Sudan University of Science and Technology

College of Graduate Studies

Entropy-Based Semantic Metrics for Software

قياسات دلالية للانظمة البرمجية مؤسسة على وظيفة الريبة

Submitted in partial Fulfillment for awarding the PhD Degree in Computer Science

By

Rasha Gaffer Mohammed Helali

M.Sc. Computer science, Sudan University of science and technology (SUST), 2008.
B.Sc. Computer science & information technology, SUST, 2003.

Director:

Ali Mili, Professor

New Jersey Institute of Technology

July, 2015

الايـه

قال تعالى:

أعوذ بالله من الشيطان الرجيم

**﴿ اللَّهُ لاَ إِلَهَ إِلاَّ هُوَ الْحَيُّ الْقَيُّومُ لاَ تَأْخُذُهُ سِنَةٌ وَلاَ نَوْمٌ لَّهُ مَا فِي السَّمَاوَاتِ وَمَا فِي
الأَرْضِ مَن ذَا الَّذِي يَشْفَعُ عِندَهُ إِلاَّ بِإِذْنِهِ يَعْلَمُ مَا بَيْنَ أَيْدِيهِمْ وَمَا خَلْفَهُمْ وَلاَ يُحِيطُونَ
بِشَيْءٍ مِّنْ عِلْمِهِ إِلاَّ بِمَا شَاء وَسِعَ كُرْسِيُّهُ السَّمَاوَاتِ وَالأَرْضَ وَلاَ يَؤُودُهُ حِفْظُهُمَا وَهُوَ
الْعَلِيُّ الْعَظِيمُ ﴾**

[سورة البقرة: 255]

Dedication

I dedicate my thesis work to my family and many friends. Special feeling, of gratitude to my loving parents and colleagues and all who assist me.

Acknowledgment

First and foremost I will thank Almighty GOD, the compassionate, the almighty Merciful, who kindly helped me to complete my thesis.

Second, it is a great pleasure to thank the many people that supported me during my Ph.D. studies. Words will never be enough to express my gratitude to my supervisor “Prof. Ali Mili” for his constant and precious guidance throughout my doctoral program, for the time spent on discussing and accurately reviewing all the research work of this thesis. Besides his scientific competence, I enjoyed a lot his positive attitude to make a friendly work atmosphere.

I would like to thank all those who assisted me in editing and reviewing my thesis and also the external reviewers of this thesis. Last but not least, my deepest gratitude goes to my family (parents, husband, sisters and brothers) for their continuous encourage and patience.

Rasha G.M.Helali

TABLE OF CONTENTS

TABLE OF CONTENTS……………………………………………….……..…….V

LIST OF FIGURES IX

LIST OF TABLES……….…..……………………………………………...X

Acronyms…………………………………………………………………...XI

English Abstract …..…………………..………………………………..…XII

Arabic Abstract…….. ………………………………………..………..……....XIII

Chapter 1: Introduction……………………………………………………………..

1.1 Introduction …….………….…………...……………………………………..2

1.2 Problem Statement..……….…………………...………………………………3

1.3 Research Questions and Hypothesis 3

1.4 Research Philosophy…….….……………………………………… ….……...4

1.5 Research objectives…………………………………………………………….5

1.6 Open research areas...…….…………………………………………………….5

1.7 Research Scope………...…….……………………………………………..6

1.8 Probosed Soluation ………….………………………………………………...7

1.9 Research Methodology……..………………………………………………….8

1.10 Thieses Organization…………………………………………………………9

Chapter2: Background ....

2.1 Introduction 11

2.2 Software Measurement and quality ……………………………………...12

2.2.1 Software measurement concept...12

2.2.2 Software quality Models 12

2.3 Software reliability measurement and predictions 21

2.3.1 The software reliability problem..23

2.3.2 Parametric Reliability models. 24

2.4 Software reliability growth Modeling/testing. 27

2.5 Software Metrics…………………………………………………………..28
2.5.1 Syntaic Metrics...31
2.5.2 Semantic Metrics..34
2.5 Chapter Summary..34
Chapter 3: Related Work ....
3.1 Introduction 36
3.2 Current research in software Metrics..36
3.2.1 Complexity Metrics 36
3.2.2 Measuering Web Application Complexity..38
3.3 Metircs for Measuering Relaiability...39
3.4 Data mining techniques for semantic metrics………………………………… 41
3.5 Semantic Metrics 43
3.5.1 Metrics Based On Entropy……………………………………………….…49
3.6 Summary of related Studies………………………………………………......51
Chapter 4: Research Methodology ....………………
4.1 Introduction……..……………………………………………...................56
4.2 Research Strategies……..……………………………………...................56
4.3 Research Process and Methods………..……………..……………………...59
4.3.1 Define goals………………………………………………….........59
4.3.2 Literature Study………………………………………………………60
4.3.3 Data Collection ………………………………………………………60
4.3.4 Prepare / analyze data……………………………………………...60
4.3.5 Design Assumptions…………………….…………………………60
4.3.6 Proposed Solution/ implementation/ testing…..……………………………61
4.3.7 Evaluation…………………………….……………………......62
4.4 Chapter Summary…………………………………………………….......62
Chapter 5: Software Reliability Mechanisms ………………………………..........
5.1 Introduction ………………………………………………………..…..64
5.2 Fault/ Error / Failure concepts.…………………………………….………… 64
5.3 Software reliability Mechanisms………………………………………………..65
3.3.1 Fault prevention………………………………………………………...65

5.3.2 Fault Removal………………………………………………………......65
5.3.3 Fault Tolerance……………………………………………….………66
5.3.4 Fault / Failure Forecasting……………………………………….…66
5.4 Information theory and entropy……………………………………........67
5.4.1 Information Theory……………………………………………....67
5.4.2 Relational Mathematics………………………………………….67
5.4.3 Entropy………………………………………………………....68
5.4.4 Measuring Information Contents………………………………70
5.5 Chapter summary………………………………………….…….……71
Chapter 6: Semantic Metrics………..…………………..…………………………
6.1 Introduction…………………..…………………………..……..73
6.2 Fault Tolerance Methodology…………...…………………………………76
6.3 Error detection: Redundancy………………….………………………76
6.3.1 State redundancy …………………………………….………...78
6.3.2 Functional redundancy……………………………………...78
6.4 Error Masking: Non injectivity………………………………………………79
6.5 Error Recovery: Non determinacy……………………………………………82
6.6 Summary of semantic metrics…………………………………………...83
6.7 Chapter Summary……………………………………………….......84
Chapter 7: Validation …….………………………………………………………
7.1 Introduction ……………………………………………………………….86
7.2 Empirical Research……………………………………………….…….86
7.2.1 Applying metrics………………………………………………………87
7.2.2 Correlation Analysis……………………………………….......89
7.2.3 Regression Results……………………………………………….90
7.3 Analytical Research…………………………………………………………91
7.3.1 Estimating probability of executing faulty statements…………………92
7.3.2 Probability of sensitization………………………………………………92
7.3.3 Probability of error propagation…………………………………………92
7.3.4 Probability of specification violation……………………………………92
7.4 Results ………………………………...……………………………..94

7.5 Using failure classification Model……………………………………...………94

7.5.1 Building classification model………………………………...………94

7.5.2 Classification result rules..………………………………...................95

7.5.3 Classification Model Limitations: ……………………………........96

7.6 Chapter Summary..…………97

Conclusion ……………………………………………………………….....98

Future Work…….………..……………..…………………………………98

REFERENCES...100

LIST OF FIGURES

Figure 1.1. Illustrates Detailed Research Process..8
Figure 2.1: Boehm ’Software Quality Characteristics Tree [11]…………………..14
Figure 2.2: McCall’s Quality Model [11]……………………………………......14
Figure 2.3: principles of Dromey’s quality model [13]……………………….……15
Figure 2.4: The ISO 9126 quality model [14]………………………………………16
Figure 2.5: ISO 9126 software product evaluation…………………………………19
Figure 2.6: Maturity Levels of (SW-) CMM…………………………..….…….20
Figure 4.1: Steps for Defining Metric………………………………………………61
Figure 6.1: A diagram showing a function that is not injective [95]……………….81
Figure 6.2: Meaning of non-determinacy……………………………………...82
Figure 7.1: Events generate system failure [98]…………………………………….91
Figure 7.2: Flow diagram including factors of system failure……………………...86
Figure 7.3: Classifiction model……………………………………………………..95
Figure 7.4: Classication rules……...………………………………………………..96

LIST OF TABLES

Table 2.1: Software entities with their attributes [9]……………………..……...11

Table 2.2: Measurements Scales [10]……………………………………………12

Table 2.3: Comparison between quality models [16]……………………………17

Table 2.4: Software reliability models..…………………………………….…28

Table 2.5: Software quality attributes evaluation against different views……..…29

Table 2.6: Different Measurements in terms of different roles………………..….29

Table .7: Table 2.7: summary of the main parametric reliability models………..34

Table 3.1: The Classification of Complexity Metrics from the view of SW …..…37

Table 3.2: The Classification of Complexity Metrics by their calculation base….37

Table 3.3: Summary of Related Work……………………………...…………51

Table 4.1 A Taxonomy of Research Methodologies……………………..…….57

Table 6.1: Measured functional redundancy……………………………………...79

Table 6.2: Metrics definition and interpretations…………………………………84

Table 7.1: Description about programs used in validation………………………..87

Table 7.2: Results of applying syntactic metrics on selected programs…....…….88

Table 7.3: Semantic Metrics applied on 10 selected programs……………………88

Table 7.4: Correlation Results……………..……………………………………...89

Table 7.5: Summary of regression results………………………………………...90

Table 7.6: Failure Probability for selected programs……………………………...94

Acronyms

N	Symbol/abbreviation	Full Text
1	ULS	Ultra Large Systems
2	MTTF	Mean Time to Failure
3	MTTR	Man Time To Repair
4	LOC	Line of Code
5	MTBF	Mean Time Between Failure
6	$\pi(xi)$	Probability distribution
7	H(X)	Entropy of variable X

Abstract

The big evolution in software field leads to dramatic increase in the need for existence of high quality quantitative measurements to insure the quality of software artifacts. Software engineering researchers and practitioners have traditionally relied on metrics to quantify attributes of software products and processes. By considering software products in particular, we find that most existing tools are typically based on a syntactic analysis. At a time when software systems grow increasingly large and complex, the focus on diagnosing, identifying and removing every fault in the software product should progress to a more measured and realistic approach like fault tolerance. Semantic metrics are a good fit for this purpose, reflecting as they do the system’s ability to avoid failure rather than its proneness to being free of faults. This study introduces four semantic metrics based on entropy concept that consider software failure life cycle. Both empirical and analytical researches were used to validate the suggested metrics. An analytical model is proposed and used to predict failure probability by using semantic metrics in addition to other simple metrics. The results show how semantic metrics can be used as an indicator to software reliability.

المستخلص

التطور الكبير في مجال البرمجيات ادى إلى ازدياد الحاجة لوجود قياسات كمية عالية الجودة لضمان جودة المنتجات البرمجية. وقد اعتمد الباحثون في مجال هندسة البرمجيات والممارسين على المقاييس الكمية التقليدية لقياس سمات المنتجات البرمجية والعمليات عليها. من خلال النظر في منتجات البرمجيات على وجه الخصوص، نجد أن معظم الأدوات المستخدمه حاليا للقياس تستند عادة على التحليل النحوي. في الوقت الذي تنمو نظم البرمجيات الكبيرة والمعقدة على نحو متزايد، مع التركيز على تشخيص وإزالة الاخطاء من منتج البرنامج يجب أن يتطور هذا النهج إلى نهج أكثر توازنا وواقعية مثل المقدرة على مواجهة الاخطاء. المقاييس الدلاليه هي الاكثر مناسبة لهذا الغرض، حيث انها تعكس قدرة النظام على تجنب الفشل بدلا من خلوه من العيوب. هذه الدراسة تقدم اربع مقاييس دلالية على أساس مفهوم وظيفة الريبة "Entropy" و تعمل على قياس مراحل حدوث الفشل في البرنامج, استخدمت الدراسة كل من الأبحاث التجريبية والتحليلية للتحقق من مدى صحة هذه القياسات. استخدم النموذج التحليلي لقياس احتمالية فشل المنتج البرمجى باستخدام القياسات الدلاليه وبعض القياسات الاخري. أظهرت النتائج امكانية استخدام المقاييس الدلالية كمؤشر لموثوقية البرمجيات.