Automation of Large Class Code Smell Detecting and Refactoring

Submitted in Partial Fulfilment of the Requirements for the Degree of Master of Science in Software Engineering

By

Basma Rakan Almasaeid

Supervisors

Dr. Aysh Alhroob **Dr. Ayad T. Al-Zobaydi**

Faculty of Information Technology
Master Program of Software Engineering
Isra University

2018/ 2019

AUTHORIZATION STATEMENT

I, Basmah Rakan Almasaeid, authorize Isra University to provide hard copies of soft copies of my thesis to libraries, institutions or individuals upon their request.

Name: Basmah Rakan Almasaeid

Signature:

Date:

إقرار تفويض

أنا بسمه راكان المساعيد، أفوض جامعة الإسراء للدراسات العليا بتزويد نسخ من رسالتي ورقياً وإلكترونياً للمكتبات أو المنظمات أوالهيئات والمؤسسات المعنية بالأبحاث والدراسات العليا عند طلبها.

الاسم :بسمه راكان المساعيد

التوقيع:

التاريخ:

The undersigned have examined the thesis entitled '**Automation of large Class Code Smell Detecting and Refactoring**' presented by **Basma R. Al-Srour**, a candidate for the degree of Master of Science in Software Engineering and hereby certify that it is worthy of acceptance.

Date	**Dr. Aysh Alhroob**
Date	**Dr. Ayad Tareq Al-Zobaydi**
Date	**Dr. Mohammad Ali Ajanini**
Date	**Dr. Yousef Elsheikh**

DEDICATION

I would like to dedicate this thesis to my parents, family and my supervisors, Dr. Aysh Alhroob and Dr.Ayad Tareq Al-Zobaydi whom without this work would have never been completed. Their constant support and encouragement allowed me to overcome my obstacles.

BASMA R. AL-SROUR

May 2019

ACKNOWLEDGMENTS

First of all, I would like to say “al hamdulillah” for all the blessings I have been blessed with, my parents, family, supervisors, friends, and teachers. I would like to express my sincerest gratitude to my advisors Dr. Aysh Alhroob and Dr. Ayad Tareq Al-Zobaydi and a special thanks to Mutaz Aramin for his help with the technical aspects of the approach.

I would like to also thank the rest of my thesis committee for their time and endeavors to aid me with my work and their constant encouragement through the good and bad times.

I also would like to extend my thanks to all my colleagues, and my professors whom without I wouldn’t be the person I am today.

Finally, and most importantly I would like to thank my parents for giving me the opportunity to grow and become a better version of me. Their constant support, help, and caring are what helped me throughout all my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviation	Word
AC	Attribute Complexity
ACMIC	Ancestors Class-Method Import Coupling
ADLCBD	Automation Detect Large Class Bad Smell
AIF	Attribute Inheritance Factor
AMWNAMM	Average Methods Weight of Not Accessor or Mutator Methods
ANN	Artificial Neural Network
AOFD	Access of Foreign Data
ArcNbr	Number of arcs
ATLD	Access to Local Data
BPNN	Back Propagation artificial Neural Network
Calls	Number of Operation Calls
CBO	Coupling Between Objects
CCM	Class Coupling Metrics
CCMetrics	Coupling and cohesion Metrics
CDISP	Coupling Dispersion
CFNAMM	Called Foreign Not Accessor or Mutator Methods
CH	Class Hierarchy Metrics
CINT	Coupling Intensity
CLC	Class Level Coupling
CLD	Class to Leaf Depth of an entity
CLNAMM	Called Local Not Accessor or Mutator Methods
CndSpnAvg	Average of decisions Span
COF	Coupling Factor
Cparser	Classes Parser
CrdNbr	Number of Decisions

CYCLO	Cyclomatric Number Complexity
DAC	Data Abstraction Coupling
DCMEC	Descendants Class-Method Export Coupling
DD	Data-Data Interaction
DIT	Depth of Inheritance Tree
DM	Data-Method Interaction
FANOUT	Number of Called Classes
FDP	Foreign Data Providers
FF	Fitness Function
FL	File Length
FV	Fitness Value
GA	Genetic Algorithm
GDC	Global Data Coupling
IC	Inheritance Coupling
jPeek	Java Peek (Tool of calculate the cohesion metrics)
KntNbr	Number of Knots
LAA	Locality of Attribute Accesses
LCC	Loose Class Cohesion
LCSD	Large Class Smell Detection
LOCC	Line Of Code Per Class
LOCM	Lack of Cohesion in Methods
LOCNAMM	Lines of Code Without Accessor or Mutator Methods
LopNbr	Number of Loops
MAXNESTING	Maximum Nesting Level
MIC	Method Invocation Coupling
MM	Method-Method Interaction
MPC	Message Passing Coupling
NAD	number of attributes declared

NdsExNbr	Number of exits in a Function
NdsNbr	Number of Nodes
NM	Number of Methods
NMA	Returns the NMA (Number of New Methods) of an entity
NMI	Returns the NMI (Number of Methods Inherited) of an entity
NMO	Number of Methods Overridden
NOAV	Number of Accessed Variables
NOC	Number of Classes
NOCS	Number of Classes
NOF	Number Of Fields
NOII	Number of Implemented Interfaces
NOLV	Number of Local Variable
NOM	Number of Methods
NOMNAMM	Number of Not Accessor or Mutator Methods
NOP	Number of Packages
NOPA	Number of Public Attribute
NOPK	Number of Packages
NORM	Number of Return Method
NPM	Number of public instance methods
NsLvLAvg	Average nesting Level
OAC	Operation Argument Complexity
OLC	Object Level Coupling
OO	Object Oriented
Opcom	Operation Complexity
OPCPL	Operation Coupling
Opcpl	Operation Coupling Metric
P	Population (Chromosomes0
PthIndNbr	Number of Independent Paths

RFC	Response For a Class
SAD	Software Architectural Defects
SDLC	Software Development Life Cycle
SIX	Returns the SIX (Specialisation IndeX) of an entity
SIZE2	Number of Properties
StmCtLNbr	Number of Control Statements
StrBrcNbr	Number of branches of structure
TCC	Tight Class Cohesion
UML	Unified Modelling Language
WMC	Weighted Method Count
WMCNAMM	Weighted Methods Count of Not Accessor or Mutator Methods
XML	Extensible Mark-up Language

ABSTRACT

Software Quality is an important issue in the development and success of the software. It is concerned with modifications and improvements necessary to meet the evolving needs and performed during maintenance phase of Software Development Life Cycle (SDLC). The problem that is accompanied to any modification is the possible low quality of the resulted software.

Large class bad smells are serious design flaws that could affect the code's quality attributes such as understand ability and readability. These flaws could ultimately lead to difficulties in maintaining the code and adding new functionalities. This work aims to detect large class bad smells automatically to help developers and engineers to detect large class bad smells from the get-go. This support keeping the code clean and easy to be understood, thus eliminating the need to constantly referring back to the documentation every time we try to add or repair functionality.

Usually, the large class bad smell is identified by using the coupling and cohesion metrics and compared to the identified class smelly elements to determine if one or more large class bad smells exist. Large Class Smell Detection (LCSD), is a proposed approach used in this work to automate the development of a large class bad smell detection model that is based on cohesion and coupling metrics. The automation of this development utilizes Genetic Algorithm (GA) and Artificial Neural Network (ANN).

LCSD's results showed that its performance is very good in finding large class bad smells. The correctness of LCSD has been measured by using binomial technique, and achieved high results, which is 96.67%.

Keywords: Large Code Smell, Software Engineering; Software Quality Evaluation, Genetic Algorithm, Neural Network, Quality Assurance, Cohesion, Coupling.