PA2559
Software Metrics - Assignment 1

DhanaLakshmi Ponugubati – *dhpo18@student.bth.se – 9805013563*
Muhammad Waqas – *muaq18@student.bth.se – 9007038251*
Waleed Rasheed – *aara18@student.bth.se – 9301155033*
SriRanganathReddy Sabbella – *srsa18@student.bth.se – 9705230119*

April 19, 2019

Abstract

This assignment consists of two parts.The first part focuses on Object oriented metrics.There are different categories in Object oriented designs.But this assignment is focused on only four categories.They are Cohesion, Coupling, Maintainability, Understandability.The second part focuses on reflections on the tools used to extract the metrics.Two tools are considered in this part for analyzing a open source software project in java.

1 Introduction

Object oriented(OO) design is getting to be more well known in object oriented environment.Metrics plays a vital role in object oriented environment.The main focus of this assignment is to get familiarized with the object oriented metrics and the tools used to extract them and also methods used to visualize.
First of all we analyzed 10 different software metric based research papers in Cohesion, Coupling, Maintainability, Understandability categories.We identified different metrics,types of systems,types of metrics extraction tools,methods used to visualize extracted tools and statistical measures in the next sections.
Next we selected two metrics extraction tools and a open source software developed in java.From these we extracted the metrics and reported our findings in the part 2 section.

2 Part 1: A Review on Object Oriented Code Metrics

2.1 Cohesion

The most important attribute of object oriented is cohesion. It is a measure of functional strength of module. It make sure that the class is designed for well-focused purpose. The most benefit of cohesion is that such classes can be reusable as compared to the other classes.

2.1.1 Summary of An Object-Oriented High-Level Design-Based Class Cohesion Metric

In this paper the author have performed an experiment to propose high level design metric. They have chosen 4 Java open source systems.The reason of selecting these system is that these systems have large number of classes, from different domains, and have available fault repository and source code. They have selected 3 HLD and 6 LLD cohesion metrics to compare with SCC as mentioned in Table 1. They have presented 3 analyses. The 1st one explore the correlation among the ten cohesion metrics. The 2nd and 3rd explore the presence of faults in the classes. They have calculated the non-parametric spearman's correlation among the cohesion metrics. The main purpose of this paper is the comparison of SCC with HLD so, the result is that SCC is empirically verified and more realistic assumption than other metrics. However the metric HLD is not basically to measure the class cohesion with maximum correctness.The limitation is that it is based on assumption because the information of the source code is not available[3].

2.1.2 Summary of Improving the Applicability of Object-Oriented Class Cohesion Metrics

This paper describes how cohesion is important for the developers in OO to understandable, reusable, modifiable and maintainable prospective. In this paper the author have proposed values for 12 cohesion metrics.Then they have studied empirical relationship between the cohesion values and the faulty classes present in 4 Java project used for this. They have assigned the values to cohesion metrics in these criteria, cohesion of a class with no metrics, cohesion of a class with single

Table 1: **Cohesion Metrics**

Metrics	Description	Category	References
CAMC	CAMC considers the relative number of distinct types used by the parameters of the methods. The value ranges within the interval [0, 1]	Interface based	[1]
NHD	Normalized Hamming Distance is the lack of cohesion metric based on measuring the agreement in the term of sharing of parameters types between the pair of method.	Interface based	[1]
SNHD	SNHD=the closeness of the NHD metric to the maximum value of NHD compared to the minimum value.	Interface based	[2]
LCOM1	LCOM1 is the number of pairs of methods that do not share attributes	Component based	[2]
LCOM2	LCOM2 calculates the difference between the number of method pairs that do and do not share instance variables.	Component based	[1]
LCOM3	LCOM3= Number of connected components in the graph that represents each method as a node and the sharing of at least one attribute as an edge.	Component based	[2]
TCC	TCC= Relative number of directly connected pairs of methods, where two methods are directly connected if they are directly connected to an attribute.	Component based	[2]
LCC	LCC=Relative number of directly or transitively connected pairs of methods, where two methods are transitively connected if they are directly or indirectly connected to an attribute.	Component based	[2]
DCD	DCD= Relative number of directly connected pairs of methods, where two methods are directly connected if they satisfy the condition mentioned above for TCC or if the two methods directly or transitively invoke the same method.	Component based	[2]
LCOM4	Similar to LCOM3 and additional edges are used to represent method invocations.	Component based	[2]
DCI	DCI= Relative number of directly or transitively connected pairs of methods, where two methods are transitively connected if they satisfy the same condition mentioned above for LCC or if the two methods directly or transitively invoke the same method.	Component based	[2]
Coh	Coh accounts for the relative number ofattributes referenced by methods, its valueranges within the interval [0, 1].	Connection based	[1]
LSCC	LSCC defines the similarity between each pair of methods as the ratio of the number of sharedattributes between the methods to the total number of attributes in the class.	Connection based	[1]
LCOM5	LCOM5 measures a lack of cohesion across multiple methods by considering the number of methods that reference each attribute. TheydefineLCOM5=(a-kl)/(l-kl).	Connection based	[1]
CC	CC= Ratio of the summation of the similarities between all pairs of methods to the total number of pairs of methods.	Connection based	[2]
SCOM	The Sensitive Class Cohesion Metric(SCOM) (Fernandez and Pena 2006) is similar to the CCmetric. The only difference liesin the definition of similarity.	Connection based	[1]
LCOM	LCOM calculatesthe difference between the number of method pairs that do and do not share instancevariables.	Product based	[3]

method and no attribute, cohesion of a class with multiple methods and no attributes.Then they presented 4 analyses. They have developed their own Java tool to automate the cohesion measurement process. The overall result came from this paper is that having the class cohesion metrics improves the applicability of the metrics and possibly increases their precision in signifying class quality. The limitation in the result is that all 4 programs are implemented in Java. Other programming languages have some features that are different from Java[1].

2.1.3 Utilization of Method Graphs to Measure Cohesion in Object Oriented Software

In this article the author conducted the graph based analysis of object oriented software by comparing widely used LCOM cohesion metric. To extract the graph they have used the Abstract Syntax Tree Java parser. They have analyzed 4 different graph method RCM, RMC, R1C, and RML. They have used the JUNG framework for the graph whereas they have used R for the similarities measurements in different clustering. There are 2 main contributions of the result. The one is software wide cohesion and the second is cohesion measurement techniques based on the static analysis. They have performed this metrics on 14 open source Java systems.The result represented in this by call graph. Statistically the average and median calculated for this data. The result of this study shows that measuring the differences grouping by using graph produce the most correlated results with the LCOM metric[4].

2.2 Maintainability

Software maintainability is making changes to the delivered software in order to improve its quality and to rectify the faults. It is the key attribute in determining the software quality of a product and is difficult to estimate, which involves in making predictions about future changes of software module.

2.2.1 Machine learning approaches for predicting software maintainability: a fuzzy-based transparent model

In this paper author proposed a process for fuzzy logic (FL) based transparent quality prediction models and these models are processed to a case study where Mamdani fuzzy interference engine is used to predict maintainability.A model that has better accuracy and transparency has been designed. Among these models Mamdani based fuzzy logic, Takagi Sugeno (T-S), BN (Bayesian Networks), NN (Neural Networks), SVM (Support vector Machines), Mamdani based fuzzy logic has better accuracy to predict the software maintainability.The transparency of Mamdani based model approach is the clustering the linguistic values to number of clusters equal to linguistic values for each variable. The limitations of this study is two data-sets are used in the study are not enough to draw conclusions[5].

2.2.2 Software maintainability prediction by data mining of software code metrics

The context of this paper is to predict the software maintainability by using the metrics extraction tools with the static code metric dataset of four different open source software (OSS). The objective is to use the new prediction metrics as well as traditional metrics by data mining for prediction of software maintainability. The additional software modules that are difficult to maintain are derived in this paper. Among Naive Bayes, Bayes Network, Logistic, Multilayer Perception and Random Forest classifiers, Random Forest models is found to be most useful in software maintainability prediction by data mining of source code metrics. The limitations in this study are one in the selection of predictor variables and the other is usefulness in predicting maintainability effort in other paradigms[6].

2.2.3 Improving software maintenance size metrics A case study: Automated report generation system for particle monitoring in Hard Disk Drive Industry

Software maintenance size metric is used to predict maintenance effort or maintenance time.In this paper author used case study for predicting maintenance time based on the proposed new maintenance size metrics using three maintenance versions in object-oriented language. The objective is to use the new metrics as well as the traditional maintenance size metric to measure the performance of each metrics.He found that weighted methods per class (MS-WMC) gave the best performance after analyzing the results.The modification of a software product after delivery includes enhancing source code based on additional requirements and refactoring source code to maintain readability. For enhancing maintenance task, maintenance size based on average number of methods per class (MS-MC) gave the best performance while in refactoring maintenance task, maintenance size based on Line of code (MS-LOC) gave the best performance. On average, maintenance size based on weighted methods per class (MS-WMC) gave the best performance. The limitations in this study is limiting to only object-oriented language[10].

2.3 Coupling

Coupling is level of dependency between the software modules.It determines the robustness of connection be-

Table 2: **Maintainability Metrics**

Metrics	**Description**	**Category**	**References**
DIT	In Depth of inheritance tree each class measure of the inheritance levels from the object hierarchy top	Product based	[5]
NOC	Number of Children is total number of subclasses of each class in project.	Product based	[5]–[7]
RFC	Response for a class is different methods that can be executed when object of that class invoked	Product based	[5], [8], [6]
LCOM	Lack of cohesion in methods counts number of null pairs of method	Product based	[5], [6]
WMC	Weighted Method Class is sum of complexities of its methods	Product based	[5], [6]
NOM	Number of methods in a system	Direct Measure	[5]
MPC	Message Passing Coupling is number of method calls in a class	Direct Measure	[5], [9], [6]
SIZE1	Count number of lines of code, comments but not whitespace.	Code based	[5]
SIZE2	Count number of lines of code but not whitespace	Code based	[5]
B (Halsted bugs)	It measures the bugs in a class	Code based	[6]
CLOC	Comments Lines of Code is the number of lines of code in each class which contain comments	Code based	[6]
Command	Number of comments in each class	Code based	[6]
Cons	Number of constructors is the total number of constructors in a class	Code based	
Inner*	Number of inner classes or interface which each class contain	Code based	[6]
Dcy*	Number of transitive dependencies is number of classes or interfaces which each class directly or indirectly depend	Dependent	[6]
DAC	Data abstraction coupling is number of instantiations of other classes within given class	Dependent	[5], [9]
MS-LOC	Maintenance size metric based on Line of code is the ratio of added and modified line of code to total number of line of code in the initial system	Ratio based	[10]
MS-TC	Maintenance size metric based on total number of classes is the ratio of numbers of changed classes to the number of total classes in the initial system	Ratio based	[10]
MS-TM	Maintenance size metric based on total number of methods is the ratio of changed methods to total number of methods in the initial system	Ratio based	[10]
MS-MC	Maintenance size metric based on methods per class is the ratio of average number of changed methods per class to average number of methods per class in the initial system	Ratio based	[10]
MS-WMC	Maintenance size metric based on weighted methods per class is the ratio of changed weighted methods per class to weighted methods per class in the initial system.	Ratio based	[10]

tween the software modules.

2.3.1 A Novel Coupling Metrics Measure difference between Inheritance and Interface to find better OOP Paradigm using C#

Object oriented programming is one of the most followed programming approach in industries for various types of the systems.Narendra Pal Singh Rathore and Ravindra Gupta measure the coupling metrics of to provide the difference between interface and inheritance programming.To explore the difference there are two different programs (interface and inheritance) are coded in C#. Purpose of the study is to help the c# developers that which program is efficient. The software metrics that are measured for the study are ,Coupling between object CBO, number of associations between classes NASSocC , number of dependencies in metric NDepIN, number of dependencies out metric NDepOut and number of children NOC. Both programs using inheritance written in C# and showed with highest possible interfaces these five metrics are applied to both programs .Metrics are measured by using metrics suites provide the idea to reduce the coupling and quantify the software quality by reusability of the program in object oriented programming[7].

2.3.2 Using Relational Topic Models to Capture Coupling among Classes in Object-Oriented Software Systems

Coupling metrics deals with the interaction and relations in class between class functions interfaces. In this article new coupling metrics is narrated named as Relational topic coupling (RTC) of the class, this newly proposed metrics is different form the other coupling metrics it is used to analyze the encapsulation and abstraction of the classes and other latent studies is use relational topic model (RTM). RTM further extends LDA (Latent Dirichlet Allocation) which is recently used for extraction analyzing the latent studies.This new metric is measured and compared with current structure and traditional coupling metric with the help of designed case study of 13 different open source software.The presented metrics is newly developed with having some validity threats.The metrics is mainly used for two large system and one medium system for generalization of the system need another large study on different programming language currently this metrics is only analyzed through C++. In this complete analysis nine static metrics are measured using different tools.From the above discussion and implementation tools of the shows that RTC is useful and this is theoretical validated[12].

2.3.3 Principal Component Analysis of Coupling Measures for Developing High Quality Object Oriented Software

Object oriented approach is one of the most fast approach programming. Object oriented model presents the class model which is a clear picture of the relations and inheritance. This design model shows the interdependency between the classes functions and logics in this paper the researchers analyze the main problems of the coupling metrics. These metrics are Number of used classes by dependency relation, used classes by the dependency relation, Ratio Of NUCD to TNUCD, Number of user classes for class, total number on use classes through class, ratio of the TNUCC, class coupling attribute hiding factors method hiding factors. These measures has been analyzed by experimental analysis which shows the prefect selection of the perfect metrics in terms of interactions between distinct components in OO environment. Researchers follows the methodology of categorization they divided metrics in to two categories one is ratio oriented and the other is ratio less this experiment has been implemented on different software system mentioned in below table row 6. For analyzing the interaction of the program system design analyzer has been used. Results of this experiment design shows that validation helps understanding OO coupling in unique frame works[11].

2.4 Understandability

Understandability is the average amount of effort that is required to comprehend.Understandability increases the quality of the software

2.4.1 Exploring the Relationships between Design Metrics and Package Understandability: A Case Study

As the quality of the software is increasing in object-oriented (OO) software. It is necessary to study and analyze the packages and size in an OO paradigm. The attributes of the packages derived from package-level metrics relate to the quality of the software.Hence a verifiable validation is required in order to relate these metrics that result in the quality increase of the OO software.This study was conducted to determine the relationship between five package level metrics and the understandability of the package in object-oriented design. The effort needed to understand the package ranges from 1.0 to 9.0 with mean of 3.6. Spearmans rank correlation analysis is performed to test the hypothesis. From the hypothesis testing the correlation between package-level metrics and average effort needed to understand the package is determined.The improved prediction could be achieved by considering other structural properties in addition to size.In this study instead of measuring the understandability of the

Table 3: **Coupling Metrics**

Metrics	**Description**	**Category**	**References**
CBO	This measure the number of coupled classes on basis of methods reusability	Quantify based	[7]
NUCD	NUCD is the counting measure which is used to count dependent classes in this measure parameter variable is enough evidence to prove the dependency.	Quantify based	[11]
NDepIN	This measure counts the number of dependent classes the output of this measure is to show the complexity of the classes.	Quantify based	[7]
CBO	This type of metrics count how many classes we coupled during the experiment	Product based	[7],[8]
ICP	Type of the metrics used to measure the information transfer between the components it also analyze statistically by measuring the complexities of the methods.	Indirect measure	[12]
Briand Suite of Coupling	This suites include ACIAC, OCAIC FCAEC is used to measure the interactions of the CM (class modeling)	Indirect measure	[13]
Ratio of the TNUCC	Calculate the ratio of dependent classes in OO program	Ratio based	[11]
Ratio Of NUCD to TNUCD	This measure is used to calculate the ratio for more than one dependent classes	Ratio based	[11]
NASSocC	This measure show the association concept between the classes either self-association relation or relation with the other classes	Dependent	[7]
TNUCD	How many evidence of the dependent class (functions parameters or var) is calculated by this measure	Dependent	[11]
Number of user classes for class NUCC	Measure shows how many classes are dependent on each other	Dependent	[11]
Number of Evidence TNUCC	The output of this measurement shows the evidence of dependent classes	Dependent	[11]
NdepOut	This measure count those classes on which many other classes are dependent	Dependent	[7]
Class Coupling (CC)	CC shows the total sum of the coupled class	Internal	[11]
Visible Member	This measure calculate the number of public members of the class (functions parameters variables).	Internal	[11]
Attribute hiding factors	This measure defines the sum of the number of visible attributes of classes and methods	Internal	[11]
Method hiding factors	This factors shows the sum of the visible or public methods of the class	Internal	[11]

Table 4: **Understandability Metrics**

Metrics	Description	Category	References
Number of classes(NC)	Number of classes present in a package.It Measures the size of a package.	Package-level	[14]
Afferent Couplings (Ca)	The number of other packages depending upon the classes present within this package.it measures incoming dependencies.	Package- level	[14]
Efferent Couplings (Ce)	Number of other packages that the classes in the package depend upon.It measures outgoing dependencies.	Package-level	[14]
Instability (I)	It is defined as the ratio between the efferent coupling to the total coupling that is I = Ce / (Ce + Ca)	Package-level	[14]
Distance (D)	It is defined as the package perpendicular distance from idealized line(A+I=1) where A is the percentage of the package abstract classes to the total number of classes.	Direct measure	[14]

package they used recently derived dataset. Limiting the size of the dataset. Believing that the data is reliable[14].

2.5 Findings from the papers

The findings from the papers include metrics used,types of systems studied,metrics extraction tools,visualization methods and statistical measures.

2.5.1 Metrics

- For cohesion studies [3],[1],[4] the metrics are tabulated in Table 1.
- For maintainability studies [5],[6],[10] the metrics are tabulated in Table 2.
- For coupling studies [7],[12],[11] the metrics are tabulated in Table 3.
- For understandability studies [14] the metrics are tabulated in Table 4.

The categorization of the metrics are mentioned in the tables.The categorization is based on the count,connection,components,products,measures.Each metric is categorized depending upon the specifications.

2.5.2 Types of system

- **Graphical systems**:Art of illusion is open source and used for creating 3D graphics.JHotdraw is used to draw structured editors written in java[6].**Management Systems**: Gantt project is used for managing and scheduling projects.JabRef is a software that uses BibTeX and BibLaTeX for reference management.**Cloud Based**: Openbravo is specialized retail solution cloud based software[3],[1].
- **Backup**:Borg is a backup software used for deduplicating.**Language based**: Clojure is used as compile language.JavaParser is parser generator.Freemind,Freeplane are mind mapping software's.**Tech based**: Logisim is used for designing digital logic circuits..WURFL is a mobile detection software [4].**Generators**: ARGS is an automated report generation tool[10].XGen source code generator is java based that creates the outputs from the inputs.Jakarta ECS is an element constructor system which contains many packages and classes.[14].Doxygen is used for generating documents[12].
- **Editors**:ANote is used for writing notes[12].Jedit is a text editor for programmers[6].JTree view is used to view the tree based on java[6].pdfsam is used to extract pages from pdf file..WinMerge compares and merges the files.[4].Tortoise CVS is a client concurrent version system.**Database systems**: Admission test management system is used for storing candidate data[11].UIMS is a user interface management system is used to separate graphics and process[5].QUES is the system used to evaluate the quality of the software[5].**Chatting software's**: Turtle chat is client-server internet protocol chatting software.Com chat uses API's of java communication for chatting[11].

2.5.3 Metrics extraction tools

- In studies [3],[1] own developed tools are used for extracting metrics.

- Abstract syntax tree(AST) is used to represent the source code in a tree structure[4].
- CK metric is Chidamber and Kemerer metrics suite consists of 6 metrics for a class[7],[5][6].
- Bugzilla is a bug tracker Bugzilla is an online open source tool for tracking bugs[12].
- Experimental design ratio oriented and ratio less category is examining the specific data[11].
- IntelliJIDEA is a tool for indexing and developing the source code.[5], [6].
- In [14] there is no direct tool mentioned for extracting metrics.

2.5.4 Methods to visualize extracted metrics

Tables and UML diagrams are used to visualize the metrics.The DMI matrix for the AccountDialog class,UML class and communication diagrams for AccountDialog,The MI matrix for the AccountDialog class ,A sample class diagram and its corresponding parameter-occurrence and DAT matrices ,The DAT matrix for the AccountDialog class,The AT matrix for the AccountDialog class ,The intuitive results for the class cohesion in different scenarios ,Descriptive statistics for the cohesion measures ,The matching percentages of the DAT and parameter-occurrence matrices,Spearman rank correlations among the cohesion metrics ,Univariate logistic regression results ,Multivariate logistic regression results are the tables and UML diagrams [3].

Tables are the visualization methods used in [1],[12][6],[14].The tables include defined and undefined values for the considered metrics, the numbers and percentages of special classes and cases, descriptive statistics for the cohesion measures after improving the applicability,applicability of the considered metrics,the numbers and percentages of special classes and cases.All the results and the comparisons are visualized in the form of tables. Correlations between CHANGE and OO metrics ,model evaluation.

Call Graph,Clustering Graph,tables are the visulaization methods used to extract metrics in [4],[7]. Average / Median LCOM HS Values and the Similarity Coefcient (Rand Index) of the Method Graphs GCM and GMC, Graphical Representation of the Correlations,correlation coefficients between rand indices and LCOM.Coupling Measures Comparison, Coupling Measures for total Inheritance and Interface programs,comparing Measures are used.

Figures,Graphs,Tables are the methods used to visualize the metrics in [5].Software quality assessment framework, Correlations between CHANGE and OO metrics.

Tables,graphs,bar charts are methods used to visualize the metrics in [10].Graphs Comparing Actual and Predicted Maintenance Time for Each Metric,,collected data during software maintenance and development.

2.5.5 Statistical measures

There are many statistical measures in the studies referrenced.Much of the data is represented in the form of tables.These tables provide the results of statistical measures.**Mean** is the average of all the numbers which is mostly used.It is calculated by summing all the values and dividing it by total values in the dataset.Mean is the statistical measure used in [3],[1],[4],[5],[10],[14]. **Quartile** is used to divide the data-set into four equal sized parts.The quartile measures depends upon the median.Quartile statistical measure is used in [3],[1].**Median** is the middle value of the dataset when the data is arranged in ascending order[3], [1], [4].**Standard Deviation** is a measure of deviations.It is used to findout the standout from normals[3], [1], [5].**Variance** is the square of standard deviation.**Eigen analysis** is performed on covariace matrix[11].There is no specific statistical measure used instead the measures are compared and the results are extracted[7].**Precision** and **recall** are two extraction tools which are used to measure the coupling measure performance[6]. Wilcoxon Test is the is used to test the statistical significance.**Principal component analysis** is performed for understanding the orthogonal dimensions on the measured coupling metrics[12].**Pearson's Correlation Coefficient** is used to measure the linear correlation between two variables[6].**Spearmans rank correlation** determines the statistical relationship between two ranked variables[14].**Multivariate Regression Analysis** is used to evaluates the multivariate variable with a single regression model[14].

3 Part 2

The selected tool for extraction of metrics are Eclipse Metrics Plug-in 1.3.8 and Chidamber and Kemerer Java Metrics (CKJM-1.9). At first, we found it difficult to select a tool because we were confused about the types of tool available. We tried installing other metrics extraction tools, but we found better tools that have many features for extracting metrics.The Open Source Software (OSS) we used is frogger a game which is developed in java and downloaded from the GitHub. It is a 100% java code.

3.1 Tool 1

Eclipse Plug- in 1.3.8 is an open source software for java programming language.The tool is suited for

Table 5: **Comparision of tools used**

Tools	Code	Input	Outputs	Metrics
Eclipse Metrics 1.3.6	Yes	Java	Eclipse console view	NOC,DIT,NORM,NOM,LOC,WMC,LCOM*,McCabe Cyclomatic Complexity,Ca,Ce,I,A,Dn
CKJM 1.9	Yes	Java	Console XML	WMC, DIT, NOC, CBO, RFC, LCOM, Ca, NPM

Eclipse IDE. The following are different types of metrics can be extracted, they are NOC(Number of Children), DIT (Depth of inheritance tree), NORM (Number of Overridden Methods), NOM (Number of Methods), LOC (Line of Code), WMC(Weighted Method Class), LCOM* (Lack of Cohesion of Methods), McCabe Cyclomatic Complexity, Ca (Afferent Coupling), Ce (Efferent Coupling), I (Instability), A (Abstractness), Dn (Normalized Distance from Main Sequence). Eclipse Metrics Plug-in 1.3.8 shows a Dependency Graph View of the metrics.

3.1.1 Ease of installation and use

In Eclipse IDE open toolbar contains Help.By clicking on that a new software is installed, a window will open which ask to insert the site and download the Eclipse plug-in 1.3.8. After completion of plug-in installation open toolbar that consists window option in that .Then moving to show view that leads to listthen clicking on other the metrics window is displayed. This shows that metrics plug-in 1.3.8 is installed.

3.1.2 Open Source Software

We cloned the project 'frogger' and loaded in Eclipse IDE and imported to package explore. Then by right clicking on the project we find an option of properties by clicking on, it shows a list in it from which we enabled metrics.The metrics are displayed on console output and Dependency Graph View metrics can also be viewed.

3.1.3 Coverage Of Object Oriented Metrics

This tool covered almost 23 Object-Oriented Metrics.

3.1.4 Type of Reports Generated

The reports generated by Eclipse Plug- in 1.3.8 are Tables that consists all the metrics and Dependency Graph view of a selected metric.The XML reports can also be generated for the results.

3.2 Tool 2

CKJM is an open source software for object-oriented systems that measures complexity in design of a class.We used version 1.9. It is a design-based metric that runs compiled files (.class files) of Java. The following types of metrics can be extracted, they are WMC(Weighted Method Class), DIT(Depth of inheritance tree), NOC(Number of Children), CBO (Coupling between object classes), RFC(Response for a class), LCOM(Lack of cohesion in methods), Ca (Afferent Couplings), NPM (Number of Public Methods). While executing the class files the output is shown in the above listed metrics order.

3.2.1 Ease of installation and use

CKJM metrics tool is an open source software.It is installed and it runs in command prompt by placing build path of executable .jar file of CKJM in the command prompt. By using the compiled class files of Java the metrics for each class are displayed.

3.2.2 Open Source Software

In CKJM, build path of executable .jar file of CKJM is set and the class files of project 'frogger' is set to build path of CKJM, and all the outputs of class files are shown in command prompt with each class having description of metrics.

3.2.3 Coverage Of Object Oriented Metrics

This tool covered only 8 Object-Oriented Metrics.

3.2.4 Type of Reports Generated

The reports generated by CKJM are list visualization of numeric data that is in the order of the metrics as WMC, DIT, NOC, CBO, RFC, LCOM, Ca, NPM. Each class has a particular list of the above order of metrics.

The metrics generated by Eclipse Plug- in 1.3.8 are at a package and class level which are shown in Metrics View in Eclipse IDE.If needed it also provides the method level metrics. But the metrics generated by CKJM are at class level and each class files with respective metrics are shown as numeric data in console of the CKJM.

3.3 Conclusion

From this assignment we learned more about object oriented code metrics.After doing the part 2 we got an overlook of software metrics.The studies well educated us about the metrics.

References

[1] J. Al Dallal, "Improving the applicability of object-oriented class cohesion metrics," *Information and Software Technology*, vol. 53, no. 9, pp. 914–928, 2011 (cit. on pp. 2, 3, 7, 8).

[2] ——, "Measuring the discriminative power of object-oriented class cohesion metrics," *IEEE Transactions on Software Engineering*, vol. 37, no. 6, pp. 788–804, 2011 (cit. on p. 2).

[3] J. Al Dallal and L. C. Briand, "An object-oriented high-level design-based class cohesion metric," *Information and software technology*, vol. 52, no. 12, pp. 1346–1361, 2010 (cit. on pp. 1, 2, 7, 8).

[4] A. Aral and T. Ovatman, "Utilization of Method Graphs to Measure Cohesion in Object Oriented Software," in *2013 IEEE 37th Annual Computer Software and Applications Conference Workshops*, IEEE, 2013, pp. 505–510 (cit. on pp. 3, 7, 8).

[5] Moataz A. Ahmed and Hamdi A. Al-Jamimi, "Machine learning approaches for predicting software maintainability: A fuzzy-based transparent model," *IET Software*, vol. 7, no. 6, pp. 317–326, Dec. 2013, ISSN: 1751-8806 (cit. on pp. 3, 4, 7, 8).

[6] A. Kaur, K. Kaur, and K. Pathak, "Software maintainability prediction by data mining of software code metrics," in *2014 International Conference on Data Mining and Intelligent Computing (ICDMIC)*, Sep. 2014, pp. 1–6 (cit. on pp. 3, 4, 7, 8).

[7] N. P. S. Rathore and R. Gupta, "A novel coupling metrics measure difference between inheritance and interface to find better oop paradigm using c," in *2011 World Congress on Information and Communication Technologies*, IEEE, 2011, pp. 467–472 (cit. on pp. 4–8).

[8] S. R. Chidamber and C. F. Kemerer, "A metrics suite for object oriented design," *IEEE Transactions on software engineering*, vol. 20, no. 6, pp. 476–493, 1994 (cit. on pp. 4, 6).

[9] W. Li and S. Henry, "Object-oriented metrics that predict maintainability," *Journal of systems and software*, vol. 23, no. 2, pp. 111–122, 1993 (cit. on p. 4).

[10] A. Wirotyakun and P. Netisopakul, "Improving software maintenance size metrics a case study: Automated report generation system for particle monitoring in hard disk drive industry," in *2012 Ninth International Conference on Computer Science and Software Engineering (JCSSE)*, May 2012, pp. 334–339 (cit. on pp. 3, 4, 7, 8).

[11] K. A. Hasan and M. S. Hasan, "Principal component analysis of coupling measures for developing high quality object oriented software," in *International Conference on Computer and Communication Engineering (ICCCE'10)*, IEEE, 2010, pp. 1–6 (cit. on pp. 5–8).

[12] M. Gethers and D. Poshyvanyk, "Using relational topic models to capture coupling among classes in object-oriented software systems," in *2010 IEEE International Conference on Software Maintenance*, IEEE, 2010, pp. 1–10 (cit. on pp. 5–8).

[13] L. C. Briand, P. Devanbu, and W. Melo, "An investigation into coupling measures for c++," in *ICSE*, vol. 97, 1997, pp. 412–421 (cit. on p. 6).

[14] M. O. Elish, "Exploring the relationships between design metrics and package understandability: A case study," in *2010 IEEE 18th International Conference on Program Comprehension*, IEEE, 2010, pp. 144–147 (cit. on pp. 7, 8).