A DOCUMENT-BASED TRACEABILITY MODEL FOR TEST MANAGEMENT

AZRI BIN AZMI

UNIVERSITI TEKNOLOGI MALAYSIA
A DOCUMENT-BASED TRACEABILITY MODEL FOR TEST MANAGEMENT

AZRI BIN AZMI

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

DECEMBER 2017
Specially dedicated to all my family and relatives.
ACKNOWLEDGEMENT

First of all, I am thankful to God for giving me the opportunity and strength to complete my PhD. I would like to take this opportunity to thank my main supervisor, Prof. Dr. Suhaimi Bin Ibrahim for his encouragement, constant support, advice and inspiration throughout this research.

I would also like to thank my wife and kids for their patience during the course of my PhD. I am also grateful to all my friends who kept me motivated during my research especially to Dr. Othman, Dr. Saiful, Mdm. Haslina and Dr. Nazri Kama.

My thanks also go to all UTM AIS staff and individuals who have been involved directly or indirectly in the project. Lastly, my appreciation also goes to the experts for evaluating my model and prototype.
ABSTRACT

Software testing has become more complicated in the emergence of distributed network, real-time environment, third party software enablers and the need to test system at multiple integration levels. These scenarios have created more concern over the quality of software testing. The quality of software has been deteriorating due to inefficient and ineffective testing activities. One of the main flaws is due to ineffective use of test management to manage software documentations. In documentations, it is difficult to detect and trace bugs in some related documents of which traceability is the major concern. Currently, various studies have been conducted on test management, however very few have focused on document traceability in particular to support the error propagation with respect to documentation. The objective of this thesis is to develop a new traceability model that integrates software engineering documents to support test management. The artefacts refer to requirements, design, source code, test description and test result. The proposed model managed to tackle software traceability in both forward and backward propagations by implementing multi-bidirectional pointer. This platform enabled the test manager to navigate and capture a set of related artefacts to support test management process. A new prototype was developed to facilitate observation of software traceability on all related artefacts across the entire documentation lifecycle. The proposed model was then applied to a case study of a finished software development project with a complete set of software documents called the On-Board Automobile (OBA). The proposed model was evaluated qualitatively and quantitatively using the feature analysis, precision and recall, and expert validation. The evaluation results proved that the proposed model and its prototype were justified and significant to support test management.
ABSTRAK

# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECLARATION</td>
<td></td>
<td>ii</td>
</tr>
<tr>
<td>DEDICATION</td>
<td></td>
<td>iii</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENT</td>
<td></td>
<td>iv</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td></td>
<td>v</td>
</tr>
<tr>
<td>ABSTRAK</td>
<td></td>
<td>vi</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td></td>
<td>vii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td></td>
<td>xiii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td></td>
<td>xv</td>
</tr>
<tr>
<td>LIST OF ABBREVIATIONS</td>
<td></td>
<td>xix</td>
</tr>
<tr>
<td>LIST OF APPENDICES</td>
<td></td>
<td>xx</td>
</tr>
</tbody>
</table>

1 INTRODUCTION

1.1 Overview 1
1.2 Background of the Problem 3
1.3 Statement of the Problem 6
1.4 Objectives of the Study 7
1.5 Scope of the Research 7
1.6 Significance of the Study 8
1.7 Thesis Outline 9

2 LITERATURE REVIEW

2.1 Introduction 11
2.2 Software Testing 11
2.2.1 Discussion on the Motivation of Software Testing 13

2.3 Software Test Management 14
2.3.1 Activities of Test Management 15
2.3.2 Current Researches on Test Management 18
  2.3.2.1 TAI Approach 18
  2.3.2.2 Sigrid's Approach 19
  2.3.2.3 Agent-based Test Management Approach 20
  2.3.2.4 Architecture Centric Approach 22
  2.3.2.5 Summary of Test Management Approaches 23

2.3.3 Test Management Issues 24
2.3.4 Test Management Summary 28

2.4 Software Documentation 28
2.4.1 Software Testing Documentations versus Test Management 30
2.4.2 Software Documentation Issues 31
2.4.3 Software Documentation Summary 35

2.5 Software Traceability 35
2.5.1 Introduction of Software Traceability 35
2.5.2 Dimension of Traceability 36
  2.5.2.1 Forward and Backward Traceability 37
  2.5.2.2 Implicit and Explicit Traceability 37
  2.5.2.3 Intra-Level and Inter-Level Traceability 38
  2.5.2.4 Material and Immaterial Traceability 39
  2.5.2.5 Pre-RS Traceability and Post-RS Traceability 39

2.5.3 The Needs for Traceability 39
2.5.4 Traceability Approach versus Traceability
Model 40

2.5.5 Some Studies on Existing Traceability Approaches 41
2.5.5.1 Information Retrieval Traceability 42
2.5.5.2 Event-based Traceability 44
2.5.5.3 Goal Centric Traceability 44
2.5.5.4 Scenario-Based Traceability 45
2.5.5.5 Rule-Based Traceability 46
2.5.5.6 Evaluation and Comparative Study of Traceability Approaches 47

2.5.6 Some Studies on Existing Traceability Model 50
2.5.6.1 Inter Requirement Traceability Model 52
2.5.6.2 Coding Phase Requirements Traceability Model 53
2.5.6.3 Total Traceability Model 54
2.5.6.4 End-to-End Traceability Model 55
2.5.6.5 TraceabilityWeb Model 56
2.5.6.6 Evaluation of Software Traceability Models 58
2.5.6.7 Comparative Study and Criteria for Model Development 59

2.5.7 Discussion of Software Traceability and Documentation 65
2.5.8 Issues in Traceability 66
2.5.9 Summary of Software Traceability 71

2.6 Summary of the Chapter 71

3 RESEARCH METHODOLOGY 73
3.1 Introduction 73
3.2 Research Design Strategies and Research Process 73
4 DESIGN AND DEVELOPMENT OF THE PROPOSED MODEL

4.1 Overview

4.2 Rationale of the DBT Model
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3</td>
<td>Construction of the DBT Model Component</td>
<td>99</td>
</tr>
<tr>
<td>4.4</td>
<td>Definition of the DBT Model</td>
<td>100</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Extractor</td>
<td>102</td>
</tr>
<tr>
<td>4.4.2</td>
<td>XML Repository</td>
<td>103</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Analyzer</td>
<td>103</td>
</tr>
<tr>
<td>4.4.4</td>
<td>Traceability Engine</td>
<td>104</td>
</tr>
<tr>
<td>4.4.5</td>
<td>Traceability Repository</td>
<td>104</td>
</tr>
<tr>
<td>4.4.6</td>
<td>Document Generator</td>
<td>105</td>
</tr>
<tr>
<td>4.5</td>
<td>The DBT Architecture</td>
<td>105</td>
</tr>
<tr>
<td>4.6</td>
<td>The DBT Process</td>
<td>106</td>
</tr>
<tr>
<td>4.7</td>
<td>Design and Development of DBT</td>
<td>109</td>
</tr>
<tr>
<td>4.7.1</td>
<td>DBT Use Cases</td>
<td>109</td>
</tr>
<tr>
<td>4.7.2</td>
<td>DBT Class Diagram</td>
<td>110</td>
</tr>
<tr>
<td>4.7.3</td>
<td>DBT Package Diagram Details</td>
<td>111</td>
</tr>
<tr>
<td>4.7.3.1</td>
<td>Extractor Package</td>
<td>113</td>
</tr>
<tr>
<td>4.7.3.2</td>
<td>Analyzer Package</td>
<td>115</td>
</tr>
<tr>
<td>4.7.3.3</td>
<td>Traceability Engine Package</td>
<td>119</td>
</tr>
<tr>
<td>4.7.3.4</td>
<td>Report Generator Package</td>
<td>126</td>
</tr>
<tr>
<td>4.7.3.5</td>
<td>Queries Package</td>
<td>127</td>
</tr>
<tr>
<td>4.7.4</td>
<td>DBT Relational Database</td>
<td>127</td>
</tr>
<tr>
<td>4.7.4.1</td>
<td>Perform Trace Links Between Artefacts</td>
<td>130</td>
</tr>
<tr>
<td>4.8</td>
<td>DBT Graphical User Interface</td>
<td>134</td>
</tr>
<tr>
<td>4.9</td>
<td>Discussion on Design and Development of the Proposed Model</td>
<td>140</td>
</tr>
<tr>
<td>4.10</td>
<td>Summary</td>
<td>141</td>
</tr>
<tr>
<td>5</td>
<td>EVALUATION OF THE PROPOSED MODEL</td>
<td>142</td>
</tr>
<tr>
<td>5.1</td>
<td>Overview</td>
<td>142</td>
</tr>
<tr>
<td>5.2</td>
<td>Features Mapping for Evaluation of Proposed Model</td>
<td>142</td>
</tr>
<tr>
<td>5.3</td>
<td>Results of Feature Analysis</td>
<td>146</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Feature Set 1</td>
<td>146</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
<td></td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
<td></td>
</tr>
<tr>
<td>5.3.2 Feature Set 2</td>
<td>150</td>
<td></td>
</tr>
<tr>
<td>5.3.3 Feature Set 3</td>
<td>152</td>
<td></td>
</tr>
<tr>
<td>5.3.4 Feature Set 4</td>
<td>153</td>
<td></td>
</tr>
<tr>
<td>5.3.5 Feature Set 5</td>
<td>155</td>
<td></td>
</tr>
<tr>
<td>5.3.6 Summary Result of Feature Analysis</td>
<td>157</td>
<td></td>
</tr>
<tr>
<td>5.4 Case Study - OBA</td>
<td>161</td>
<td></td>
</tr>
<tr>
<td>5.4.1 OBA Functionalities</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>5.4.2 Traceability in OBA</td>
<td>164</td>
<td></td>
</tr>
<tr>
<td>5.4.3 OBA Dataset</td>
<td>165</td>
<td></td>
</tr>
<tr>
<td>5.5 Quantitative Evaluation Method – Precision and Recall</td>
<td>167</td>
<td></td>
</tr>
<tr>
<td>5.5.1 Traceability Between STD and SRS</td>
<td>167</td>
<td></td>
</tr>
<tr>
<td>5.5.2 Traceability Between SDD and SRS</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>5.6 Expert Validation</td>
<td>173</td>
<td></td>
</tr>
<tr>
<td>5.6.1 Questionnaire Result</td>
<td>173</td>
<td></td>
</tr>
<tr>
<td>5.7 Discussion on the Evaluation Results</td>
<td>179</td>
<td></td>
</tr>
<tr>
<td>5.8 Summary</td>
<td>181</td>
<td></td>
</tr>
<tr>
<td>6 CONCLUSION</td>
<td>182</td>
<td></td>
</tr>
<tr>
<td>6.1 Summary and Achievement</td>
<td>182</td>
<td></td>
</tr>
<tr>
<td>6.2 Contributions of the Research</td>
<td>184</td>
<td></td>
</tr>
<tr>
<td>6.3 Limitation and Feature Works</td>
<td>185</td>
<td></td>
</tr>
<tr>
<td>REFERENCES</td>
<td>187</td>
<td></td>
</tr>
<tr>
<td>Appendices A - E</td>
<td>207-235</td>
<td></td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Test management approach comparison</td>
<td>23</td>
</tr>
<tr>
<td>2.2</td>
<td>Test management issues</td>
<td>26</td>
</tr>
<tr>
<td>2.3</td>
<td>Software documentation issues</td>
<td>33</td>
</tr>
<tr>
<td>2.4</td>
<td>Evaluation of traceability approaches</td>
<td>48</td>
</tr>
<tr>
<td>2.5</td>
<td>Comparative study of traceability approaches</td>
<td>51</td>
</tr>
<tr>
<td>2.6</td>
<td>Comparative result of traceability models</td>
<td>58</td>
</tr>
<tr>
<td>2.7</td>
<td>Comparative study of traceability models</td>
<td>62</td>
</tr>
<tr>
<td>2.8</td>
<td>Criteria for model of component</td>
<td>63</td>
</tr>
<tr>
<td>2.9</td>
<td>Traceability issues</td>
<td>67</td>
</tr>
<tr>
<td>3.1</td>
<td>Operational framework</td>
<td>77</td>
</tr>
<tr>
<td>3.2</td>
<td>Method proposed by DESMET</td>
<td>81</td>
</tr>
<tr>
<td>3.3</td>
<td>Scoring scale</td>
<td>83</td>
</tr>
<tr>
<td>3.4</td>
<td>Level of importance of a feature with multiplier</td>
<td>84</td>
</tr>
<tr>
<td>3.5</td>
<td>Features and sub-features used in the analysis</td>
<td>85</td>
</tr>
<tr>
<td>3.6</td>
<td>Feature set weighting</td>
<td>87</td>
</tr>
<tr>
<td>3.7</td>
<td>Usability attributes adoption</td>
<td>91</td>
</tr>
<tr>
<td>4.1</td>
<td>Selection of component for DBT model</td>
<td>99</td>
</tr>
<tr>
<td>4.2</td>
<td>List of table in DBT</td>
<td>128</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>5.1</td>
<td>Selection of features for software development tool evaluation</td>
<td>145</td>
</tr>
<tr>
<td>5.2</td>
<td>Comparison of overall scores details of feature analysis</td>
<td>148</td>
</tr>
<tr>
<td>5.3</td>
<td>Feature set scores and overall scores comparison</td>
<td>158</td>
</tr>
<tr>
<td>5.4</td>
<td>OBA documentation set</td>
<td>166</td>
</tr>
<tr>
<td>5.5</td>
<td>OBA component/work product</td>
<td>166</td>
</tr>
<tr>
<td>5.6</td>
<td>OBA tracing activities</td>
<td>167</td>
</tr>
<tr>
<td>5.7</td>
<td>Tracing between STD and SRS results</td>
<td>169</td>
</tr>
<tr>
<td>5.8</td>
<td>Tracing between SDD and SRS results</td>
<td>171</td>
</tr>
<tr>
<td>5.9</td>
<td>Experts profile</td>
<td>174</td>
</tr>
<tr>
<td>5.10</td>
<td>Cross tabulation percentage of gender and qualification</td>
<td>174</td>
</tr>
<tr>
<td>5.11</td>
<td>Cross tabulation of working experience</td>
<td>174</td>
</tr>
<tr>
<td>5.12</td>
<td>List of questions and usability criteria</td>
<td>175</td>
</tr>
<tr>
<td>5.13</td>
<td>Mean and standard deviation calculation</td>
<td>176</td>
</tr>
<tr>
<td>5.14</td>
<td>Comments/suggestions received from the experts</td>
<td>178</td>
</tr>
<tr>
<td>5.15</td>
<td>5-level mean analysis</td>
<td>178</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE NO.</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The V-model with software testing life cycle</td>
<td>13</td>
</tr>
<tr>
<td>2.2</td>
<td>System architecture of TAI</td>
<td>19</td>
</tr>
<tr>
<td>2.3</td>
<td>The test preparation phase in test management system</td>
<td>20</td>
</tr>
<tr>
<td>2.4</td>
<td>Adaptive test management system</td>
<td>21</td>
</tr>
<tr>
<td>2.5</td>
<td>Structure element service diagram</td>
<td>23</td>
</tr>
<tr>
<td>2.6</td>
<td>Software documentation in SDLC</td>
<td>29</td>
</tr>
<tr>
<td>2.7</td>
<td>Dimension of traceability</td>
<td>38</td>
</tr>
<tr>
<td>2.8</td>
<td>Inter-requirements traceability model</td>
<td>53</td>
</tr>
<tr>
<td>2.9</td>
<td>Coding phase requirements traceability model</td>
<td>54</td>
</tr>
<tr>
<td>2.10</td>
<td>Total traceability model</td>
<td>55</td>
</tr>
<tr>
<td>2.11</td>
<td>End-to-end traceability model</td>
<td>56</td>
</tr>
<tr>
<td>2.12</td>
<td>TraceabilityWeb model</td>
<td>57</td>
</tr>
<tr>
<td>2.13</td>
<td>The proposed research area</td>
<td>72</td>
</tr>
<tr>
<td>3.1</td>
<td>Traceability theoretical framework</td>
<td>75</td>
</tr>
<tr>
<td>3.2</td>
<td>Flowchart of research process</td>
<td>76</td>
</tr>
<tr>
<td>3.3</td>
<td>Research procedure and activities</td>
<td>78</td>
</tr>
<tr>
<td>3.4</td>
<td>General concepts of precision and recall</td>
<td>88</td>
</tr>
<tr>
<td>3.5</td>
<td>OBA interfaces – context diagram</td>
<td>94</td>
</tr>
<tr>
<td>4.1</td>
<td>Proposed document-based traceability (DBT) model</td>
<td>101</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>4.2</td>
<td>Multi linked-list implementation</td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>The architecture of DBT</td>
<td></td>
</tr>
<tr>
<td>4.4</td>
<td>The process of DBT</td>
<td></td>
</tr>
<tr>
<td>4.5</td>
<td>DBT use cases</td>
<td></td>
</tr>
<tr>
<td>4.6</td>
<td>DBT class diagram</td>
<td></td>
</tr>
<tr>
<td>4.7</td>
<td>DBT packages (static organization)</td>
<td></td>
</tr>
<tr>
<td>4.8</td>
<td>DBT extractor package</td>
<td></td>
</tr>
<tr>
<td>4.9</td>
<td>DBT analyzer package</td>
<td></td>
</tr>
<tr>
<td>4.10</td>
<td>Lucene indexing process</td>
<td></td>
</tr>
<tr>
<td>4.11</td>
<td>DBT traceability engine package</td>
<td></td>
</tr>
<tr>
<td>4.12</td>
<td>Multi-bidirectional traceability links between artefacts</td>
<td></td>
</tr>
<tr>
<td>4.13</td>
<td>Part of SRS document - capturing ID and term</td>
<td></td>
</tr>
<tr>
<td>4.14</td>
<td>Part of SDD document - capturing SDD ID</td>
<td></td>
</tr>
<tr>
<td>4.15</td>
<td>Part of SDD document – capturing SDD ID and term</td>
<td></td>
</tr>
<tr>
<td>4.16</td>
<td>Mapping the source code term with SDD ID – SDD document</td>
<td></td>
</tr>
<tr>
<td>4.17</td>
<td>Part of STD document – mapping STD ID</td>
<td></td>
</tr>
<tr>
<td>4.18</td>
<td>Searching for ID in STD for traceability</td>
<td></td>
</tr>
<tr>
<td>4.19</td>
<td>Report generator package</td>
<td></td>
</tr>
<tr>
<td>4.20</td>
<td>Queries package</td>
<td></td>
</tr>
<tr>
<td>4.21</td>
<td>DBT entity relationship diagram (ERD)</td>
<td></td>
</tr>
<tr>
<td>4.22</td>
<td>STD relation (table) with partial data</td>
<td></td>
</tr>
<tr>
<td>4.23</td>
<td>SRS relation (table) with partial data</td>
<td></td>
</tr>
<tr>
<td>4.24</td>
<td>STDxSRS relation (table) with partial data</td>
<td></td>
</tr>
<tr>
<td>4.25</td>
<td>Links between STD and SRS from view of set</td>
<td></td>
</tr>
<tr>
<td>4.26</td>
<td>Macro icon for DBT in Microsoft Word</td>
<td></td>
</tr>
</tbody>
</table>
4.27 STR (partial) - operate cruise control result

4.28 STR (partial) – set car calibration result

4.29 The failed result change into red and highlighted (set car calibration)

4.30 The failed result change into red and highlighted (cruise control)

4.31 Failed summary report

4.32 SRS data interface

4.33 STD data interface

4.34 SRS to STD traceability interface

5.1 Mapping of features for evaluation of proposed model (sub-features not shown)

5.2 Feature set F1 level of scores (traceability)

5.3 F1 percentage of feature set score

5.4 Feature set F2 level of scores (document management)

5.5 F2 percentage of feature set score

5.6 Feature set 3 – level of score (defect management)

5.7 F3 percentage of feature set score

5.8 Feature set 4 – level of score (report)

5.9 F4 percentage of feature set score

5.10 F5 feature set score for activity support

5.11 F5 percentage of feature set score

5.12 Multiple metric graph for feature set score

5.13 Multiple metric graph for overall % score using feature set weightings

5.14 % feature set score versus models

5.15 OBA sub-system component
LIST OF ABBREVIATIONS

CSCI  -  Computer Software Configuration Item
DAS   -  Driving Assistance System
DBT   -  Document-based Traceability
DoD   -  Department of Defense
ERD   -  Entity Relationship diagram
IEEE  -  Institute of Electrical and Electronics Engineers
IRS   -  Interface Requirements Specification
ISO   -  International Organization for Standardization
OBA   -  On-Board Automobile
SDCP  -  Safe Drive Control Panel
SDD   -  Software Design Document
SDLC  -  Software Development Life Cycle
SDP   -  Software Development Plan
SRS   -  Software Requirements Specification
STD   -  Software Test Description
STR   -  Software Test Result
UML   -  Unified Modeling Language
UTM   -  Universiti Teknologi Malaysia
XML   -  Xtensible Markup Language
LIST OF APPENDICES

<table>
<thead>
<tr>
<th>APPENDIX</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>OBA Data Set (Data Extraction – Traceability)</td>
<td>211</td>
</tr>
<tr>
<td>B</td>
<td>OBA Data Set (Data Description)</td>
<td>218</td>
</tr>
<tr>
<td>C</td>
<td>OBA Documents</td>
<td>222</td>
</tr>
<tr>
<td>D</td>
<td>Expert Validation Questionnaire / Questionnaire Answer Sheet</td>
<td>223</td>
</tr>
<tr>
<td>E</td>
<td>List of Publications</td>
<td>239</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

1.1 Overview

Software testing is a vital phase in software development life cycle preceding the software maintenance. Software testing has increased colossal significance in the present competitive world of innovation, complexity and challenging age of which software is expected to be more efficient and reliable (Kassab et al., 2016). Software testing activities are carried out throughout the software development lifecycle (SDLC) that involves several phases towards the end of the test summary (Spillner et al., 2014). In general, software testing can be divided into four categories; unit testing, integration testing, system testing and acceptance testing (Jorgensen, 2014). As testing is an important platform to ensure software quality and conformity involving many staff and documentations, it is quite hard to effectively manage these activities at a time (Naik and Tripathy, 2011). Test management is dedicatedly engaged to manage all these activities and to find ways to reduce the complexities.

Test management is the process of organizing testing and validating the software. Effective test management is a vital part of developing high quality software product (Kukreja et al., 2015). Through well-planned and well-managed testing processes, the team can ensure that they are producing the high quality software. The team is led by test manager who has the responsibility to manage risk, reviews, assessments and audits. The function of a software test manager is to
effectively and efficiently lead the testing team. To fulfil this role, the leader must comprehend the order of testing and how to successfully execute a testing procedure while satisfying the customary administration part of a manager (Shuja and Krebs, 2007). The role includes quality and test advocacy, asset arranging and administration, and determination of issues that block the test effort.

One of the main challenges in test management is to manage software documentations. Documentation in software engineering is an artefact with the purpose to share the information of which systems it belongs to. Test documentation is identified as the pivotal point as stated by the IEEE829:2010 in order to manage and to report test contents (Sidek, Noraziah and Wahab, 2011). In other perspective, test maturity model takes documentation as an important measure to associate test management to software test process improvement (Van Veenendaal and Cannegieter, 2010). In other words, for test management to remain useful throughout the phases with acceptable maintenance features built-in is to adopt a good documentation model.

One of the activities involved in test management is traceability. Traceability is the ability of linking various artefacts in software development life cycle in forward and backward way (Schwarz, 2012). In the test management, traceability is used to track the bugs back to the corresponding version of requirements. Traceability has been proven to increase the effectiveness or the efficiency of test management.
1.2 Background of the Problem

Nowadays software is becoming more complex. It consists of diverse components with distributed locations, complex algorithms, on varieties of platforms, many sub-contractors with different kind of development methodologies. Complexity brings the fact that no software parts are indistinguishable. A software can be considered as good and high quality if it has a vigorous software testing. Software testing starts as early as software development begins with an enormous testing activities (Parizi et al., 2014). These activities in software testing need to be planned and managed properly; especially the defects or bugs are found during testing. Each of the defects found needs to be traced to the corresponding requirements. This practice is called software traceability.

Traceability is a vital part of software development and maintenance and broadly recognized as a key to quality of the software (Zhang et al., 2016). It is used to capture the link between software artefacts. It is required for the development of safety-critical systems such as in domain of an aerospace (ISO12207, DO-178B), railway (EN50128) and etcetera (Bouillon et al., 2013). In addition, several international quality standards recommended traceability such as IEEE 1291, ISO 9000ff, ISO 15504 and SEI, CMM/CMMI (Wiederseiner et al., 2011).

Currently there are many researchers working on traceability. This is due to the arise of many problem in the industries (Mustafa and Labiche, 2015). Though traceability is proven to be having great impact on software project, there are still a lot of problems such as it is an error prone and time consuming (Marques et al., 2015b), cost-intensive (Maro et al., 2016; Regan et al., 2012), laborious (Shao et al., 2013; Kamalabalan et al., 2015), ad-hoc traceability without strategy (Bouillon, Mäder and Philippow, 2013) and difficult (Regan et al., 2012). There are a few researches on traceability regarding to testing artefacts such as unit testing and class (Qusef et al., 2010), test artefacts and code (Wiederseiner et al., 2011), test cases
and requirements (Noack et al., 2014), bugs and test cases (Kaushik et al., 2011), design and test (Lormans and van Deursen, 2009). Although studies have shown an increase in testing traceability, the research focuses on test result, bugs and test cases is still vague (Garousi, Eskandar and Herkiloğlu, 2016). Research has revealed that poor traceability can be an essential contributing factor to software project failure (Parizi et al., 2014). Though, notwithstanding the available commercial tools to support traceability, the actual practice of traceability remains poorly documented (Cleland-Huang et al., 2012; Maro et al., 2016).

A poorly documented traceability would jeopardize the quality of the software product especially in the critical-safety system. Software engineers depend on system documentation as a guide in comprehension of the practical, architectural design, and the usage of subtle elements of complex applications. Software engineers are compelled to depend exclusively on source code when the documentation does not exist. This is a failure-prone process and a time consuming (Roth et al., 2013), particularly when one considers the amount of information adaptation and domain mapping that is required to comprehend the architecture of a multi-function software system. There are various inadequacies in current project documentation methods (de Graaf et al., 2016). Since the initial days of software development some of these insufficiencies have existed, for example the absence of consistency between the source code and documentations. Other deficiencies have only recently become apparent as vital issues, such as the intricacy in incorporating existing documentations with newly created artefacts (Herwig, 2014). Numerous studies have demonstrated that documentation regularly experiences the accompanying issues:

(i) Nonexistent or of low quality (Alaranta and Betz, 2012; McBurney, 2015)
(ii) Out-dated (McBurney, 2015; Garousi et al., 2013; Satish and Anand, 2016)
(iii) Over abundant and without a definite objectives / incomplete (Parnas, 2011; Dautovic, 2011)
Difficult to access and manage (for instance when the records are scattered on different computers or in distinctive format: diagrams and text) (Choudhury and Thushara, 2014)

Difficult to trace / Lack of traceability (Satish and Anand, 2016; Plosch, Dautovic and Saft, 2014)

The key point solution to the above problems is not the documentation itself, but how to manage the documentation. One or more types of documentation may be made available at each testing phase. The contents of document may reflect some duplication while others are disintegrated that make it difficult for test manager to access, update and control the visibility of current status of testing (Khan and Mattsson, 2012). Currently many researchers have been working on the software documentation however very few are working on the importance of test documentation as a way forward to support test management (Donald, 2013).

Despite this, test documentation is not given due respect by many testers (Andrade et al., 2013). Test documentation is treated as a time consuming task that not many people would like to get involved with. Some organisations give less attention on documentation with reason being the lack of staffing (Khan and Mattsson, 2012). Worse, the distribution of man power allocated to testing activities is not justified in that it is far less than the allocation assigned to the development activities (Treude, Robillard and Dagenais, 2015). This gives more strong reason to why there is a need to have a special emphasis on the need of test documentations and the way to manage them.

Based on the evidence mention, there are fewer endeavours done to manage document traceability in software testing artefacts. Hence, the need to develop new traceability model that support test management is crucial.
1.3 **Statement of the Problem**

There is a need to establish integration amongst documentation such that all can be made accessible and easy to manage. Secondly as different organization may adopt different test documentation, it is necessary to make a survey to understand the most relevant information that is practically used and adopted by the industries. Thirdly, the existing software test documentations are difficult to manage. Thus, there is a need to propose a special mechanism or model to manage software testing documentation in integration. The key solution to above problems is to establish an effective traceability model to support software testing documentations.

This research investigates the need for customized software testing documentation and formulates a software traceability model to support documentation in software testing. The main research question is “*How to design and implement an effective software engineering documentation model based on Software Engineering Standards using traceability model to support Test Management?*”

The sub questions of the main research questions are as follows:

(i) **RQ1**: Why the existing software engineering documentation are not fully adopted by test management and why the existing traceability model still not able to manage the link between the artefacts?

(ii) **RQ2**: What is the effective way to help test management in maintaining a software traceability within a software engineering documentation?

(iii) **RQ3**: How to provide traceability links between artefacts that will support test management?

(iv) **RQ4**: How to evaluate the usability of the proposed model to support test management at some significant degree?
1.4 Objectives of the Study

The research objectives are mentioned based on the problem statement, are as follows:

(i) To study and investigate current issues in software traceability associated to software documentation and test management.
(ii) To formulate a new traceability model that integrates all software engineering artefacts within a repository to support test management.
(iii) To design and develop the prototype of the proposed document-based traceability model.
(iv) To evaluate the effectiveness and the efficiency of the proposed model.

1.5 Scope of the Research

The scope of this study covers the following:

(i) This research focuses on traceability for software testing and its associated components. This will involve the study on system level of software testing (unit, integration, system, and acceptance) but not on types of testing (example – smoke, security, performance, regression, compliance etc.)
(ii) The testing documents will be used are Software Test Description (STD) and Software Test Result (STR). No other testing documents will be used.
(iii) Software engineering documents will be used besides software testing documents are Software Requirements Specifications (SRS), Software Design Document (SDD) and source code.
(iv) This is not a bug tracking system. It just uses documentation to highlight the bugs inside the document.
1.6 Significance of the Study

Requirement traceability has been shown to give numerous advantages to organization that make utilization of traceability methods. This is the reason traceability is an imperative part of numerous standards for software development, such as the CMMI, ISO 9001:2000 and ISO/IEC 15504/SPICE (Gotel et al., 2012). Disregarding the advantages that traceability offers to the software engineering industries, its practice confronts numerous difficulties (Kannenberg and Saiedian, 2009; Cleland-Huang et al., 2014). These difficulties can be distinguished under the zones of cost in terms of endeavor and time, the trouble of keeping up traceability through change, tool support, distinctive perspective focuses on traceability by diverse stakeholders, hierarchical issues and legislative issues, and poor documentation.

On the other hand, documentation plays a vital role in software development and maintenance. Typical software system documentation consists of different type of artefacts, ranging from source code, requirements, architecture design, testing and many more. Good software documentation provides multiple views of a system at different abstraction level and using different formats. As the quantity and variety of information about software system develops, so does the requirement for supporting consistency and traceability among distinctive levels of abstraction for engineers (Nair et al., 2013).

A survey conducted by (Bouillon, Mäder and Philippow, 2013; Mustafa and Labiche, 2017) shows that traceability between requirements and others artefacts (especially testing) was rarely maintained in practice. Meanwhile, research conducted by (Regan et al., 2012) indicates the needs of documentation in practice, and the tools and technologies used to maintain, verify and validate such documents.

Clearly, traceability is very important to trace the link between artefacts involved in software development and maintenance of a software system.
1.7 Thesis Outline

This thesis discusses the issues concerning to traceability that relate to testing artefacts which support test management. It highlights the problems and limitation of software documentation, test management, traceability and the similarity link between them. This thesis is organized as follows:

Chapter 2: Discusses in general about software testing, followed by test management and it approaches and issues. This chapter discusses about software documentation and the problems/issues. This chapter highlights the traceability approaches, traceability models and issues. A comparison study was tabulated and identifies the limitations and issues.

Chapter 3: Describes the research methodology in this research. It explains the resign design, procedure and activities which are used in this research. This chapter also discusses on the evaluation method, instrumentation, case study, assumptions and limitation that have been adopted and observed in this research.

Chapter 4: Presents a conceptual of the proposed model. It also describes the detailed component of the proposed model including the architecture and the process. This chapter explains the development of the proposed model in the UML notation.

Chapter 5: Elucidates the evaluation of the proposed model in terms of effectiveness, efficiency and satisfactory. The quantitative and qualitative method is apply; feature analysis, precision and recall, and expert validation. The results are based on customer perception and metric calculation.
Chapter 6: This chapter concludes the research by describing the research achievement and contributions. The last part explains the limitations and suggestions for future works.
REFERENCES


