دار المنظومة
DAR ALMANDUMAH
الرواد في قواعد المعلومات العربية

العنوان:	Design and evaluation of mutation operators for the asmetal language
المؤلف الرئيسي:	Al Krarha, Osama J.
مؤلفين آخرين:	Hassine, Jamel Aldine(Advisor)
التاريخ الميلادي:	2014
موقع:	الظهران
الصفحات:	1 - 177
رقم MD:	738586
نوع المحتوى:	رسائل جامعية
اللغة:	English
الدرجة العلمية:	رسالة ماجستير
الجامعة:	جامعة الملك فهد للبترول والمعادن
الكلية:	عمادة الدراسات العليا
الدولة:	السعودية
قواعد المعلومات:	Dissertations
مواضيع:	لغات البرمجة، مشغلات الطفرة، هندسة البرمجيات
رابط:	https://search.mandumah.com/Record/738586

© 2019 دار المنظومة. جميع الحقوق محفوظة.
هذه المادة متاحة بناء على الإتفاق الموقع مع أصحاب حقوق النشر، علما أن جميع حقوق النشر محفوظة. يمكنك تحميل أو طباعة هذه المادة للاستخدام الشخصي فقط،
ويمنع النسخ أو التحويل أو النشر عبر أي وسيلة (مثل مواقع الانترنت أو البريد الالكتروني) دون تصريح خطي من أصحاب حقوق النشر أو دار المنظومة.

المنارة للاستشارات
www.manaraa.com

ABSTRACT

Full Name	: Osama Jamil AlKrarha
Thesis Title	: Design and Evaluation of Mutation Operators for AsmetaL Language
Major Field	: Master of Science in Software Engineering
Date of Degree	: May, 2014

Abstract State Machines (ASMs) have been introduced by Gurevich in 1984. Abstract State Machines aim to bridge the gap between informal and formal descriptions by transforming informal specifications to clear and concise specifications. ASM Models are simple, concise, and executable. In addition, they support various levels of abstraction, and provide a well-defined refinement models. ASMs support concurrent and non-deterministic specifications. Several ASM-based languages were proposed to develop and validate Abstract State Machines specifications. Asmeta is an interoperable and integrated framework that provides a standardized infrastructure that serves different specific domain tools and languages. Mutation testing is fault-based testing technique aims to assess the adequacy of test suites by introducing errors into program code to reveal the seeded errors. This thesis proposes a mutation based approach to test ASM specifications. A set of mutation operators were designed for AsmetaL language. The proposed AsmetaL-based operators are analyzed and evaluated empirically using several case studies. Furthermore, the proposed set of operators have been implemented in MuAsmetaL, an AsmetaL mutation testing tool, allowing for validation and execution of mutants, as well as the generation of related statistics. As an application of the proposed approach, test suites generated using ATGT, an AsmetaL compatible testing tool implementing various coverage criteria, were assessed. Mutation testing is known for its high computation cost. In this thesis, both selective and random mutation were applied to AsmetaL mutants resulting in substantial gains in terms of effectiveness and cost savings.

ملخص الرسالة

الاسم الكامل: أسامة جميل القرارعة

عنوان الرسالة: تصميم وتقييم مشغلات الطفرة للغة AsmetaL

التخصص: درجة الماجستير في هندسة البرمجيات

تاريخ الدرجة العلمية: مايو، 2014

استحدثت آلات الحالة المجردة (ASM) بواسطة جورفيتش في عام 1984. وتهدف آلات الحالة المجردة لسد الفجوة بين المواصفات غير الرسمية والرسمية من خلال تحويل المواصفات غير الرسمية لمواصفات رسمية واضحة وموجزة. وتعتبر نماذج ASM بسيطة وموجزة، وقابلة للتنفيذ. بالإضافة إلى أنها تدعم مستويات مختلفة من التجريد، وتوفر نماذج صقل واضحة المعالم. وتدعم ASMs كل من المواصفات المتزامنة وغير القطعية. وقد تم اقتراح عدة لغات على أساس ASM للتطوير والتحقق من صحة مواصفات آلات الحالة المجردة. Asmeta هي عبارة عن إطار للتشغيل المتبادل و المتكامل والتي توفر بنية تحتية موحدة تخدم مختلف لغات وأدوات مجال معين. ويعد اختبار الطفرة تقنية تهدف لتقييم مدى ملاءمة مجموعات الاختبار من خلال تعمد إدخال أخطاء في التعليمات البرمجية للبرنامج وذلك من أجل تقييم مدى قدرة مجموعة الاختبار الكشف عن الأخطاء التى تم إدخالها آنفا. وتقترح هذه الرسالة نهج اختبار الطفرة يستند على تقنية المواصفات ASM. وفي هذه الرسالة، تم تصمم مجموعة من مشغلات الطفرة للغة AsmetaL. وتم تحليل وتقييم هذه المشغلات تجريبيا باستخدام عدة دراسات حالة. وعلاوة على ذلك، فإن مجموعة المشغلات المقترحة تم تنفيذها بواسطة MuAsmetaL، والتي تعتبر أداة لإجراء اختبار الطفرة للغة AsmetaL، مما يسمح للتحقق من صحة وتنفيذ الطفرات، فضلا عن توليد الإحصاءات ذات الصلة. وكتطبيق للنهج المقترح، تم تولد مجموعات اختبار باستخدام أداة ATGT المتوافقة مع لغة AsmetaL بناء على معايير التغطية المختلفة، وجرى تقييمها. ومن المعروف عن اختبار الطفرة أنه ذا تكلفة حسابية عالية. وفي هذه الرسالة، تم تطبيق كل من الطفرة الانتقائية والعشوائية للغة AsmetaL مما أدى لنتائج ايجابية من حيث الفعالية وخفض التكلفة الحسابية.

دار المنظومة
DAR ALMANDUMAH
الرواد في قواعد المعلومات العربية

العنوان:	Design and evaluation of mutation operators for the asmetal language
المؤلف الرئيسـي:	Al Krarha, Osama J.
مؤلفين آخرين:	Hassine, Jamel Aldine(Advisor)
التاريخ الميلادي:	2014
موقع:	الظهران
الصفحات:	1 - 177
رقم MD:	738586
نوع المحتوى:	رسائل جامعية
اللغة:	English
الدرجة العلمية:	رسالة ماجستير
الجامعة:	جامعة الملك فهد للبترول والمعادن
الكلية:	عمادة الدراسات العليا
الدولة:	السعودية
قواعد المعلومات:	Dissertations
مواضيع:	لغات البرمجة، مشغلات الطفرة، هندسة البرمجيات
رابط:	https://search.mandumah.com/Record/738586

© 2019 دار المنظومة. جميع الحقوق محفوظة.
هذه المادة متاحة بناء على الإتفاق الموقع مع أصحاب حقوق النشر، علما أن جميع حقوق النشر محفوظة. يمكنك تحميل أو طباعة هذه المادة للاستخدام الشخصي فقط،
ويمنع النسخ أو التحويل أو النشر عبر أي وسيلة (مثل مواقع الانترنت أو البريد الالكتروني) دون تصريح خطي من أصحاب حقوق النشر أو دار المنظومة.

المنارة للاستشارات
www.manaraa.com

TABLE OF CONTENTS

ACKNOWLEDGMENTS V

TABLE OF CONTENTS VI

LIST OF TABLES XI

LIST OF FIGURES XIII

LIST OF ABBREVIATIONS XV

ABSTRACT XIX

ملخص الرسالة XXI

CHAPTER 1 INTRODUCTION 1

1.1 Motivation 1

1.2 Problem Statement 2

1.3 Research Hypothesis 3

1.4 Thesis Approach 3

1.5 Thesis Contributions 4

1.5.1 Contribution 1: Design and Evaluation of Mutation Operators for the AsmetaL Language .. 4

1.5.2 Contribution 2: Empirical Evaluation of the Proposed Approach 5

1.5.3 Contribution 3: Development of MuAsmetaL 5

1.5.4 Contribution 4: Investigation of Cost Reduction Techniques in the ASM Context 6

1.6 Issues not Addressed in this Thesis 6

1.7 Thesis Outline 7

CHAPTER 2 BASIC DEFINITIONS AND NOTATIONS 9

2.1 Abstract State Machines 9

2.1.1 ASM Thesis....................9

2.1.2 ASM in a Nutshell....................10

2.1.3 ASM Languages....................12

1. AsmL (Abstract State Machine Language) [10]....................13

2. CoreASM [12]....................13

3. ASM Meta-model [14]....................14

3.1. Asmeta [16]....................15

3.2. AsmM [17]....................15

4. ASM SL and ASM Workbench [18]....................15

5. Comparison of ASM environments....................16

2.1.4 AsmetaL....................17

2.1.5 AsmetaL Tools....................27

2.2 Mutation Testing....................28

2.2.1 Mutation Score....................30

2.2.2 Equivalency Analysis Techniques....................31

2.2.3 Reduction Techniques....................32

2.3 Chapter Summary....................38

CHAPTER 3 TESTING ABSTRACT STATE MACHINES: STATE OF THE ART 39

3.1 Testing Abstract State Machines....................39

3.1.1 Generation of Finite State Machines (FSM) from ASM....................39

3.1.2 Conformance Testing....................41

3.1.3 Coverage Criteria....................44

3.1.4 Model Checking....................45

3.1.5 ASMs Test Case Generation....................46

3.2 Mutation Testing of Formal Specifications....................47

3.3 Chapter Summary 48

CHAPTER 4 ASMETAL MUTATION TESTING APPROACH 50

4.1 AsmetaL Mutation Testing Approach 50

4.1.1 Design of mutation operators 52

4.1.2 AsmetaL mutation tool design and implementation 54

4.1.3 Empirical evaluation of the proposed approach 54

4.1.4 Selective mutation 55

4.2 AsmetaL Mutation Operators 55

4.2.1 Function mutation operators 56

4.2.2 Rule mutation operators 56

4.2.3 Term mutation operators 64

4.2.4 Invariant mutation operators 68

4.2.5 Initialization mutation operators 68

4.3 Generation of Test Cases 70

4.4 Analysis of the proposed operators 73

4.5 Chapter Summary 77

CHAPTER 5 MUASMETAL: AN ASMETAL MUTATION EXPERIMENTAL TOOL 78

5.1 Tool Requirements 78

5.2 MuAsmetaL Architecture 79

5.3 MuAsmetaL in Practice 83

5.4 Benchmarking the MuAsmetaL tool 99

5.5 MuAsmetaL Limitation 101

5.6 Chapter Summary 101

CHAPTER 6 EMPIRICAL EVALUATION OF THE ASMETAL-BASED MUTATION OPERATORS 102

6.1 Description of the AsmetaL Case Studies 102

6.1.1 Case Study 1: ferrymanSimulator Specification 103

6.1.2 Case Study 2: railroadGate Specification 104

6.1.3 Case Study 3: sluiceGateGround Specification 106

6.1.4 Case Study 4: cruiseControl Specification 108

6.1.5 Case Study 5: AdvancedClock Specification 109

6.1.6 Case Study 6: AdvancedClock2 Specification 111

6.1.7 Case Study 7: fattoriale Specification 112

6.2 ATGT Test Criteria Comparison using Mutation Testing 113

6.2.1 CruiseControl Specification 113

6.2.2 RailroadGate Specification 117

6.2.3 SluiceGateGround Specification 120

6.2.4 Results Summary 122

6.3 Chapter Summary 123

CHAPTER 7 APPLICATION OF COST REDUCTION TECHNIQUES TO ASMETAL MUTATION TESTING 125

7.1 Introduction 125

7.2 Evaluation Criteria of the Mutation Operators Cost Reduction Techniques 127

7.2.1 Effectiveness 127

7.2.2 Cost Saving 128

7.2.3 Stability 129

7.3 N-selective-based Mutation 130

7.4 Random-based Selective Mutation 130

7.5 Applying Cost Reduction Techniques to Case Studies 130

7.5.1 Case Study 1: ferrymanSimulator Specification 130

7.5.2 Case Study 2: railroadGate Specification 133

7.5.3 Case Study 3: sluiceGateGround Specification 136

7.5.4 Case Study 4: cruiseControl Specification 138

7.5.5 Case Study 5: AdvancedClock Specification 141

7.5.6 Case Study 6: AdvancedClock2 Specification 141

7.5.7 Case Study 7: fattoriale Specification 141

7.5.8 Results Summary 144

7.6 Overall Operator-Based Selection Mutation 147

7.7 General Discussion 150

7.8 Threats to Validity 154

CHAPTER 8 CONCLUSIONS AND FUTURE WORK 157

8.1 Hypothesis of the Thesis 157

8.2 Thesis Contributions of the Thesis 159

8.2.1 Contribution 1: Design and Evaluation of Mutation Operators for the AsmetaL language 159

8.2.2 Contribution 2: Empirical Evaluation of the Proposed Approach 159

8.2.3 Contribution 3: Development of MuAsmetaL 160

8.2.4 Contribution 4: Investigation of Cost Reduction Techniques in the ASM Context 160

8.3 Future Work 161

REFERENCES 163

VITAE 177

LIST OF TABLES

Table 1: Comparison of ASM Programs 17
Table 2: AsmetaL simple example 18
Table 3: concrete domain signature 19
Table 4: enumerator domain signature 19
Table 5: abstract domain signature 19
Table 6: basic domain signature 19
Table 7: static function signature 20
Table 8: dynamic function signature 20
Table 9: derived function signature 20
Table 10: macro rule declaration 21
Table 11: turbo rule declaration 21
Table 12: main rule declaration 21
Table 13: AsmetaL rule structures 22
Table 14: FTP operator example 56
Table 15: turbo rule operator's examples 60
Table 16: Traditional rule mutation operators examples 65
Table 17: Asmetal term operators examples 67
Table 18: ICR and IDD operators examples 68
Table 19: AsmetaL initialization operators examples 69
Table 20: AVaLLA test case example (.test) 71
Table 21: AVaLLA test case results generated by AsmetaV 71
Table 22: The upper bound for the number of generatred mutants per operator 73
Table 23: Time spent to generate and validate mutants per case study 99
Table 24: Time spent to execute test cases and generate reports per case study 100
Table 25: Case studies summary 102
Table 26: ferrymanSimulator specification mutation results 103
Table 27: railroadGate specification mutation testing results 105
Table 28: sluiceGateGround specification mutation testing results 107
Table 29: cruiseControl specification mutation testing results 108
Table 30: AdvancedClock specification mutation testing results 110
Table 31: AdvancedClock2 specification mutation testing results 111
Table 32: fattoriale specification mutation testing results 112
Table 33: CruiseControl specification mutation testing based on update rule coverage. 113
Table 34: CruiseControl specification mutation testing based basic rule coverage 114
Table 35: CruiseControl specification mutation testing based MCDC coverage 114
Table 36: CruiseControl specification mutation testing based fault coverage 114
Table 37: CruiseControl specification mutation testing based pair-wise coverage 115
Table 38: RailroadGate specification mutation testing based update rule coverage 117
Table 39: RailroadGate specification mutation testing based basic rule coverage 117

Table 40: RailroadGate specification mutation testing based basic MCDC coverage.... 117
Table 41: RailroadGate specification mutation testing based fault coverage 118
Table 42: RailroadGate specification mutation testing based pair-wise coverage.......... 118
Table 43: RailroadGate specification mutation testing based three-wise coverage 118
Table 44: SluiceGateGround specification mutation testing based update rule coverage 120
Table 45: SluiceGateGround specification mutation testing based basic rule coverage. 120
Table 46: SluiceGateGround specification mutation testing based MCDC coverage..... 120
Table 47: SluiceGateGround specification mutation testing based fault coverage 121
Table 48: SluiceGateGround specification mutation testing based pair-wise coverage . 121
Table 49: 2, 4, 6-N-selective results for the case studies 144
Table 50: 10%, 25%, 50% random selection results for the case studies........................ 146
Table 51: Ranking dominant operators (All case studies).. 148
Table 52: Overall 2-Operators Selection mutation.. 149
Table 53: Overall 4-Operators Selection mutation.. 149
Table 54: Overall 6-Operators Selection mutation.. 150

LIST OF FIGURES

Figure 1: Thesis tasks workflow 4
Figure 2: ASM Workbench model verification process 16
Figure 3: AsmetaL basic structure 18
Figure 4: AsmetaL rule types 24
Figure 5: AsmetaL function types 25
Figure 6: AsmetaL domain types 26
Figure 7: Typical procedure of mutation testing 30
Figure 8: Conformance Testing Concepts 42
Figure 9: AsmetaL mutation testing procedure 52
Figure 10: Mutant generation process 54
Figure 11: MuAsmetaL Structure 80
Figure 12: Example of an AsmetaL Tree 82
Figure 13: Absolute value AsmetaL specification 84
Figure 14: Creating new AsmetaL specification using MuAsmetaL 85
Figure 15: Editing existing AsmetaL specification using MuAsmetaL 86
Figure 16: Visualizing ASMetaLTree using MuAsmetaL 87
Figure 17: Statistical information about AsmetaL Specification using MuAsmetaL 88
Figure 18: MuAsmetaL mutation generation interface 89
Figure 19: MuAsmetaL mutation generation summary 90
Figure 20: MuAsmetaL handles manual input from the user 90
Figure 21: AsmetaL specification correctness validation and syntactic equivalency validation 91
Figure 22: MuAsmetaL mutants' viewer 92
Figure 23: Import AVaLLA test cases using MuAsmetaL 92
Figure 24: Viewing/Ordering test cases using MuAsmetaL 93
Figure 25: Running test cases against original Specification using MuAsmetaL to obtain test oracles 93
Figure 26: Test case results 94
Figure 27: MuAsmetaL custom testing 95
Figure 28: Running test cases against mutants 95
Figure 29: Report file (CSV) generated by MuAsmetaL 96
Figure 30: Simulating AsmetaL specification using MuAsmetaL 96
Figure 31: MuAsmetaL mutation testing results 1 97
Figure 32: MuAsmetaL mutation testing results 2 98
Figure 33: MuAsmetaL mutation testing results 3 98
Figure 34: MuAsmetaL mutation testing results 4 99
Figure 35: ferrymanSimulator specification mutation testing results 104
Figure 36: railroadGate specification mutation testing results 106
Figure 37: sluiceGateGround specification mutation testing results 107

Figure 38: cruiseControl specification mutation testing results 109
Figure 39: AdvancedClock specification mutation testing results 110
Figure 40: AdvancedClock2 specification mutation testing results 112
Figure 41: fattoriale specification mutation testing results 113
Figure 42: Overall deference of mutation testing over different testing criteria for CruiseControl Specification 116
Figure 43: Overall deference of mutation testing over different testing criteria for RailroadGate Specification... 119
Figure 44: Overall deference of mutation testing over different testing criteria for SluiceGateGround Specification 122
Figure 45: Selective mutation reduction procedure... 128
Figure 46: ferrymanSimulator specification random selection (10%) 131
Figure 47: ferrymanSimulator specification random selection (25%) 132
Figure 48: ferrymanSimulator specification random selection (50%) 133
Figure 49: railroadGate specification random selection (10%)....................................... 134
Figure 50: railroadGate specification random selection (25%)....................................... 135
Figure 51: railroadGate specification random selection (50%)....................................... 135
Figure 52: sluiceGateGround specification random selection (10%).............................. 137
Figure 53: sluiceGateGround specification random selection (25%).............................. 137
Figure 54: sluiceGateGround specification random selection (50%).............................. 138
Figure 55: cruiseControl specification random selection (10%) 139
Figure 56: cruiseControl specification random selection (25%) 140
Figure 57: cruiseControl specification random selection (50%) 141
Figure 58: fattoriale specification random selection (10%) 143
Figure 59: fattoriale specification random selection (25%) 143
Figure 60: fattoriale specification random selection (50%) 144

LIST OF ABBREVIATIONS

ABS	:	Absolute Value Operator
AOR	:	Arithmetic Operator Replacement Operator
ARO	:	Add Rule Operator
ASM	:	Abstract State Machine
ASM SL	:	Abstract State Machine Standard Language
AsmetaL	:	Abstract State Machine Meta Model Language
AsmL	:	Abstract State Machine Language
CDoR	:	Choose DoRule Replacement Operator
CDR	:	Choose Domain Replacement Operator
CIR	:	Choose IfNoneRule Replacement Operator
CLI	:	Command Line Interface
CRE	:	Choose Rule Exchange Operator
CRRO	:	Case Rule Replacement Operator
CTM	:	Constant Term Modification Operator
CTR	:	Constant Term Replacement Operator
CTRO	:	Case Term Replacement Operator

DIR	:	Default Initialization Replacement Operator
DSC	:	Delete Switch Case Operator
EBNF	:	Extended Backus–Naur Form
EDR	:	Extend Domain Replacement Operator
EIR	:	Extend ID Replacement Operator
ENF	:	Expression Negation Fault Operator
ERR	:	Else Rule Replacement Operator
ERRO	:	Extend Rule Replacement Operator
ETR	:	Else Term Replacement Operator
FCRP	:	Forall Choose Rules Permutation Operator
FDoR	:	Forall DoRule Replacement Operator
FQTDR	:	Finite Quantification Term Domain Replacement Operator
FQTP	:	Finite Quantification Terms Permutation Operator
FSM	:	Finite State Machine
FTP	:	Function Type Permutation Operator
GUI	:	Graphical User Interface
ICR	:	Invariant Condition Replacement Operator

IDD	:	Invariant Declaration Deletion Operator
IDE	:	Integrated Development Environment
IIP	:	Initialization ID Permutation Operator
ISD	:	Initialization Statement Deletion Operator
LNF	:	Literal Negation Fault
LOR	:	Logical Operator Replacement Operator
LRR	:	Let Rule Replacement Operator
LRVA	:	Let Rule Variable Assignment Operator
LRVR	:	Let Rule Variable Replacement Operator
LTS	:	Label Transition System
MCDC	:	Multiple Condition Coverage
MRR	:	Main Rule Replacement Operator
MS	:	Mutation Score
RGCR	:	Rule Guard Condition Replacement Operator
ROR	:	Relational Operator Replacement Operator
RRO	:	Replace Rule Operator
RTS	:	Rule to Skip Rule Operator

S2PB	:	Sequential to Parallel Block Operator
SBSDL	:	Sequential Block Statement Deletion Operator
SCP	:	Switch Case Permutation Operator
SSM	:	Sequence Rule Order Permutation Operator
SSSC	:	Stuck Switch to Specific Case Operator
STF	:	Stuck at True False Operator
TGCR	:	Term Guard Condition Replacement Operator
TRR	:	Then Rule Replacement Operator
TTR	:	Then Term Replacement Operator
UOI	:	Unary Operator Insertion Operator

دار المنظومة
DAR ALMANDUMAH
الرواد في قواعد المعلومات العربية

العنوان:	Design and evaluation of mutation operators for the asmetal language
المؤلف الرئيسي:	Al Krarha, Osama J.
مؤلفين آخرين:	Hassine, Jamel Aldine(Advisor)
التاريخ الميلادي:	2014
موقع:	الظهران
الصفحات:	1 - 177
رقم MD:	738586
نوع المحتوى:	رسائل جامعية
اللغة:	English
الدرجة العلمية:	رسالة ماجستير
الجامعة:	جامعة الملك فهد للبترول والمعادن
الكلية:	عمادة الدراسات العليا
الدولة:	السعودية
قواعد المعلومات:	Dissertations
مواضيع:	لغات البرمجة، مشغلات الطفرة، هندسة البرمجيات
رابط:	https://search.mandumah.com/Record/738586

© 2019 دار المنظومة. جميع الحقوق محفوظة.
هذه المادة متاحة بناء على الإتفاق الموقع مع أصحاب حقوق النشر، علما أن جميع حقوق النشر محفوظة. يمكنك تحميل أو طباعة هذه المادة للاستخدام الشخصي فقط،
ويمنع النسخ أو التحويل أو النشر عبر أي وسيلة (مثل مواقع الانترنت أو البريد الالكتروني) دون تصريح خطي من أصحاب حقوق النشر أو دار المنظومة.

المنارة للاستشارات

www.manaraa.com

DESIGN AND EVALUATION OF MUTATION OPERATORS FOR THE ASMETAL LANGUAGE

BY

Osama J. AlKrarha

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

Software Engineering

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN- 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Osama Jamil AlKrarha** under the direction his thesis advisor and approved by his thesis committee, has been presented and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN SOFTWARE ENGINEERING.**

Dr. Adel Ahmad
Department Chairman

Dr. Jameleddine Hassine
(Advisor)

Dr. Salam A. Zummo
Dean of Graduate Studies

Dr. Moataz Ahmed
(Member)

1/6/14
Date

عمادة الدراسات العليا
DEANSHIP OF GRADUATE STUDIES

Dr. Sami Zhioua
(Member)

© Osama J. AlKrarha

2014

To My Parents

ACKNOWLEDGMENTS

All praise and thank to almighty Allah (GOD) for giving me the strength, resolve, health, patience, and knowledge to complete this work.

I acknowledge, with deep gratitude and appreciation, the inspiration, encouragement, valuable time and guidance given to me by DR. Jameleddine Hussine, who served as my major advisor and mentor. Thereafter, I am deeply indebted and grateful to Dr. Moataz Ahmad, and Dr. Sami Zhioua, my committee member, for their extensive guidance, continuous support, and personal involvement in all phases of this research. I am also grateful to my director, Dr. Sadiq Sait their constructive guidance, valuable advice and cooperation.

I also would like to express my deepest gratitude to my mother, father, brothers, sisters, and friends, for their emotional and moral support throughout my academic career and also for their love, patience, encouragement and prayers.

Finally, I would like to thank all the colleagues at the CCITR/Research Institute at King Fahd University for their and support during the period this research took place is greatly acknowledged.

TABLE OF CONTENTS

ACKNOWLEDGMENTS V

TABLE OF CONTENTS VI

LIST OF TABLES XI

LIST OF FIGURES XIII

LIST OF ABBREVIATIONS XV

ABSTRACT XIX

ملخص الرسالة XXI

CHAPTER 1 INTRODUCTION 1

1.1 Motivation 1

1.2 Problem Statement 2

1.3 Research Hypothesis 3

1.4 Thesis Approach 3

1.5 Thesis Contributions 4

1.5.1 Contribution 1: Design and Evaluation of Mutation Operators for the AsmetaL Language .. 4

1.5.2 Contribution 2: Empirical Evaluation of the Proposed Approach 5

1.5.3 Contribution 3: Development of MuAsmetaL 5

1.5.4 Contribution 4: Investigation of Cost Reduction Techniques in the ASM Context 6

1.6 Issues not Addressed in this Thesis 6

1.7 Thesis Outline 7

CHAPTER 2 BASIC DEFINITIONS AND NOTATIONS 9

2.1 Abstract State Machines 9

2.1.1 ASM Thesis 9

2.1.2 ASM in a Nutshell 10

2.1.3 ASM Languages 12

1. AsmL (Abstract State Machine Language) [10] 13

2. CoreASM [12] 13

3. ASM Meta-model [14] 14

3.1. Asmeta [16] 15

3.2. AsmM [17] 15

4. ASM SL and ASM Workbench [18] 15

5. Comparison of ASM environments 16

2.1.4 AsmetaL 17

2.1.5 AsmetaL Tools 27

2.2 Mutation Testing 28

2.2.1 Mutation Score 30

2.2.2 Equivalency Analysis Techniques 31

2.2.3 Reduction Techniques 32

2.3 Chapter Summary 38

CHAPTER 3 TESTING ABSTRACT STATE MACHINES: STATE OF THE ART 39

3.1 Testing Abstract State Machines 39

3.1.1 Generation of Finite State Machines (FSM) from ASM 39

3.1.2 Conformance Testing 41

3.1.3 Coverage Criteria 44

3.1.4 Model Checking 45

3.1.5 ASMs Test Case Generation 46

3.2 Mutation Testing of Formal Specifications 47

3.3 Chapter Summary 48

CHAPTER 4 ASMETAL MUTATION TESTING APPROACH 50

4.1 AsmetaL Mutation Testing Approach 50

4.1.1 Design of mutation operators 52

4.1.2 AsmetaL mutation tool design and implementation 54

4.1.3 Empirical evaluation of the proposed approach 54

4.1.4 Selective mutation 55

4.2 AsmetaL Mutation Operators 55

4.2.1 Function mutation operators 56

4.2.2 Rule mutation operators 56

4.2.3 Term mutation operators 64

4.2.4 Invariant mutation operators 68

4.2.5 Initialization mutation operators 68

4.3 Generation of Test Cases 70

4.4 Analysis of the proposed operators 73

4.5 Chapter Summary 77

CHAPTER 5 MUASMETAL: AN ASMETAL MUTATION EXPERIMENTAL TOOL 78

5.1 Tool Requirements 78

5.2 MuAsmetaL Architecture 79

5.3 MuAsmetaL in Practice 83

5.4 Benchmarking the MuAsmetaL tool 99

5.5 MuAsmetaL Limitation 101

5.6 Chapter Summary 101

CHAPTER 6 EMPIRICAL EVALUATION OF THE ASMETAL-BASED MUTATION OPERATORS 102

6.1 Description of the AsmetaL Case Studies 102

6.1.1 Case Study 1: ferrymanSimulator Specification 103

6.1.2 Case Study 2: railroadGate Specification 104

6.1.3 Case Study 3: sluiceGateGround Specification 106

6.1.4 Case Study 4: cruiseControl Specification 108

6.1.5 Case Study 5: AdvancedClock Specification 109

6.1.6 Case Study 6: AdvancedClock2 Specification 111

6.1.7 Case Study 7: fattoriale Specification 112

6.2 ATGT Test Criteria Comparison using Mutation Testing 113

6.2.1 CruiseControl Specification 113

6.2.2 RailroadGate Specification 117

6.2.3 SluiceGateGround Specification 120

6.2.4 Results Summary 122

6.3 Chapter Summary 123

CHAPTER 7 APPLICATION OF COST REDUCTION TECHNIQUES TO ASMETAL MUTATION TESTING 125

7.1 Introduction 125

7.2 Evaluation Criteria of the Mutation Operators Cost Reduction Techniques 127

7.2.1 Effectiveness 127

7.2.2 Cost Saving 128

7.2.3 Stability 129

7.3 N-selective-based Mutation 130

7.4 Random-based Selective Mutation 130

7.5 Applying Cost Reduction Techniques to Case Studies 130

7.5.1 Case Study 1: ferrymanSimulator Specification 130

7.5.2 Case Study 2: railroadGate Specification 133

7.5.3 Case Study 3: sluiceGateGround Specification 136

7.5.4 Case Study 4: cruiseControl Specification 138

7.5.5 Case Study 5: AdvancedClock Specification 141

7.5.6 Case Study 6: AdvancedClock2 Specification 141

7.5.7 Case Study 7: fattoriale Specification 141

7.5.8 Results Summary 144

7.6 Overall Operator-Based Selection Mutation 147

7.7 General Discussion 150

7.8 Threats to Validity 154

CHAPTER 8 CONCLUSIONS AND FUTURE WORK 157

8.1 Hypothesis of the Thesis 157

8.2 Thesis Contributions of the Thesis 159

8.2.1 Contribution 1: Design and Evaluation of Mutation Operators for the AsmetaL language 159

8.2.2 Contribution 2: Empirical Evaluation of the Proposed Approach 159

8.2.3 Contribution 3: Development of MuAsmetaL 160

8.2.4 Contribution 4: Investigation of Cost Reduction Techniques in the ASM Context 160

8.3 Future Work 161

REFERENCES 163

VITAE 177

LIST OF TABLES

Table 1: Comparison of ASM Programs 17
Table 2: AsmetaL simple example 18
Table 3: concrete domain signature 19
Table 4: enumerator domain signature 19
Table 5: abstract domain signature 19
Table 6: basic domain signature 19
Table 7: static function signature 20
Table 8: dynamic function signature 20
Table 9: derived function signature 20
Table 10: macro rule declaration 21
Table 11: turbo rule declaration 21
Table 12: main rule declaration 21
Table 13: AsmetaL rule structures 22
Table 14: FTP operator example 56
Table 15: turbo rule operator's examples 60
Table 16: Traditional rule mutation operators examples 65
Table 17: Asmetal term operators examples 67
Table 18: ICR and IDD operators examples 68
Table 19: AsmetaL initialization operators examples 69
Table 20: AVaLLA test case example (.test) 71
Table 21: AVaLLA test case results generated by AsmetaV 71
Table 22: The upper bound for the number of generatred mutants per operator 73
Table 23: Time spent to generate and validate mutants per case study 99
Table 24: Time spent to execute test cases and generate reports per case study 100
Table 25: Case studies summary 102
Table 26: ferrymanSimulator specification mutation results 103
Table 27: railroadGate specification mutation testing results 105
Table 28: sluiceGateGround specification mutation testing results 107
Table 29: cruiseControl specification mutation testing results 108
Table 30: AdvancedClock specification mutation testing results 110
Table 31: AdvancedClock2 specification mutation testing results 111
Table 32: fattoriale specification mutation testing results 112
Table 33: CruiseControl specification mutation testing based on update rule coverage. 113
Table 34: CruiseControl specification mutation testing based basic rule coverage 114
Table 35: CruiseControl specification mutation testing based MCDC coverage 114
Table 36: CruiseControl specification mutation testing based fault coverage 114
Table 37: CruiseControl specification mutation testing based pair-wise coverage 115
Table 38: RailroadGate specification mutation testing based update rule coverage 117
Table 39: RailroadGate specification mutation testing based basic rule coverage 117

Table 40: RailroadGate specification mutation testing based basic MCDC coverage.... 117
Table 41: RailroadGate specification mutation testing based fault coverage 118
Table 42: RailroadGate specification mutation testing based pair-wise coverage.......... 118
Table 43: RailroadGate specification mutation testing based three-wise coverage 118
Table 44: SluiceGateGround specification mutation testing based update rule coverage 120
Table 45: SluiceGateGround specification mutation testing based basic rule coverage. 120
Table 46: SluiceGateGround specification mutation testing based MCDC coverage..... 120
Table 47: SluiceGateGround specification mutation testing based fault coverage 121
Table 48: SluiceGateGround specification mutation testing based pair-wise coverage . 121
Table 49: 2, 4, 6-N-selective results for the case studies 144
Table 50: 10%, 25%, 50% random selection results for the case studies........................ 146
Table 51: Ranking dominant operators (All case studies)... 148
Table 52: Overall 2-Operators Selection mutation... 149
Table 53: Overall 4-Operators Selection mutation... 149
Table 54: Overall 6-Operators Selection mutation... 150

LIST OF FIGURES

Figure 1: Thesis tasks workflow....4
Figure 2: ASM Workbench model verification process....16
Figure 3: AsmetaL basic structure....18
Figure 4: AsmetaL rule types....24
Figure 5: AsmetaL function types....25
Figure 6: AsmetaL domain types....26
Figure 7: Typical procedure of mutation testing....30
Figure 8: Conformance Testing Concepts....42
Figure 9: AsmetaL mutation testing procedure....52
Figure 10: Mutant generation process....54
Figure 11: MuAsmetaL Structure....80
Figure 12: Example of an AsmetaL Tree....82
Figure 13: Absolute value AsmetaL specification....84
Figure 14: Creating new AsmetaL specification using MuAsmetaL....85
Figure 15: Editing existing AsmetaL specification using MuAsmetaL....86
Figure 16: Visualizing ASMetaLTree using MuAsmetaL....87
Figure 17: Statistical information about AsmetaL Specification using MuAsmetaL....88
Figure 18: MuAsmetaL mutation generation interface....89
Figure 19: MuAsmetaL mutation generation summary....90
Figure 20: MuAsmetaL handles manual input from the user....90
Figure 21: AsmetaL specification correctness validation and syntactic equivalency validation....91
Figure 22: MuAsmetaL mutants' viewer....92
Figure 23: Import AVaLLA test cases using MuAsmetaL....92
Figure 24: Viewing/Ordering test cases using MuAsmetaL....93
Figure 25: Running test cases against original Specification using MuAsmetaL to obtain test oracles....93
Figure 26: Test case results....94
Figure 27: MuAsmetaL custom testing....95
Figure 28: Running test cases against mutants....95
Figure 29: Report file (CSV) generated by MuAsmetaL....96
Figure 30: Simulating AsmetaL specification using MuAsmetaL....96
Figure 31: MuAsmetaL mutation testing results 1....97
Figure 32: MuAsmetaL mutation testing results 2....98
Figure 33: MuAsmetaL mutation testing results 3....98
Figure 34: MuAsmetaL mutation testing results 4....99
Figure 35: ferrymanSimulator specification mutation testing results....104
Figure 36: railroadGate specification mutation testing results....106
Figure 37: sluiceGateGround specification mutation testing results....107

Figure 38: cruiseControl specification mutation testing results 109
Figure 39: AdvancedClock specification mutation testing results 110
Figure 40: AdvancedClock2 specification mutation testing results 112
Figure 41: fattoriale specification mutation testing results 113
Figure 42: Overall deference of mutation testing over different testing criteria for CruiseControl Specification 116
Figure 43: Overall deference of mutation testing over different testing criteria for RailroadGate Specification.. 119
Figure 44: Overall deference of mutation testing over different testing criteria for SluiceGateGround Specification 122
Figure 45: Selective mutation reduction procedure... 128
Figure 46: ferrymanSimulator specification random selection (10%) 131
Figure 47: ferrymanSimulator specification random selection (25%) 132
Figure 48: ferrymanSimulator specification random selection (50%) 133
Figure 49: railroadGate specification random selection (10%)...................................... 134
Figure 50: railroadGate specification random selection (25%)...................................... 135
Figure 51: railroadGate specification random selection (50%)...................................... 135
Figure 52: sluiceGateGround specification random selection (10%)............................ 137
Figure 53: sluiceGateGround specification random selection (25%)............................ 137
Figure 54: sluiceGateGround specification random selection (50%)............................ 138
Figure 55: cruiseControl specification random selection (10%) 139
Figure 56: cruiseControl specification random selection (25%) 140
Figure 57: cruiseControl specification random selection (50%) 141
Figure 58: fattoriale specification random selection (10%) 143
Figure 59: fattoriale specification random selection (25%) 143
Figure 60: fattoriale specification random selection (50%) 144

LIST OF ABBREVIATIONS

ABS	:	Absolute Value Operator
AOR	:	Arithmetic Operator Replacement Operator
ARO	:	Add Rule Operator
ASM	:	Abstract State Machine
ASM SL	:	Abstract State Machine Standard Language
AsmetaL	:	Abstract State Machine Meta Model Language
AsmL	:	Abstract State Machine Language
CDoR	:	Choose DoRule Replacement Operator
CDR	:	Choose Domain Replacement Operator
CIR	:	Choose IfNoneRule Replacement Operator
CLI	:	Command Line Interface
CRE	:	Choose Rule Exchange Operator
CRRO	:	Case Rule Replacement Operator
CTM	:	Constant Term Modification Operator
CTR	:	Constant Term Replacement Operator
CTRO	:	Case Term Replacement Operator

DIR	:	Default Initialization Replacement Operator
DSC	:	Delete Switch Case Operator
EBNF	:	Extended Backus–Naur Form
EDR	:	Extend Domain Replacement Operator
EIR	:	Extend ID Replacement Operator
ENF	:	Expression Negation Fault Operator
ERR	:	Else Rule Replacement Operator
ERRO	:	Extend Rule Replacement Operator
ETR	:	Else Term Replacement Operator
FCRP	:	Forall Choose Rules Permutation Operator
FDoR	:	Forall DoRule Replacement Operator
FQTDR	:	Finite Quantification Term Domain Replacement Operator
FQTP	:	Finite Quantification Terms Permutation Operator
FSM	:	Finite State Machine
FTP	:	Function Type Permutation Operator
GUI	:	Graphical User Interface
ICR	:	Invariant Condition Replacement Operator

IDD	:	Invariant Declaration Deletion Operator
IDE	:	Integrated Development Environment
IIP	:	Initialization ID Permutation Operator
ISD	:	Initialization Statement Deletion Operator
LNF	:	Literal Negation Fault
LOR	:	Logical Operator Replacement Operator
LRR	:	Let Rule Replacement Operator
LRVA	:	Let Rule Variable Assignment Operator
LRVR	:	Let Rule Variable Replacement Operator
LTS	:	Label Transition System
MCDC	:	Multiple Condition Coverage
MRR	:	Main Rule Replacement Operator
MS	:	Mutation Score
RGCR	:	Rule Guard Condition Replacement Operator
ROR	:	Relational Operator Replacement Operator
RRO	:	Replace Rule Operator
RTS	:	Rule to Skip Rule Operator

S2PB	:	Sequential to Parallel Block Operator
SBSDL	:	Sequential Block Statement Deletion Operator
SCP	:	Switch Case Permutation Operator
SSM	:	Sequence Rule Order Permutation Operator
SSSC	:	Stuck Switch to Specific Case Operator
STF	:	Stuck at True False Operator
TGCR	:	Term Guard Condition Replacement Operator
TRR	:	Then Rule Replacement Operator
TTR	:	Then Term Replacement Operator
UOI	:	Unary Operator Insertion Operator

ABSTRACT

Full Name : Osama Jamil AlKrarha

Thesis Title : Design and Evaluation of Mutation Operators for AsmetaL Language

Major Field : Master of Science in Software Engineering

Date of Degree : May, 2014

Abstract State Machines (ASMs) have been introduced by Gurevich in 1984. Abstract State Machines aim to bridge the gap between informal and formal descriptions by transforming informal specifications to clear and concise specifications. ASM Models are simple, concise, and executable. In addition, they support various levels of abstraction, and provide a well-defined refinement models. ASMs support concurrent and non-deterministic specifications. Several ASM-based languages were proposed to develop and validate Abstract State Machines specifications. Asmeta is an interoperable and integrated framework that provides a standardized infrastructure that serves different specific domain tools and languages. Mutation testing is fault-based testing technique aims to assess the adequacy of test suites by introducing errors into program code to reveal the seeded errors. This thesis proposes a mutation based approach to test ASM specifications. A set of mutation operators were designed for AsmetaL language. The proposed AsmetaL-based operators are analyzed and evaluated empirically using several case studies. Furthermore, the proposed set of operators have been implemented in MuAsmetaL, an AsmetaL mutation testing tool, allowing for validation and execution of mutants, as well as the generation of related statistics. As an application of the proposed approach, test suites generated using ATGT, an AsmetaL compatible testing tool implementing various coverage criteria, were assessed. Mutation testing is known for its high computation cost. In this thesis, both selective and random mutation were applied to AsmetaL mutants resulting in substantial gains in terms of effectiveness and cost savings.

ملخص الرسالة

الاسم الكامل: أسامة جميل القرارعة

عنوان الرسالة: تصميم وتقييم مشغلات الطفرة للغة AsmetaL

التخصص: درجة الماجستير في هندسة البرمجيات

تاريخ الدرجة العلمية: مايو، 2014

استحدثت آلات الحالة المجردة (ASM) بواسطة جورفيتش في عام 1984. وتهدف آلات الحالة المجردة لسد الفجوة بين المواصفات غير الرسمية والرسمية من خلال تحويل المواصفات غير الرسمية لمواصفات رسمية واضحة وموجزة. وتعتبر نماذج ASM بسيطة وموجزة، وقابلة للتنفيذ. بالإضافة إلى أنها تدعم مستويات مختلفة من التجريد، وتوفر نماذج صقل واضحة المعالم. وتدعم ASMs كل من المواصفات المتزامنة وغير القطعية. وقد تم اقتراح عدة لغات على أساس ASM للتطوير والتحقق من صحة مواصفات آلات الحالة المجردة. Asmeta هي عبارة عن إطار للتشغيل المتبادل و المتكامل والتي توفر بنية تحتية موحدة تخدم مختلف لغات وأدوات مجال معين. ويعد اختبار الطفرة تقنية تهدف لتقييم مدى ملاءمة مجموعات الاختبار من خلال تعمد إدخال أخطاء في التعليمات البرمجية للبرنامج وذلك من أجل تقييم مدى قدرة مجموعة الاختبار الكشف عن الأخطاء التى تم إدخالها آنفا. وتقترح هذه الرسالة نهج اختبار الطفرة يستند على تقنية المواصفات ASM. وفي هذه الرسالة، تم تصمم مجموعة من مشغلات الطفرة للغة AsmetaL. وتم تحليل وتقييم هذه المشغلات تجريبيا باستخدام عدة دراسات حالة. وعلاوة على ذلك، فإن مجموعة المشغلات المقترحة تم تنفيذها بواسطة MuAsmetaL، والتي تعتبر أداة لإجراء اختبار الطفرة للغة AsmetaL، مما يسمح للتحقق من صحة وتنفيذ الطفرات، فضلا عن توليد الإحصاءات ذات الصلة. وكتطبيق للنهج المقترح، تم تولد مجموعات اختبار باستخدام أداة ATGT المتوافقة مع لغة AsmetaL بناء على معايير التغطية المختلفة، وجرى تقييمها. ومن المعروف عن اختبار الطفرة أنه ذا تكلفة حسابية عالية. وفي هذه الرسالة، تم تطبيق كل من الطفرة الانتقائية والعشوائية للغة AsmetaL مما أدى لنتائج ايجابية من حيث الفعالية وخفض التكلفة الحسابية.

CHAPTER 1

INTRODUCTION

The demand for high quality software has increased in various fields and disciplines. Therefore, it led to an increased focus on the effectiveness of the processes used in the software industry. Software testing is considered one of the most critical processes that lead to software projects success or failure, therefore, software engineers and researchers in this area aim to put more emphasis on the effectiveness of software testing. Software testing spans the entire software life cycle from requirements stage to the maintenance stage. The magnitude of faults can be reduced if they were detected at the early stages.

1.1 Motivation

The typical way to validate unstructured software specifications is through inspection [1], which is usually carried out manually and takes considerable time and effort. In contrast, the usage of formal specifications reduces such an effort and time, while allowing for automated validation. Abstract State Machines (ASMs) [2] is a formal paradigm that has proved its merit in many fields such as software requirements engineering, network protocols engineering, and system engineering. Handling software requirements using Abstract State Machine overcomes the natural language with the following advantages: Simplicity, precise semantics, various levels of abstractions, and executability. In addition, it provides a well-defined validation and verification model. Moreover, ASM Models can be used to generate portions of the implementation.

Mutation testing technique is a fault-based technique that has been successfully used to test various programming and specification languages. This thesis introduces a new ASM-based mutation testing approach to assess the adequacy of ASM test suites.

1.2 Problem Statement

The goal of this research is to develop a mutation testing approach for AsmetaL, an ASM-based language. The proposed approach would allow both practitioners and researchers to assess and improve the adequacy of AsmetaL test suites. The main goal is decomposed into the following sub-goals:

- *Sub-Goal 1:* Definition of a set of mutation operators for AsmetaL as a concrete incarnation of ASM mutation operators.
- *Sub-Goal 2:* Investigation of the applicability of the proposed mutation operators to various case studies.
- *Sub-Goal 3:* Assessment of the effectiveness of the designed operators.
- *Sub-Goal 4:* Investigation the applicability of cost reduction techniques such as selective and random mutation in the context of the AsmetaL language.
- *Sub-Goal5:* Develop an AsmetaL mutation testing tool that allows for validation and execution of mutants and the generation of mutation related statistics.

دار المنظومة
DAR ALMANDUMAH
الرواد في قواعد المعلومات العربية

العنوان:	Design and evaluation of mutation operators for the asmetal language
المؤلف الرئيسي:	Al Krarha, Osama J.
مؤلفين آخرين:	Hassine, Jamel Aldine(Advisor)
التاريخ الميلادي:	2014
موقع:	الظهران
الصفحات:	1 - 177
رقم MD:	738586
نوع المحتوى:	رسائل جامعية
اللغة:	English
الدرجة العلمية:	رسالة ماجستير
الجامعة:	جامعة الملك فهد للبترول والمعادن
الكلية:	عمادة الدراسات العليا
الدولة:	السعودية
قواعد المعلومات:	Dissertations
مواضيع:	لغات البرمجة، مشغلات الطفرة، هندسة البرمجيات
رابط:	https://search.mandumah.com/Record/738586

© 2019 دار المنظومة. جميع الحقوق محفوظة.
هذه المادة متاحة بناء على الإتفاق الموقع مع أصحاب حقوق النشر، علما أن جميع حقوق النشر محفوظة. يمكنك تحميل أو طباعة هذه المادة للاستخدام الشخصي فقط،
ويمنع النسخ أو التحويل أو النشر عبر أي وسيلة (مثل مواقع الانترنت أو البريد الالكتروني) دون تصريح خطي من أصحاب حقوق النشر أو دار المنظومة.

المنارة للاستشارات

www.manaraa.com

DESIGN AND EVALUATION OF MUTATION OPERATORS FOR THE ASMETAL LANGUAGE

BY

Osama J. AlKrarha

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

Software Engineering