MAT REJAB
NURULHUDA FIRDAUS MOHD AZMI
SURIAYATI CHUPRAT
HAIRULNIZAM MAHDIN

Penerbit UTHM

Traceability Matrix

for Effort Estimation in Change Request

Traceability Matrix

for Effort Estimation in Change Request

MAZIDAH MAT REJAB
NURULHUDA FIRDAUS MOHD AZMI
SURIAYATI CHUPRAT
HAIRULNIZAM MAHDIN

Penerbit UTHM
2023

Published and printed by:
Penerbit UTHM
Universiti Tun Hussein Onn Malaysia
86400 Parit Raja,
Batu Pahat, Johor
Tel: 07-453 7051 / 7454
Fax: 07-453 6145

Website: http://penerbit.uthm.edu.my
E-mail: pt@uthm.edu.my
http://e-bookstore.uthm.edu.my

Penerbit UTHM is a member of
Majlis Penerbitan Ilmiah Malaysia (MAPIM)

Cataloguing-in-Publication Data

Perpustakaan Negara Malaysia

A catalogue record for this book is available from the National Library of Malaysia

ISBN 978-967-0061-62-7

Dedication

My special gratitude to my beloved family for their continuous support.

"To Dr. NurulHuda Firdaus and AP Dr. Suriayati Chuprat, thank you for your guidance, advice, and motivations".

"For my lovely husband Muhammad Razin Bin Marzuki and my son Umar Waldan, Izz Fateh, Fahim Fitri thanks you for the undivided support and encouragement for me to finish this book "

"Thank you for all".

Table of Contents

DEDICATION		v
LIST OF TABLES		xi
LIST OF FIGURES		xiii
LIST OF ABBREVIATIONS		xv
ACKNOWLEDGEMENT		xvii
PREFACE		xix
CHAPTER 1	**CHALLENGE SOFTWARE CHANGES IN SOFTWARE MAINTENANCE**	
1.1	Overview	1
1.2	Criticality Software Change in Maintenance Process	5
CHAPTER 2	**SOFTWARE LIFE CYCLE OF SOFTWARE MAINTENANCE**	
2.1	Overview	7
2.2	Software Changes	9
2.3	The Connection Between Software Change And Maintenance	10
2.4	Software Changes Problems and Challenges	11
2.5	Change Management Process	14
2.6	Change Request Specification	15
2.7	Bohner's Change Management Model	18
2.8	Small & Downey Change Model	19
2.9	Sommerville Change Model	20

2.10	Olsen's Change Management Model	21
2.11	Ince's Change Process Model	22
2.12	Software Testing	23
2.13	Maintenance Testing	25
2.14	Regression Testing	26

CHAPTER 3 SOFTWARE TRACEABILITY

3.1	Overview	31
3.2	Traceability Matrix	33
3.3	Software Traceability Approaches	33
3.4	Critical Analysis of Traceability Approach	37
3.5	Impact Analysis	38
3.6	Fundamental Issues of Impact Analysis	39
3.7	Impact Analysis Process	39
3.8	Impact Analysis Techniques	38
3.9	Change Impact Analysis	48
3.10	Integrated Change Impact Analysis	51

CHAPTER 4 TEST EFFORT ESTIMATION

4.1	Overview	55
4.2	Source Lines of Code (SLOC)	59
4.3	The Role Of Estimation in Software Project	61
4.4	Test Coverage	63
4.5	Test Coverage Analysis	63
4.6	Test Coverage Items	64
4.7	Requirement Coverage	66
4.8	Test Case Coverage	67
4.9	Hybrid coverage	67

4.10	Evaluation Criteria of Test Coverage Approaches	68
4.11	Comparative Evaluation of Test Coverage Approaches	72
4.12	Existing Work on Software Traceability Analysis of Test Coverage	75
4.13	Comprative studies in Integrating Traceability Model for Effort Estimation	84

CHAPTER 5 CONCLUSION

5.1	Overview	87

REFERENCES 89

BIOGRAPHY 115

List of Tables

Table 2.1	Change Request Specification Attributes	10
Table 3.1	Analysis of Traceability Approaches	37
Table 3.5	Integrated Techniques of Change Impact Analysis	53
Table 4.1	Effort Estimation Techniques Categories	57
Table 4.2	SLOC Categories (Hihn and Monson, 2011)	60
Table 4.3	Evaluation of Test Coverage Approaches	73
Table 4.4	Evaluation Of Traceability Model Integrating With Effort Estimation	85

List of Figures

Figure 2.1	A Change Request Scenario	10
Figure 2.2	Failed Project from 2011-2015 (CHAOS Report, 2015)	11
Figure 2.3	Change Request Form (Sommerville, 2007)	18
Figure 2.4	Change Management Process (Bohner, 1996)	19
Figure 2.5	Change Process Model (Small & Downey, 2001)	20
Figure 2.6	Change Process Model (Sommerville, 2004)	21
Figure 2.7	Change Management Model (Olsen, 1993)	22
Figure 2.8	Change Model (Ince, 1995)	23
Figure 3.1	Structure of the Impact Analysis Process	40
Figure 3.2	Directed Graph Example	43
Figure 3.3	String of Letters Example	44
Figure 3.4	Whole Path DAG Example (Law and Rothermel, 2003)	45
Figure 3.5	Influences and Influence Graph Example (Breech et al., 2006)	46
Figure 4.1	Logical SLOC in C++ Code Sample	61
Figure 4.2	OCCF Framework for Test Coverage (Sakamoto, 2010)	77

Figure 4.3	Test Coverage Approach (Kapfhammer, 2008)	79
Figure 4.4	Test Coverage Approach (Lingempally, 2007)	80
Figure 4.5	A View of the CATIA System (Ibrahim, 2006)	81
Figure 4.6	System Architecture of HYCAT	82
Figure 4.7	Change Effort Prediction Model (CEPM)	83

List of Abbreviations

AIS	Actual Impact Set
CCB	Change Control Board
CFG	Control Flow Graph
CIS	Candidate Impact Set
CSC	Computer Software Components
CSCI	Computer Software Configuration Item
GUI	Graphical User Interface
IREQ	Inter-Requirement Traceability
JRE	Java Runtime Environment
LOC	Lines of Code
OBA	On-Board Automobile
OOP	Object Oriented Programming
PIS	Primary Impact Set
PCR	Program Change Request
RTOM	Requirement to Object Model
RTS	Regression Test Selection
SIS	Secondary Impact Set
SCM	Software Configuration Management
SDD	Software Design Description
SDLC	Software Development Lifecycle
SRS	Software Requirement Specification
STD	Software Test Description
SUT	Software Under Test
UML	Unified Modelling Language
XML	Extensible Markup Language

Acknowledgement

In the name of Allah, The Most Gracious, The Most Merciful, first and foremost, all the praises to Allah who created us and gave us intelligence and guidance. Moreover, peace be upon our prophet, the teacher of all humankind, and peace be upon his family.

First of all, I would like to thank Allah for giving me a chance to complete this book. To my father, Mat Rejab, and my mom Zaitun and family, there is no word to illustrate how much I'm thankful to have them in my life. I can only pray for their well-being as a token of appreciation for them. I did not forget my beloved husband, Muhammad Razin Marzuki. He has given the spirit and encouragement to me, also my son Umar Waldan, Izz Fateh, and Fahim Fitri for giving me strength in completing this book.

Millions of thanks and appreciation to Dr. Nurulhuda Firdaus and Ass. Prof. Dr. Suriayati Chuprat, for guiding.

Last but not least, FSKTM Staff, UTHM for all the support and assistance that have been extended. To all involved, once again, I would like to express my deepest thanks. May Allah bless our efforts in this world and in the hereafter.

Preface

Software engineering is, a critically important technology for the future of mankind. We must continue to educate software engineers and develop the discipline so that we can create more complex software systems. Of course, there are still problems with software projects. Software is still sometimes late and costs more than expected. However, we should not let these problems conceal the real successes in software engineering and the impressive software engineering methods and technologies that have been developed. Software engineering is now such a huge area that it is impossible to cover the whole subject in one book. My focus, therefore, is on key topics that are fundamental to all software maintenance processes. We need to combine the best of these approaches to build better software systems. Books inevitably reflect the opinions and prejudices of their authors. Some readers will inevitably disagree with my opinions and choice of material. Such disagreement is a healthy reflection of the diversity of the discipline and is essential for its evolution. Nevertheless, I hope that all software engineers and software engineering students can find something of interest here.

This book addresses software testing and effort estimation in the context of an overall effort to achieve quality. It is designed for use as a primary textbook for a course in software testing or as a supplementary text in a software engineering course, and as a resource for software developers. The main characteristics of this book are: • It assumes that the reader's goal is to achieve a suitable balance of cost, schedule, and quality. It is not oriented toward critical systems for which ultra-high reliability must be obtained regardless of cost, nor will it be helpful if one's aim is to cut cost

or schedule regardless of consequence. • It presents a selection of techniques suitable for near-term application, with sufficient technical background to understand their domain of applicability and to consider variations to suit technical and organizational constraints. Techniques of only historical interest and techniques that are unlikely to be practical in the near future are omitted. • It promotes a vision of software testing and analysis as integral to modern software engineering practice, equally as important and technically demanding as other aspects of development. This vision is generally consistent with current thinking on the subject, and is approached by some leading organizations, but is not universal. • It treats software testing and static analysis techniques together in a coherent framework, as complementary approaches for achieving adequate quality at acceptable cost. I hope this book can help student, industry worker and lecture in software engineering field.

Ts. Dr. Mazidah Binti Mat Rejab
Faculty of Computer Science & Information Technology,
Universiti Tun Hussein Onn Malaysia.

Chapter 1
CHALLENGE SOFTWARE CHANGES IN SOFTWARE MAINTENANCE

1.1 Overview

Software maintenance generally involves changing the software product following delivery to end-users. It is a broader task involving correcting errors, improving performance, improving the functionality, and removing obsolete functionality. It is, therefore, not limited to programming only but also refers to other parts of the lifecycle of software such as software specifications, testing, and development. Model Rajlich and Bennett (2001) differentiate between the phases of development, evolution, and maintenance. The maintenance process comes to an end when the main system development finishes. The development phase will last until the system enters operation. The evolution process is between, where the system is still in development but still increasing. Developing software is necessary to deal with emerging changes.

Many tasks need to be handled during the maintenance phase. Such practices maintain traceability, change management, and the maintenance of artefacts. Before any changes are implemented, they need to be known before and after changes occur. Mäder et al., 2012 say modifications made throughout the software development process at the test level or at any other stage have often occurred without updating the relevant documentation. Functionality changes may occur due to internal or external requirements, i.e., user requests. When a software project complete, a user changes requirements will arise, such as frameworks, documents, or as the software development process progresses, including configuration, design, and testing. Software changes or software evolution are,

Chapter 2
SOFTWARE LIFE CYCLE OF SOFTWARE MAINTENANCE

2.1 Overview

Software maintenance is an indispensable part of a software life cycle. According to IEEE Inc. (1991), "it is the process of modifying the software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a change in an environment". The objective of software maintenance is to modify existing software while preserving its integrity (IEEE SWEBOK, 2012).

There are four (4) software maintenance types that IEEE Inc., 1998 classify as follows:

a) Corrective maintenance - this category includes repair structure and programming defects discovered after delivery. This category also includes emergency repair, which is an unscheduled alteration operation.

b) Adaptive Maintenance-it refers to the alteration of a software system to environmental changes without first altering the functionality of the program. Operating systems, for example, maybe upgraded.

c) Good maintenance-Improvements are made in this maintenance category to add new functionality to improve performance, maintainability, documentation of the system, or other software attributes.

d) Preventive maintenance -it is performed after distribution to adjust a software product to avoid the faults before they occur.

Chapter 3
SOFTWARE TRACEABILITY

3.1 Overview

Software traceability is useful to detect the affected software artefact during change. Software change not only affects source code but also other artefacts like design, test artefacts, and requirements. Requirements traceability is one of the most important and challenging tasks in the software industry to ensure that a software product meets all user requirements. It is linking requirements to software artefacts, such as source code, test cases, and design documents.

Requirement traceability offers support for numerous software engineering tasks like requirement verification and validation, coverage analysis, impact analysis, and regression testing. In addition, requirement traceability is the recognized component of several software process improvement campaigns. Since the software evolution is inevitable, any traceability approach must consider signification affecting aspects so that it can reduce the evolution attempt (Rochimah, S. et al., 2007).

Requirement traceability is concerned with the documenting of the life of a requirement and extending di-directional traceability between other associated requirements. It enables individuals to capture the origin of each requirement and inherit any change that was made to this requirement with the passage of time.

Abran A. et al. (2001) describes requirement tracing as being concerned with recovering the source of requirement and predicting the effect of requirement. Ramesh (2001) relates

Chapter 4
TEST EFFORT ESTIMATION

4.1 Overview

Estimation is governed by the intelligent belief of the quantity of area to be performed and the specific material (i.e., human resources, financial resources, material resources, and time resources) needed to conduct the finding in a given environment with specified methods at a future date. Test Effort Calculation is the estimate of the test duration, effort, expense, and schedule for a particular software test project in an individual setting with various methods, tools, and techniques (A. Bertilino 2007). The test effort is to determine the effort spent on test operation and the effort spent on debug operation (E. Aranha et al. 2007).

For the evaluation of test effort, the earlier part of approaches was created, but the majority of the estimate of effort considers the basis for classification of use case, code, and requirement. A. Bertilino (2007) presents a similar machine check effort estimation work with drawbacks. A. Bertilino (2007) suggested a roadmap evaluating applications to clarify the software development benefits, challenges, and goals. E. Aranha (2007) considered a method to use managed natural language (CNL) tool to translate test specifications into natural language and quantify effort. This instrument also computes the calculation of test quantity and the complexity of test execution. Z. Xiochun (2008) suggested an approach to estimate test suit sizes. The number of test cases, the complexity of execution, and the tester are defined as a 3-dimensional vector for finding a test fit. Aranha & Borba (2007) addressed a model for test selection on test execution and test automation effort estimation.

Chapter 5
CONCLUSION

5.1 Overview

In this book had discussed an overview of software change and issues and challenges related to a software change. Next, further reviewed on software testing, regression testing, software traceability, test coverage, and large software system development. This chapter also consists discussion on the effort the estimation and do some evaluation and comparative about the traceability approach from another informtaion that come out with the proposed model. This book explained an overview of the software change and its current issues, software maintenance, change management process, traceability, test coverage, impact analysis, and regression testing. The later part describes the methods, steps, algorithms, and formal models used in a few state-of-art approaches and tools related to test coverage measurement and analysis. Then presented a comparative analysis for all the approaches and tools related to test coverage that was introduced in the chapter. Some potential information problems can be summarised as follows:

Software maintenance and software change management have some common issues like traceability problems, requirements change reasons, and higher cost and tools related issues. All these concerns are related to changes in software artefacts. Most of the information deal with change before its real implementation. As of now, there is no such support for change management of artefacts that can address both before and after change issues.

Coverage measurement output is utilized in many manners for improvement of the verification process and software quality.

REFERENCES

Abran, A., Bourque, P., Dupuis, R., & Moore, J. W. (2001). *Guide to the software engineering body of knowledge-SWEBOK*. (3rd ed.), IEEE Press.

Adler, M., & Ziglio, E. (1996). *Gazing into the oracle: The Delphi method and its application to social policy and public health*. Jessica Kingsley Publishers.

Adler, Y., Behar, N., Raz, O., Shehory, O., Steindler, N., Ur, S., & Zlotnick, A. (2011, May). Code coverage analysis in practice for large systems. In *Proceedings of the 33rd International Conference on Software Engineering* (pp. 736-745). ACM.

Agarwal, N., & Rathod, U. (2006). Defining 'success' for software projects: An exploratory revelation. *International journal of project management*, *24*(4), 358-370.

Alenljung, B., & Persson, A. (2006). Decision-making activities in requirements engineering decision processes: a case study. In *Advances in Information Systems Development* (pp. 707-718). Springer, Boston, MA.

Ali, H. O., Rozan, M. Z. A., and Sharif, A. M. (2012), Identifying challenges of impact analysis for software projects. *Innovation Management and Technology Research (ICIMTR), 2012 International Conference on*. 21-22 May 2012, 407-411.

Allen, J., Dyas, J., & Jones, M. (2004). Building consensus in health care: a guide to using the nominal group technique. *British journal of community nursing*, *9*(3), 110-114.

Antoniol, G., Canfora, G., Casazza, G., De Lucia, A., & Merlo, E. (2002). Recovering traceability links between code and documentation. *Software Engineering, IEEE Transactions on, 28*(10), 970-983.

Apiwattanapong, T., Orso, A., & Harrold, M. J. (2004, September). A differencing algorithm for object-oriented programs. In *Proceedings of the 19th IEEE international conference on Automated software engineering* (pp. 2-13). IEEE Computer Society.

Aranha, E., & Borba, P. (2007, September). An estimation model for test execution effort. In *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)* (pp. 107-116). IEEE.

Ariavie, G.O. & Ovuwo, G.C. (2012). Delphi Fuzzy Elicitation Technique in the Determination of Third Party Failure Probability of Onshore Transmission Pipeline in the Niger Delta region of Nigeria. J. Appl. Sci. Environ. Manage,16(1), 95 - 101.

Asl, M. H. (2013). *A Change Effort Estimation Model Using Impact Analysis for the Software Development Phase* (Doctoral dissertation, Universiti Teknologi Malaysia).

Asl, M. H., and Kama, N. (2013, 4-7 June 2013). *A Change Impact Size Estimation Approach during the Software Development.* Paper presented at the Software Engineering Conference (ASWEC), 2013 22nd Australian, 68-77.

Atkins DL, Ball T, Graves TL, Mockus A (2002) Using version control data to evaluate the impact of software tools: A case study of the version editor. IEEE Trans Softw Eng 28(7):625–637.

Attarzadeh, I., Mehranzadeh, A., and Barati, A. (2012, 24-26 July 2012). *Proposing an Enhanced Artificial Neural Network Prediction Model to Improve the Accuracy in Software Effort Estimation.* Paper presented at the Computational Intelligence, Communication Systems and Networks (CICSyN), 2012 Fourth International Conference on, 167-172.

Auer M, Trendowicz A, Graser B, Haunschmid E, Biffl S (2006) Optimal project feature weights in analogy based cost estimation: Improvement and limitations. IEEE Trans Softw Eng 32:83–92. doi:10.1109/ TSE.2006.1599418

Aurum, A., & Wohlin, C. (2003). The fundamental nature of requirements engineering activities as a decision-making process. *Information and Software Technology*, *45*(14), 945-954.

Ball, T. (1998). On the limit of control flow analysis for regression test selection. *ACM SIGSOFT Software Engineering Notes*, *23*(2), 134-142.

BASRI, M. S. B. (2016). *An algorithmic-based software change effort prediction model using change impact analysis for software development* (Doctoral dissertation, Universiti Teknologi Malaysia).

Bee Bee, C. (2010, 22-27 Aug. 2010). *Rework Requirement Changes in Software Maintenance.* Paper presented at the Software Engineering Advances (ICSEA), 2010 Fifth International Conference on, 252-258.

Beizer, B. (1990). *Software Testing Techniques.* 2nd edition, New York: Van Nostrand Reinhold.

Bennett, K. H., & Rajlich, V. T. (2000, May). Software maintenance and evolution: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering* (pp. 73-87). ACM.

Bennett, K., Rajlich, V.: "Software Main tenance and Evolution – A Staged Model" Proc. of the Future of Software Eng., ICSE-2000, IEEE Press, Limerick, 2001, p. 73- 89

Berliner, D. C. (2004a). Expert teachers: Their characteristics, development and accomplishments. *Bulletin of Science, Technology and Society*, *24*(3), 200-12.

Berliner, D. C. (2004b). Describing the behavior and documenting the accomplishments of expert teachers. *Bulletin of Science, Technology & Society*, *24*(3), 200-212.

Bertolino, A. (2007, May). Software testing research: Achievements, challenges, dreams. In *2007 Future of Software Engineering* (pp. 85-103). IEEE Computer Society.

Bissi, W., Neto, A. G. S. S., & Emer, M. C. F. P. (2016). The effects of test driven development on internal quality, external quality and productivity: A systematic review. *Information and Software Technology*, *74*, 45-54

Bodjanova, S. (2006). Median alpha-levels of a fuzzy numbe. Fuzzy Sets and Systems,157(7),879 - 891.

Boehm, B. W. (1987). Improving software productivity. *Computer*, (9), 43-57.

Boehm, Beck. (2010). Perspectives the changing nature of software evolution; The inevitability of evolution. *Software, IEEE*, *27*(4), 26-29.

Bohner, S. A. (1996, November). Impact analysis in the software change process: A year 2000 perspective. In *Software Maintenance 1996, Proceedings., International Conference on* (pp. 42-51). IEEE.

Bohner, S. A. (2002). Software change impacts-an evolving perspective. In *Software Maintenance, 2002. Proceedings. International Conference on* (pp. 263-272). IEEE.

Bohner, S. A. (2002, December). Extending software change impact analysis into cots components. In *Software Engineering Workshop, 2002. Proceedings. 27th Annual NASA Goddard/IEEE* (pp. 175-182). IEEE.

Bojadziev, G., & Bojadziev, M. (2007). Fuzzy Set For Business, Finance and Management. Singapore: World Scientific Publishing Co. Pte. Ltd.

Bourque, P., Dupuis, R., Abran, A., Moore, J. W., & Tripp, L. (2004). Guide to the software engineering body of knowledge.

Breech, B., Tegtmeyer, M., & Pollock, L. (2006, September). Integrating influence mechanisms into impact analysis for increased precision. In *Software Maintenance, 2006. ICSM'06. 22nd IEEE International Conference on* (pp. 55-65). IEEE.

Buckley, J., Mens, T., Zenger, M., Rashid, A., & Kniesel, G. (2005). Towards a taxonomy of software change. *Journal of Software Maintenance and Evolution: Research and Practice*, *17*(5), 309-332.

Cai, H., Santelices, R., and Xu, T. (2014, June 30 2014-July 2 2014). *Estimating the Accuracy of Dynamic Change-Impact Analysis Using Sensitivity Analysis*. Paper presented at the Software Security and Reliability (SERE), 2014 Eighth International Conference on, 48-57.

Carlos, T. Article, Requirements Traceability Matrix - RTM. 21 October 2008, http:// www. pmhut. com

Chang, P. L., Hsu, C. W., & Chang, P. C. (2011). Fuzzy Delphi method for evaluating hydrogen production technologies. *International Journal of Hydrogen Energy*, *36*(21), 14172-14179.

Chang, P. T., Huang, L. C., & Lin, H. J. (2000). The fuzzy Delphi method via fuzzy statistics and membership function fitting and an application to the human resources. *Fuzzy sets and systems*, *112*(3), 511-520.

Chen, C.-Y., and Chen, P.-C. (2009). A holistic approach to managing software change impact. *Journal of Systems and Software*, *82*(12), 2051-2067.

Cheng, C. H., & Lin, Y. (2002). Evaluating the best main battle tank using fuzzy decision theory with linguistic criteria evaluation. *European journal of operational research*, *142*(1), 174-186.

Chittimalli, P. K., & Harrold, M. J. (2008, February). Regression test selection on system requirements. In *Proceedings of the 1st India software engineering conference* (pp. 87-96). ACM.

Chua, B. (2010). Rework Requirement Changes in Software Maintenance. *Software Engineering Advances (ICSEA), 2010 Fifth International Conference on*. 22- 27 Aug. 2010 252-258.

Chua, B., Bernardo, D., and Verner, J. (2008). *Criteria for Estimating Effort for Requirements Changes.* In R. O'Connor, N. Baddoo, K. Smolander & R. Messnarz (Eds.), *Software Process Improvement* (Vol. 16, pp. 36-46): Springer Berlin Heidelberg.

Cleland-Huang, J., Chang, C. K., & Christensen, M. (2003). Event-based traceability for managing evolutionary change. *IEEE Transactions on Software Engineering*, *29*(9), 796-810

Cleland-Huang, J., Gotel, O. C., Huffman Hayes, J., MÃd'der, P. and Zisman, A. (2014). Software traceability: trends and future directions. In *Proceedings of the on Future of Software Engineering*. ACM, 55–69.

Clover (2013), http://www.atlassian.com/software/clover/overview, Last accessed on September 29, 2013

Cobertura (2013), http://cobertura.github.io/cobertura/, accessed on Sep. 29, 2013

Dang, V. H. (2015). The Use of Nominal Group Technique: Case Study in Vietnam. *World Journal of Education*, *5*(4), 14-25.

Deslandes, S. F., Mendes, C. H. F., Pires, T. D. O., & Campos, D. D. S. (2010). Use of the Nominal Group Technique and the Delphi Method to draw up evaluation indicators for strategies to deal with violence against children and adolescents in Brazil. *Revista Brasileira de Saúde Materno Infantil*, *10*, s29-s37.

Dobbie, A., Rhodes, M., Tysinger, J. W., & Freeman, J. (2004). Using a modified nominal group technique as a curriculum evaluation tool. *FAMILY MEDICINE-KANSAS CITY-*, *36*, 402-406.

Downing, D. (2006). Managing Quality in your ERP Project: 12 Mistakes to Avoid & Best Practices to Adopt. *White Paper, Mentora Group [Online]*.

Drew Procaccino, J., Verner, J. M., Overmyer, S. P., and Darter, M. E. (2002). Case study: factors for early prediction of software development success. *Information and software technology*, *44*(1), 53-62.

Duan, C., & Cleland-Huang, J. (2006, September). Visualization and analysis in automated trace retrieval. In *Requirements Engineering Visualization, 2006. REV'06. First International Workshop on* (pp. 5-5). IEEE.

E. Aranha, F. D. Almeida, T. Diniz, V. Fonte and P. Borba, (2007) "Automated Test Execution Effort Estimation Based On Functional Test Specification", Proceedings of Testing, Academic and Industrial Conference Practice and Research Techniques, pp 1-4,

Eclemma, http://www.eclemma.org/, Last accessed on September 30, 2013

Egyed, A. (2001, July). A scenario-driven approach to traceability. In *Proceedings of the 23rd international conference on software engineering* (pp. 123-132). IEEE Computer Society.

Elbaum, S., Malishevsky, A., & Rothermel, G. (2001, May). Incorporating varying test costs and fault severities into test case prioritization. In *Proceedings of the 23rd International Conference on Software Engineering. ICSE 2001* (pp. 329-338). IEEE.

Eleiche, A. M., Ahmad, I., & Elish, M. O. (2011). Design Requirements in Software and Engineering Systems. In *Proceedings of 12th Asia Pacific Industrial Engineering & Management Systems Conference (APIEMS 2011)*

Faizah, Ibrahim, S., (2009), A Software Traceability Approach to Support Test Coverage Analysis, In the proceedings of *Third International Conference on Advances in Human-oriented and Personalized Mechanisms*, Technologies, and Services CENTRIC, China.

Faizah, Ibrahim, S., (2013), A Software Traceability Approach to Support Requirement Based Test Coverage Analysis, Doctor Philosophy, Universiti Teknologi Malaysia, Kuala Lumpur.

Finkelsteiin, A., & Kramer, J. (2000, May). Software engineering: a roadmap. In *Proceedings of the Conference on the Future of Software Engineering* (pp. 3-22).

Fisher II, M., Wloka, J., Tip, F., Ryder, B. G., & Luchansky, A. (2011, September). An evaluation of change-based coverage criteria. In *Proceedings of the 10th ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools* (pp. 21-28). ACM.

Freese, T. (2003). Towards Software Configuration Management for Test-Driven Development. In *Software Configuration Management*. (pp. 143–150).

Galorath, D. D., & Evans, M. W. (2006). *Software sizing, estimation, and risk management: when performance is measured performance improves*. CRC Press

Garcia, C. A. L., and Hirata, C. M. (2008). *Integrating functional metrics, COCOMO II and earned value analysis for software projects using PMBoK*. Paper presented at the Proceedings of the 2008 ACM symposium on Applied computing.

Garousi, V., & Elberzhager, F. (2017). Test automation: not just for test execution. *IEEE Software*, *34*(2), 90-96.

Garousi, V., Felderer, M., & Hacaloğlu, T. (2017). Software test maturity assessment and test process improvement: A multivocal literature review. *Information and Software Technology*, *85*, 16-42.

Ghosh, S. M., Sharma, H. R., & Mohabay, V. (2011). Analysis and modeling of change management process model. *International Journal of Software Engineering and Its Applications*, *5*(2).

Graham, D. (2002). Requirements and testing: Seven missing-link myths. *IEEE Software*, *19*(5), 15-17.

Graham, D., Van Veenendaal, E., Evans, I. and Black, R. (2008). *Foundations of software testing: ISTQB certification*. Course Technology Cengage Learning.

Grechanik, M., McKinley, K. S., & Perry, D. E. (2007, September). Recovering and using use-case-diagram-to-source-code traceability links. In *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering* (pp. 95-104). ACM.

Grimstad, S., and Jørgensen, M. (2006). A framework for the analysis of software cost estimation accuracy. *Proceedings of the 2006 ACM/IEEE international symposium on Empirical software engineering*. Rio de Janeiro, Brazil: 58-65.

Harrold, M. J. (2009). Reduce, reuse, recycle, recover: Techniques for improved regression testing. In *2009 IEEE International Conference on Software Maintenance*. September. Edmonton, AB, Canada, 5–5. doi:10.1109/ICSM.2009. 5306347.

Harvey, N., & Holmes, C. A. (2012). Nominal group technique: an effective method for obtaining group consensus. *International journal of nursing practice*, *18*(2), 188-194.

Hassine, J. (2015). Early modeling and validation of timed system requirements using Timed Use Case Maps. *Requirements Engineering*, *20*(2), 181-211.

Hassine, J., Rilling, J., Hewitt, J., and Dssouli, R. (2005, 5-6 Sept. 2005). *Change impact analysis for requirement evolution using use case maps*. Paper presented at the Principles of Software Evolution, Eighth International Workshop on, 81-90.

Hayhurst, K.J., Veerhusen, D.S., Chilenski, J.J., Rierson, L.K. (2001). *A Practical Tutorial on Modified Condition/Decision Coverage*. Langley Research Center, National Aeronautics and Space Administration (NASA).

Heindl, M. and S. Bim. (2005), A Case Study on Value-Based Requirement Tracing. In *International Conference on Empirical Software Engineering (ESEC- FSE'05)*. ACM Press.

Hihn, J., and Monson, E. (2011). Sizing the system : JPL software estimation class for NASA. *Pasadena, CA : Jet Propulsion*

Laboratory, National Aeronautics and Space Administration, 2011.

Ho, Y. F., & Chen, H. L. (2007). Healthy housing rating system. *Journal of Architecture, 60,* 115-136.

Hoffmann, M., Kuhn, N., Weber, M., & Bittner, M. (2004, September). Requirements for requirements management tools. In *Requirements Engineering Conference, 2004. Proceedings. 12th IEEE International* (pp. 301-308). IEEE.

Holtzapple, M. T. dan Reece, WD (2010). *Asas Kejuruteraan. Kuala Lumpur: Institut Terjemahan Negara Malaysia Berhad.*

Hsu, C.C. & Sandford, B.A. (2007). The Delphi Technique: Making Sense of Consensus. Practical Assessment, Research & Evaluation, 12(10). Hlm. 1-8.

Hsu, Y.L., Lee, C.H. & Kreng,V.B. (2010). The application of Fuzzy Delphi Method and Fuzzy AHP in lubricant regenerative technology selection. Expert Systems with Applications, 37, 419-425

Huang, L., & Song, Y. T. (2006, August). Dynamic impact analysis using execution profile tracing. In *Software Engineering Research, Management and Applications, 2006. Fourth International Conference on* (pp. 237-244). IEEE.

Huang, L., & Song, Y. T. (2007, August). Precise dynamic impact analysis with dependency analysis for object-oriented programs. In *Software Engineering Research, Management & Applications, 2007. SERA 2007. 5th ACISInternational Conference on* (pp. 374-384). IEEE.

Ibrahim, R. (2008). Setting up a research question for determining the research methodology. *International Journal on Sustainable Tropical Design Research and Practice, 3*(1), 99-102.

Ibrahim, R. (2011). Demystifying the Arduous Doctoral Journey: The Eagle Vision of a Research Proposal. *Electronic Journal of Business Research Methods, 9*(2).

Ibrahim, S. (2006). A document-based software traceability to support change impact analysis of object-oriented

software (Doctoral dissertation, Universiti Teknologi Malaysia, Faculty of Computer Science and Information System).

Ibrahim, S., Idris, N. B., Munro. M. and Deraman, A. (2005). Implementing a Document-based Requirements Traceability: A Case Study. IASTED International Conference on Software Engineering. 15-17 February 2005. Innsbruck, Austria: Octa Press, 124-131.

IEEE Inc. (1991). *IEEE Standard Glossary of Software Engineering Terminology.*USA, IEEE Std 610.12-1990.

IEEE Inc. (1998). *IEEE Standard for Developing Software Life Cycle Processes.*USA, IEEE Std 1074-1997.

IEEE Inc. (2012), SWEBOK Guide V3 – Alpha Version

IEEE-Software Engineering Standards Committee. (2012). IEEE Standard for Configuration Management in Systems and Software Engineering. *IEEE Std*, 828-2012.

IEEE-Std 1042-1998. IEEE Recommended Practice for Software Configuration Management.

Ince, D. C. (1995). *Introduction to software quality assurance and its implementation.* McGraw-Hill, Inc. ISBN 0-07-707924-8.

ISO 10007-1995, 1995. Quality Management, Guidance for Configuration Management.

Jafari, A., Jafarian, M., Zareei, A., & Zaerpour, F. (2008). Using fuzzy Delphi method in maintenance strategy selection problem. *Journal of Uncertain Systems*,

Jamil, M. R. M., Siraj, S., & Hussin, Z. Nurulrabihah Mat Noh., & Ahmad Arifin Sapar.(2017). *Pengenalan Asas Kaedah Fuzzy Delphi Dalam Penyelidikan Rekabentuk dan Pembangunan. (Mohd Ridhuan Mohd Jamil, Ed.). Kuala Lumpur, Malaysia: Minda Intelek Agency.*

Jashki, M. A., Zafarani, R., & Bagheri, E. (2008, November). Towards a more efficient static software change impact analysis method. In *Proceedings of the 8th ACM SIGPLAN-*

SIGSOFT workshop on Program analysis for software tools and engineering (pp. 84-90). ACM.

Jeng, D. J. F., & Tzeng, G. H. (2012). Social influence on the use of clinical decision support systems: revisiting the unified theory of acceptance and use of technology by the fuzzy DEMATEL technique. *Computers & Industrial Engineering*, *62*(3), 819-828.

JFeature(2013),http://www.technobuff.net/webapp/product/showProduct.do?name=jfeature

Johnson, Karen N. (2007) searchsoftwarequality.techtarget.com, February 13, 2007

Jonsson, R, 2007, Exploring Approach Aspects of Change Impact Analysis, Ph.D Thesis, Blekinge Institute of Technology Sweden. Jones, C. (2007). *Estimating Software Costs: Bringing Realism to Estimating* (2nd ed.). New York: NY: McGraw-Hill.

Jonsson, R., Lefteris, A., and Claes, W, 2008, An Empirical Study on Views of Importance of Change Impact Analysis Issues, IEEE Transactions On Software Engineering 34, 15, 4,

Jorgensen, M. (2005). Practical guidelines for expert-judgment-based software effort estimation. *Software, IEEE*, *22*(3), 57-63.

Jorgensen, M., and Shepperd, M. (2007). A Systematic Review of Software Development Cost Estimation Studies. *Software Engineering, IEEE Transactions on*, *33*(1), 33-53.

Junqueira, D. C., Bittar, T. J. and Fortes, R. P. M. (2008). A fine-grained and flexible version control for software artifacts. In *Proceedings of the 26th annual ACM international conference on Design of communication*. Lisbon, Portugal: ACM. ISBN 978-1-60558-083-8, 185–192. doi:10.1145/1456536.1456576.

Kama, N. (2013a). Change Impact Analysis for the Software Development Phase: State-of-the-art. *International Journal of Software Engineering and Its Applications*, *7*(2), 10.

Kama, N. (2013b). Integrated Change Impact Analysis Approach for the Software Development Phase. *International Journal of Software Engineering & Its Applications*, *7*(2), 9.

Kama, N. M. (2011). *A change impact analysis framework for the software development phase/Mohd Nazri Kama.* Thesis (Ph.D). University of Western Australia.

Kama, N., & Halimi, M. (2013). Extending Change Impact Analysis Approach for Change Effort Estimation in the Software Development Phase. In *WSEAS International Conference. Proceedings. Recent Advances in Computer Engineering Series* (No. 12). WSEAS.

Kama, N., and Azli, F. (2012). *A Change Impact Analysis Approach for the Software Development Phase.* Paper presented at the Proceedings of the 2012 19th Asia-Pacific Software Engineering Conference - Volume 01.

Kama, N., French, T., and Reynolds, M. (2010). Impact Analysis using Class Interaction Prediction Approach. *Proceedings of the 2010 conference on New Trends in Software Methodologies, Tools and Techniques: Proceedings of the 9th SoMeT_10.* 96-111.

Kannenberg, A., & Saiedian, H. (2009). Why software requirements traceability remains a challenge. *In The Journal of Defense Software Engineering.*

Kapfhammer, G. M., & Soffa, M. L. (2008, February). Database-aware test coverage monitoring. In *Proceedings of the 1st India software engineering conference* (pp. 77-86). ACM.

Kaur, R., and Sengupta, J. (2013). Software Process Models and Analysis on Failure of Software Development Projects. *CoRR, abs/1306.1068.*

Kitchenham, B. A. (1996). Evaluating software engineering methods and tool part 1: The evaluation context and

evaluation methods. *ACM SIGSOFT Software Engineering Notes*, *21*(1), 11-14.

Kitchenham, B., Linkman, S., & Law, D. (1997). DESMET: a methodology for evaluating software engineering methods and tools. *Computing & Control Engineering Journal*, *8*(3), 120-126.

Kushwaha, D. S., & Misra, A. K. (2008). Software test effort estimation. *ACM SIGSOFT Software Engineering Notes*, *33*(3), 1-5.

Law, J., & Rothermel, G. (2003, May). Whole program path-based dynamic impact analysis. In *Software Engineering, 2003. Proceedings of 25th International Conference on* (pp. 308-318). IEEE.

Lehman, M., & Fernández-Ramil, J. C. (2006). Software Evolution. *SOFTWARE EVOLUTION AND FEEDBACK*, 7.

Lehman, M.M. and Ramil, J.F. (2002). Software Evolution and Software Evolution Processes. *Annals of Software Engineering*. 14: 275-309.

Lehtinen, T. O. A., Mäntylä, M. V., Vanhanen, J., Itkonen, J., and Lassenius, C. (2014). Perceived causes of software project failures – An analysis of their relationships. *Information and Software Technology*, *56*(6), 623-643.

Lempia, D., & Miller, S. (2006). Requirements engineering management. In *National Software and Complex Electronic Hardware Standardization Conference, Atlanta, GA.*

Li, B., Sun, X., Leung, H., & Zhang, S. (2013). A survey of code-based change impact analysis techniques. *Software Testing, Verification and Reliability*, *23*(8), 613-646.

Li, J., Ruhe, G., Al-Emran, A., and Richter, M. (2007). A flexible method for software effort estimation by analogy. *Empirical Software Engineering*, *12*(1), 65-106.

Li, Y., Li, J., Yang, Y., & Li, M. (2008). Requirement-centric traceability for change impact analysis: a case study. In

Making Globally Distributed Software Development a Success Story (pp. 100-111). Springer Berlin Heidelberg.

Lingampally, R., Gupta, A., & Jalote, P. (2007, January). A multipurpose code coverage tool for Java. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on* (pp. 261b-261b). IEEE.

Liu, W.K. (2013). Application of the Fuzzy Delphi Method and the Fuzzy Analytic Hierarchy Process for the Managerial Competence of Multinational Corporation Executives. International Journal of e-Education, e-Business, e-Management and e-Learning,*3*(4), 313-317.

Lormans, M., & Van Deursen, A. (2005, November). Reconstructing requirements coverage views from design and test using traceability recovery via LSI. In *Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering* (pp. 37-42).

Lormans, M., & Van Deursen, A. (2006, March). Can LSI help reconstructing requirements traceability in design and test?. In *Software Maintenance and Reengineering, 2006. CSMR 2006. Proceedings of the 10th European Conference on* (pp. 10-pp). IEEE.

Lormans, M., Van Dijk, H., Van Deursen, A., Nocker, E. and de Zeeuw, A. (2004). Managing evolving requirements in an outsourcing context: an industrial experience report. In *Software Evolution, 2004. Proceedings. 7th International Workshop on Principles of.* IEEE, 149–158

Lucia, A. D., Fasano, F., Oliveto, R., & Tortora, G. (2007). Recovering traceability links in software artifact management systems using information retrieval methods. *ACM Transactions on Software Engineering and Methodology (TOSEM), 16*(4), 13.

Lulu, H., and Yeong-Tae, S. (2007). Precise Dynamic Impact Analysis with Dependency Analysis for Object-oriented Programs. *Software Engineering Research, Management*

& Applications, 2007. SERA 2007. 5th ACIS International Conference on. 20-22 Aug. 2007 374-384.

M. Khezrian (2013), *An Efficient Service Selection Approach Based on Multi- Criteria Decision Making And Web Service Modeling Ontology*, Doctor Philosophy, Universiti Teknologi Malaysia, Skudai.

Mack, Z., & Sharples, S. (2009). The importance of usability in product choice: A mobile phone case study. *Ergonomics*, *52*(12), 1514-1528.

Mäder, P., & Egyed, A. (2012, September). Assessing the effect of requirements traceability for software maintenance. In *2012 28th IEEE International Conference on Software Maintenance (ICSM)* (pp. 171-180). IEEE.

Mäkäräinen, M. (2000). Software change management process in the development of embedded software. Dissertation, *VTT Publications*, Technical Research Center of Finland, *4*(1), 6.

Maletic, J. I., Collard, M. L., & Simoes, B. (2005, November). An XML based approach to support the evolution of model-to-model traceability links. In *Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering* (pp. 67-72). ACM.

McGee, S., & Greer, D. (2009, September). A software requirements change source taxonomy. In *2009 Fourth International Conference on Software Engineering Advances* (pp. 51-58). IEEE.

Mei, H., Zhang, L. and Yang, F. (2002). A component-based software configuration management model and its supporting system. *Journal of Computer Science and Technology*. 17(4), 432–441. doi:10.1007/BF02943283

Milano, M., & Ullius, D. (1998). Designing powerful training: The sequential-iterative model.

Military Standard (1994) MIL-STD-498: Software Development and Documentation. Department of Defense United States.

Misurda, J., Clause, J. A., Reed, J. L., Childers, B. R., & Soffa, M. L. (2005, May). Demand-driven structural testing with dynamic instrumentation. In *Software Engineering, 2005. ICSE 2005. Proceedings. 27th International Conference on* (pp. 156-165). IEEE.

Muhammad Imran Yousof. 2007. Using experts' opinions through Delphi technique. *Practical Assessment Research & Evaluation*,12 (4): 1-8.

Muhammad, I. Y. (2007). The Delphi technique. *Essays in Education, 20,* 80-89.

Murta, L., CorrÃła, C., PrudÃłncio, J. G. and Werner, C. (2008a). Towards odysseyVCS 2: improvements over a UML-based version control system. In *Proceedings of the 2008 international workshop on Comparison and versioning of software models*. Leipzig, Germany: ACM. ISBN 978-1-60558-045-6, 25–30. doi:10.1145/1370152. 1370159

Myers, G. J., Sandler, C., & Badgett, T. (2011). *The art of software testing*. John Wiley & Sons

Nageswaran, S. (2001, June). Test effort estimation using use case points. In *Quality Week* (pp. 1-6).

Najumudheen, E. S. F., Mall, R., & Samanta, D. (2011). Test coverage analysis based on an object-oriented program model. *Journal of Software Maintenance and Evolution: Research and Practice, 23*(7), 465-493.

Nasurddin, A. M., & Osman, I. (2006). *Pengantar pengurusan*. Utusan Publications.

Ngah, A. (2012). Regression test selection by exclusion (Doctoral dissertation, Durham University).

Nguyen, T., Munson, E. and Thao, C. (2005). Object-oriented configuration management technology can improve software architectural

traceability. In *Third ACIS International Conference on Software Engineering Research, Management and Applications, 2005*. August. 86–93. doi:10.1109/SERA.2005.54.

Nguyen, V., Deeds-Rubin, S., Tan, T. and Boehm, B. (2007). A SLOC counting standard. In *COCOMO II Forum*, vol. 2007.

Nguyen, V., Huang, L., and Boehm, B. (2011). An analysis of trends in productivity and cost drivers over years. *Proceedings of the 7th International Conference on Predictive Models in Software Engineering*. Banff, Alberta, Canada: 1-10.

Nguyen, V., Steece, B., and Boehm, B. (2008). *A constrained regression technique for cocomo calibration.* Paper presented at the Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement.

Nistor, E. C., Erenkrantz, J. R., Hendrickson, S. A., & Van Der Hoek, A. (2005, September). ArchEvol: versioning architectural-implementation relationships. In *Proceedings of the 12th international workshop on Software configuration management* (pp. 99-111). ACM.

Noh, N. M., Siraj, S., Jamil, M. R. M., Husin, Z., & Sapar, A. A. (2015). Design of Guidelines on the Learning Psychology in the Use of Facebook as a Medium for Teaching & Learning in Secondary School. *Turkish Online Journal of Educational Technology-TOJET*, *14*(1), 39-44.

Nurmuliani, N., Zowghi, D., & Powell, S. (2004). Analysis of requirements volatility during software development life cycle. In *Software Engineering Conference, 2004. Proceedings. 2004 Australian* (pp. 28-37). IEEE.

Nurmuliani, N., Zowghi, D., & Williams, S. (2006). Requirements volatility & its impact on change effort: Evidence based research n software development projects. In *Verified OK*. University of South Australia.

Nurulrabihah, M. N. (2013). Siti Hajar. *AR, Norlidah, A., Saedah, S., Mohd Ridhuan, MJ & Zaharah, H*, 1261-1270.

Oliveto, P. S., He, J., & Yao, X. (2007). Time complexity of evolutionary algorithms for combinatorial optimization: A decade of results. *International Journal of Automation and Computing*, *4*(3), 281-293.

Oliveto, R., Antoniol, G., Marcus, A., & Hayes, J. (2007, October). Software artefact traceability: the never-ending challenge. In *Software Maintenance, 2007. ICSM 2007. IEEE International Conference on* (pp. 485-488). IEEE.

Olsen, N. C. (1993). The software rush hour (software engineering). *Software, IEEE*, *10*(5), 29-37.

Omar, S.F. (2013). A Software Traceability Approach To Support Requirement Based Test Coverage Analysis. Masters. Universiti Teknologi Malaysia.

Orso, A., Apiwattanapong, T., & Harrold, M. J. (2003). Leveraging field data for impact analysis and regression testing. *ACM SIGSOFT Software Engineering Notes*, *28*(5), 128-137.

Orso, A., Harrold, M. J., Rosenblum, D., Rothermel, G., Soffa, M. L., & Do, H. (2001). Using component metacontent to support the regression testing of component-based software. In *Software Maintenance, 2001. Proceedings. IEEE International Conference on* (pp. 716-725). IEEE.

Orso, A., Shi, N., & Harrold, M. J. (2004, October). Scaling regression testing to large software systems. In *ACM SIGSOFT Software Engineering Notes* (Vol. 29, No. 6, pp. 241-251). ACM.

Pachauri, A., & Srivastava, G. (2013). Automated test data generation for branch testing using genetic algorithm: An improved approach using branch ordering, memory and elitism. *Journal of Systems and Software*, *86*(5), 1191-1208.

Passos, L., Czarnecki, K., Apel, S., Wąsowski, A., Kästner, C., & Guo, J. (2013, January). Feature-oriented software

evolution. In *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems* (p. 17). ACM.

Perry, J., & Linsley, S. (2006). The use of the nominal group technique as an evaluative tool in the teaching and summative assessment of the inter-personal skills of student mental health nurses. *Nurse education today*, *26*(4), 346-353.

Pfleeger, S. L., & Bohner, S. A. (1990, November). A framework for software maintenance metrics. In *Software Maintenance, 1990., Proceedings., Conference on* (pp. 320-327). IEEE.

Powell, C. (2003). The Delphi technique: myths and realities. Journal of Advanced Nursing, 41(4), 376-382.

Pravin, A., & Srinivasan, S. (2013). Effective test case selection and prioritization in regression testing, *Journal of Computer Science*, *9* (5),p654.

Qian Yang, J. Jenny Li and David M. Weiss. (2007). A survey of coverage-based testing tools. *The Computer Journal of Advance Access*, *23*(8), 99-103.

Rafi, D. M., Moses, K. R. K., Petersen, K., & Mäntylä, M. V. (2012, June). Benefits and limitations of automated software testing: Systematic literature review and practitioner survey. In *2012 7th International Workshop on Automation of Software Test (AST)* (pp. 36-42). IEEE.

Ragin, C. C. (2007). Qualitative comparative analysis using fuzzy sets (fsQCA). In Configurational comparative analysis. London: Sage Publications.

Ramesh, B., & Jarke, M. (2001). Toward reference models for requirements traceability. *Software Engineering, IEEE Transactions on*, *27*(1), 58-93.

Requirements Traceability (2013), http:// en.wikipedia.org/ wiki/ Requirements_traceability

Richey, R. C., & Klein, J. D. (2008). Research on design and development. In J. M. Spector, M. D. Merrill, J. van Merrienboer, & M. P.

Driscoll (Eds.), *Handbook of research for educational communications and technology* (3rd ed., pp. 748–757). Mahwah, NJ: Lawrence Erlbaum Associates, Publishers

Richey, R. C., & Klein, J. D. (2014). Design and development research. In *Handbook of research on educational communications and technology* (pp. 141-150). Springer, New York, NY.

Richey, R., & Klien, J. (2007). *Design and development research: Method, strategies and issues.* London: Erlbaum

Riebisch, M. (2004, May). Supporting evolutionary development by feature models and traceability links. In *Engineering of Computer-Based Systems, 2004. Proceedings. 11th IEEE International Conference and Workshop on the* (pp. 370-377). IEEE

Rochimah, S., Kadir, W. M. N., & Abdullah, A. H. (2007, August). An evaluation of traceability approaches to support software evolution. In *Software Engineering Advances, 2007. ICSEA 2007. International Conference on* (pp. 19-19). IEEE.

Rochimah, S., Kadir, W. M. N., & Abdullah, A. H. (2009). Multifaceted Requirement Traceability Approach to Support Software Evolution 5th Postgraduate Annual Seminars - PARS.

Rohatgi, A., Hamou-Lhadj, A., & Rilling, J. (2008, June). An approach for mapping features to code based on static and dynamic analysis. In *Program Comprehension, 2008. ICPC 2008. The 16th IEEE International Conference on* (pp. 236-241). IEEE.

Rothermel, G., & Harrold, M. J. (1997). A safe, efficient regression test selection technique. *ACM Transactions on Software Engineering and Methodology (TOSEM), 6*(2), 173-210.

Rothermel, G., Untch, R. H., Chu, C., & Harrold, M. J. (2001). Prioritizing test cases for regression testing. *Software Engineering, IEEE Transactions on, 27*(10), 929-948. RTCA

(1992). DO-178B, Software considerations in airborne systems and equipment certification. Washington D.C

Rowe, G., & Wright, G. (2011). The Delphi technique: Past, present, and future prospects—Introduction to the special issue. *Technological forecasting and social change*, *78*(9), 1487-1490.

Royce, W. (2005). Successful software management style: Steering and balance. *IEEE software*, *22*(5), 40-47.

Sakamoto, K., Washizaki, H., & Fukazawa, Y. (2010, July). Open code coverage framework: A consistent and flexible framework for measuring test coverage supporting multiple programming languages. In *Quality Software (QSIC), 2010 10th International Conference on* (pp. 262-269). IEEE.

Sanchez-Lezama, A. P., Cavazos-Arroyo, J., & Albavera-Hernandez, C. (2014). Applying the Fuzzy Delphi Method for determining socio-ecological factors that influence adherence to mammography screening in rural areas of Mexico. *Cadernos de saúde pública*, *30*, 245-258.

Sapar, A. A., Siraj, S., & Noh, N. R. M. (2014). Aplikasi Teknik Fuzzy Delphi Terhadap Keperluan Aspek 'Riadhah Ruhiyyah'untuk Profesionalisme Perguruan Pendidikan Islam. *The Online Journal of Islamic Education July*, *2*(2).

Seo, K.I, Choi, E.M (2006). Comparison of Five Black-box Testing Methods for Object-Oriented Software. *Proceedings f the Fourth International Conference on Software Engineering Research, Management and Application (SERA'06).*

Shahid, M. and Ibrahim, S. (2013). A New Model For Requirements to Code Traceability to Support Code Coverage Analysis. *Asian Academic Research Journal of Multidisciplinary (AARJMD).* 1(14), 159–172.

Shahid, M., & Ibrahim, S. (2016, January). Change impact analysis with a software traceability approach to support software maintenance. In *2016 13th International Bhurban conference on applied sciences and technology (IBCAST)* (pp. 391-396).

IEEE.Sharif, B., Khan, S. A., and Bhatti, M. W. (2012). Measuring the Impact of Changing Requirements on Software Project Cost: An Empirical Investigation. *IJCSI International Journal of Computer Science Issues*. 9(3), 170-174.

Sharma, A., & Kushwaha, D. S. (2012). Estimation of software development effort from requirements based complexity. *Procedia Technology*, *4*, 716-722.

Singh, M., and Vyas, R. (2012). Requirements Volatility in Software Development Process. *International Journal of Soft Computing*, *2*.

Siraj, S. (2008). *Kurikulum masa depan*. Penerbit Universiti Malaya.

Small, A.W.; Downey, E.A. (2001). Managing Change: Some Important Aspects. *Proceedings of the Change Management and the New Industrial Revolutio (IEMC'01)*. October 7-9. USA: IEEE Computer Society. 50-57.

Sommerville, I. (2000). Evolution and change. *Computing and Control Engineering Journal*, *11*(4), 154-155.

Sommerville, I. (2004). *Software Engineering*. Seventh Ed. USA: Addison-Wesley Publishing Company

Sommerville, I. (2007). *Software Engineering*. In (pp. 659-713): Addison-Wesley.

Spanoudakis, G., Zisman, A., Pérez-Minana, E., & Krause, P. (2004). Rule-based generation of requirements traceability relations. *Journal of Systems and Software*, *72*(2), 105-127.

Spillner, J., Bombach, G., Matthischke, S., Muller, J., Tzschichholz, R., & Schill, A. (2011, December). Information dispersion over redundant arrays of optimal cloud storage for desktop users. In *2011 Fourth IEEE International Conference on Utility and Cloud Computing* (pp. 1-8). IEEE.

Stammel, J., and Trifu, M. (2011). *Tool-supported estimation of software evolution effort in service-oriented systems*. Paper

presented at the First International Workshop on Model-Driven Software Migration (MDSM 2011), 56.

Suri, P. K., and Ranjan, P. (2012). Comparative Analysis of Software Effort Estimation Techniques. *International Journal of Computer Applications (0975–8887)*, *48*(21).

Swanson, R. A., & Falkman, S. K. (1997). Training delivery problems and solutions: Identification of novice trainer problems and expert trainer solutions. *Human Resource Development Quarterly*, *8*(4), 305-314.

Tahriri, F., Mousavi, M., Haghighi, S. H., & Dawal, S. Z. M. (2014). The application of fuzzy Delphi and fuzzy inference system in supplier ranking and selection. *Journal of Industrial Engineering International*, *10*(3), 66.

Tang, C.W. and , Wu, C.T. (2010). Obtaining a picture of undergraduate education quality: a voice from inside the university, Springer. Higher Education,60, 269-286.

Tester, C. (2007). Foundation Level Syllabus.

The Chaos Report, 2015.

The Standish Group, Chaos, Standish Group Report, 2015

Van De, A., & Delbecq, A. L. (1971). Nominal versus interacting group processes for committee decision-making effectiveness. *Academy of Management Journal*, *14*(2), 203-212

Vasa, R., Schneider, J. G., & Nierstrasz, O. (2007, October). The inevitable stability of software change. In *Software Maintenance, 2007. ICSM 2007. IEEE International Conference on* (pp. 4-13). IEEE.

Verner, J. M., Evanco, W. M., and Cerpa, N. (2007). State of the practice: An exploratory analysis of schedule estimation and software project success prediction. *Information and Software Technology*, *49*(2), 181-193.

Wahl, N. J. (1999). An overview of regression testing. *ACM SIGSOFT Software Engineering Notes*, *24*(1), 69-73.

Whalen, M. W., Rajan, A., Heimdahl, M. P., & Miller, S. P. (2006, July). Coverage metrics for requirements-based testing. In *Proceedings of the 2006 international symposium on Software testing and analysis* (pp. 25-36). ACM.

Whalen, M.W, Rajan, A., Heimdahl, M.P.E, Miller, S.P. (2003). *Coverage Metrics for Requirements-Based Testing*. ACM. ISSTA'06, July 17–20, 2006, Portland, Maine, USA.

Williams, P. L., White, N., Klem, R., Wilson, S. E., & Bartholomew,P. (2006). Clinical education and training: jousing the nominal group technique in research with radiographers to identify factors affecting quality and capacity. *Radiography*, *12*(3), 215-224

Wu, X., Li, J. J., Weiss, D., & Lee, Y. (2007, May). Coverage-based testing on embedded systems. In *Proceedings of the Second International Workshop on Automation of Software Test* (p. 7). IEEE Computer Society.

Yadla, S., Hayes, J. H., & Dekhtyar, A. (2005). Tracing requirements to defect reports: an application of information retrieval techniques. *Innovations in Systems and Software Engineering*, *1*(2), 116-124.

Yinhuan, Z., Beizhan, W., Yilong, Z., and Liang, S. (2009, 25-28 July 2009).*Estimation of software projects effort based on function point.* Paper presented at the Computer Science & Education, 2009. ICCSE '09. 4th International Conference on, 941-943.

Z. Xiaochun, Z. Bo, W. Fan, Q. Yi. Chen and Lu, (2008.) "Estimate Test Execution Effort at an Early Stage: An Empirical Study", International Conference on Cyber World, IEEE Computer Society, pp 195-200,

Zadeh L.A. (1965). Fuzzy sets and systems, System Theory (Fox J., ed.), Microwave Research Institute Symposia Series XV, Polytechnic Press, Brooklyn, NY, 29- 37. Reprinted in Int. J. of General Systems, 17, 1990, 129-138

BIOGRAPHY

MAZIDAH MAT REJAB is Senior Lecture at Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn. She received BSc in Software Engineering at Universiti Teknologi Malaysia (UTM) and MSc in Software Engineering at Universiti Teknologi Malaysia (UTM). Her PhD degree in Software Engineering from Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia. She had 12 years industry experience in software engineering field.

NURULHUDA FIRDAUS MOHD AZMI is Senior Lecturer at Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia. She received BSc in Computer Science at Universiti

Teknologi Malaysia (UTM) and MSc in Applied Statistics at Universiti Putra Malaysia (UPM). Her PhD degree is in Computer Science at University of York, UK with the research focus on Artificial Immune Systems for Adaptive Information Filtering. She currently active in research in the area of Data Science and Operational Research (OR). She is a member of the Malaysia Association of Information System (myAIS) and Association for Computing Machinery (ACM). She is currently attached in Biostatistics Data Repository Sector at National Institute of Health (NIH) under Malaysia Ministry of Health.

SURIAYATI CHUPRAT is Associate Professor at Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia. She received BSc and MSc degrees in Computer Science (Software Engineering) and PhD in Mathematics from Universiti Teknologi Malaysia. In part of her PhD research, she was attached to the University of North Carolina, USA. She did a postdoctoral program at the University of York, UK. Her research interests include Software Engineering, Algorithms and Scheduling Theories, Real-time Systems and Parallel Computing. She currently active in research and development projects in the area of Big Data Analytics and Cyber Security. She is a member of the ACM Professional and IEEE Computer.

HAIRULNIZAM MAHDIN received the Ph.D. degree from Deakin University, Australia, in 2012. He is currently an Associate Professor with the Department of Information Security and Web Technology, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia. He is also currently serving as the Deputy Dean of Research, Development, and Publication. His research interests include edge data processing, software engineering and information security. He has published almost 100 publications and also has served numbers of international conferences and journal in various capacity including chair, co-chair and internation editor.

Traceability Matrix

for Effort Estimation in Change Request

In the last decade, the management of IT projects has become a challenging task. The latest published figures on the status of IT projects indicate a large failure rate, which has created a crucial challenge for project managers. In maintenance phase, the impact of changes is an important aspect due to the evolving environment of the software development life cycle. The aim of this book is to investigate the need and significant use of the traceability matrix for effort estimation to accommodate changes request in maintenance tasks. Change request is necessary to keep task current and reusable. Software evolves over time due to specific changes during development and maintenance at every point, the management aspect of modifications can become more complicated and potentially risky. Many of the current traceability approaches and tools are devoted to and restricted to high-level objects such as specifications and fewer capabilities made available to handle lower-level artefacts such as classes and codes While effort estimates have been in place for decades, it remains a major challenge for project management to make accurate estimates and, ultimately, to successfully complete the IT project.

THE AUTHORS

MAZIDAH MAT REJAB is Lecture at Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia (UTHM). She received BSc in Software Engineering at Universiti Teknologi Malaysia (UTM) and MSc in Software Engineering at UTM. Her PhD degree in Software Engineering from Razak Faculty of Technology and Informatics, UTM. She had 12 years industry experience in software engineering field.

NURULHUDA FIRDAUS MOHD AZMI is Senior Lecturer at Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia (UTM). She received BSc in Computer Science at UTM and MSc in Applied Statistics at Universiti Putra Malaysia (UPM). Her PhD degree is in Computer Science at University of York, UK with the research focus on Artificial Immune Systems for Adaptive Information Filtering. She currently active in research in the area of Data Science and Operational Research (OR). She is a member of the Malaysia Association of Information System (myAIS) and Association for Computing Machinery (ACM). She is currently attached in Biostatistics Data Repository Sector at National Institute of Health (NIH) under Malaysia Ministry of Health.

SURIAYATI CHUPRAT is Associate Professor at Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia (UTM). She received BSc and MSc degrees in Computer Science (Software Engineering) and PhD in Mathematics from UTM. In part of her PhD research, she was attached to the University of North Carolina, USA. She did a postdoctoral program at the University of York, UK. Her research interests include Software Engineering, Algorithms and Scheduling Theories, Real-time Systems and Parallel Computing. She currently active in research and development projects in the area of Big Data Analytics and Cyber Security. She is a member of the ACM Professional and IEEE Computer.

HAIRULNIZAM MAHDIN received the PhD degree from Deakin University, Australia, in 2012. He is currently an Associate Professor with the Department of Information Security and Web Technology, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia (UTHM). He is also currently serving as the Deputy Dean of Research, Development, and Publication. His research interests include edge data processing, software engineering and information security. He has published almost 100 publications and also has served numbers of international conferences and journal in various capacity including chair, co-chair and internation editor.

Penerbit UTHM

ISBN 978-967-0061-62-7

9 789670 061627

For more information, please scan the code