Effectiveness of Coupling Metrics in Identifying Change-Prone Object-Oriented Classes

Mahmoud O. Elish[1] **and Ali A. Al-Zouri**[2]
[1]Information and Computer Science Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, elish@kfupm.edu.sa
[2]IT Development Department, IT Head Office, SABB, Riyadh, Saudi Arabia, ali.alzouri@sabb.com

Abstract - *This paper empirically evaluate the effectiveness of a set of coupling metrics, identified in a literature survey, as early indicators of change-prone classes from one release to the next in object-oriented software evolution process. Several hypotheses were tested, and different logistic regression models were constructed for predicting change-prone classes. Coupling metrics were found to be statistically correlated with change-proneness of classes. The results also indicate that a prediction model based on coupling metrics is generally more accurate than a model based on cohesion metrics. Furthermore, a prediction model based on import coupling metrics is more accurate than a model based on export coupling metrics. We also found that the coupling metrics in the C&K suite are not necessarily more accurate than other metrics in the suite in identifying change-prone classes in evolving object-oriented software. Moreover, there is no confounding effect of class size in the validity of some of the investigated coupling metrics.*

Keywords: Coupling metrics; object-oriented software; software evolution.

1 Introduction

Identification of change-prone classes in an object-oriented software system is an important activity especially in large and complex systems. A great majority of changes is rooted in a small proportion of classes [14]. In other words, around 80% of the changes are actually rooted in around 20% of the classes. This phenomenon has been known as Pareto's Law (also as the 80:20 rule) [14]. Identifying change-prone classes can therefore be very useful in guiding software maintenance and evolution; distributing resources more efficiently and effectively; and thus enabling the project manager and his team to focus their effort and attention on the change-prone classes during the evolution process.

Many coupling metrics have been proposed in the literature. Most of them have been empirically validated by exploring the relationships between them and certain software quality attributes such as fault-proneness [3, 4, 8, 13, 16], testability [5], reusability [12], and maintenance effort [15]. Results from the previous empirical studies indicate that coupling metrics are good indicators of several quality attributes.

This paper aims to empirically evaluate a set of coupling metrics, identified in a literature survey [1], as early indicators of change-prone classes from one release to the next in evolving object-oriented software. This set of metrics covers comprehensively different type of interaction and relationships within a class and between classes. Metrics that are correlated with class change-proneness will be essential for objective and quantitative identification and characterization of change-prone classes and for effective prediction of change-proneness during software evolution throughout the releases. In addition, these metrics will provide useful guidance to practitioners involved in development and maintenance of evolving large-scale software.

The rest of this paper is organized as follows. Section 2 reviews related works. Section 3 describes the empirical study and discusses its results. Section 4 concludes the papers and suggests directions for future work.

2 Related Work

In the literature, the relationships between coupling metrics and several software quality attributes have been explored. Briand et al. [4] explored the relationships between design measures and fault-proneness of classes. They found that most of the coupling metrics are good predictors of fault-proneness. Gyimothy et al. [13] found that CBO (coupling between object classes) and RFC (response set for class) metrics are good predictors of fault-proneness. El-Emam et al. [8] found that OCMEC (export coupling based on class-method interaction) and OCAEC (export coupling based on class-attribute interaction) metrics have strong association with fault-proneness.

Bruntink and Deursen [5] empirically studied the relationship between several object-oriented metrics and software testability. RFC metric was one of them. Testability was measured in terms of test efforts which were taken from the size of test suites. Results showed that there is significant relationship between RFC and testability.

Gui and Scott [12] empirically studied the relationship between CBO, RFC, MPC (message passing coupling) and DAC (data abstraction coupling) metrics and software reusability. Reusability was measured by the number of lines of code that were added, modified or deleted in order to extend some function in the system. Their results indicated that there

is strong relationship between coupling metrics and reusability, especially CBO and RFC.

Li and Henry [15] studied coupling metrics (CBO, RFC, MPC, DAC) with respect to maintenance effort, which was measured by the number of lines changed per class. A line change could be an addition or a deletion. Their results showed that there is strong relationship between coupling metrics and maintenance effort.

Wilkie and Kitchenham [18] studied two versions of CBO metric according to direction of coupling, CBO(backward) to count export coupling and CBO(forward) to count import coupling, with respect to how the final version of the system differs from the first version. Changes were counted as the number of changes per class. They found that CBO(forward) is a good predictor of change-prone classes. CBO(backward) had slightly lower correlation with change-proneness and found to be not significant.

Koru and Liu [14] tested and validated the Pareto's Law which implies that a great majority (around 80%) of changes are rooted in a small proportion (around 20%) of the classes. They also identified and characterized the change-prone classes in two products (KOffice and Mozilla) by producing tree-based models. Their results from both systems strongly supported Pareto's law. The resulting tree-based model consists of several metrics and OCMEC and OCMIC (import coupling based on class-method interaction) coupling metrics were part of the model.

In addition to the coupling metrics, other metrics and approaches have been proposed in the literature to predict change-proneness. For example, Tsantalis et al. [17] proposed a probabilistic approach to estimate the change proneness of an object-oriented design. Moreover, Elish and Al-Khiaty [9] proposed a suite of metrics for quantifying historical changes to predict future change-prone classes in object-oriented software.

This study explores the relationships between a comprehensive set of coupling metrics and class change-proneness, whereas previous studies have been limited to few coupling metrics. In addition, we are assessing these metrics in a software evolution context throughout the releases.

3 Empirical Study

The objective of this study is to empirically investigate the relationships between a set of coupling metrics and the change-proneness of classes in evolving object-oriented software. In other words, we want to evaluate the capability of these metrics as early indicators of change-prone classes from one software release to the next.

3.1 Independent and Dependent Variables

The main independent variables are 22 coupling metrics, which were identified in a literature survey on object-oriented design measures [1]. All of them are static and language independent. For comparison purposes, cohesion metrics (identified in a literature survey on object-oriented design measures [2]) and C&K metrics (Chidamber and Kemerer [6]) were used as other independent variables to build other prediction models. Definitions of the coupling, cohesion and C&K metrics are provided in [1], [2] and [6] respectively.

As a dependent variable, we used a dichotomous variable (named CHANGE) that indicates whether or not a class was changed from one software release to the next release. A class is considered changed if at least one of its lines of source code was changed or deleted, or at least one new line of code was added to it. Comment and blank lines were excluded.

3.2 Hypotheses

- Hypothesis 1: There is a statistically significant correlation between each of the investigated coupling metrics and change-proneness of classes in evolving object-oriented software.
- Hypothesis 2: A prediction model based on the investigated coupling metrics is more accurate than a model based on cohesion metrics in identifying change-prone classes in evolving object-oriented software.
- Hypothesis 3: A prediction model based on import coupling metrics is more accurate than a model based on export coupling metrics in identifying change-prone classes in evolving object-oriented software.
- Hypothesis 4: Coupling metrics in the C&K suite are more accurate than other metrics in the suite in identifying change-prone classes in evolving object-oriented software.

3.3 Software Systems Analyzed

Two multi-release object-oriented software systems of different size and from different application domains were analyzed in this study: Stellarium[1] and LabPlot[2]. Both systems are open source systems and written in C++ programming language. Stellarium is an educational system for astronomy, and the goal of the system is to render 3D photo-realistic skies in real time with OpenGL. It displays stars, constellations, planets, nebulas and others things like ground, landscape, atmosphere, etc. LabPlot is a desktop environmental system for visualization data. The goal of the system is data plotting and function analysis.

All releases of both systems were analyzed from the first release to the most recent release at the time of this study. Stellarium system has seven releases, whereas LabPlot system has six releases. Release numbers and size measures of these two systems are provided in Table 1. The percentage of changed and unchanged classes from one release to the next are shown in Figure 1 and Figure 2 for Stellarium and LabPlot systems respectively.

[1] www.sourceforge.net/projects/stellarium/
[2] www.sourceforge.net/projects/labplot/

Table 1. Release numbers and size measures

Stellarium system			LabPlot system		
Release Number	Number of classes	Lines of code (LOC)	Release Number	Number of classes	Lines of code (LOC)
0.6.2	102	9499	1.4.0	35	1399
0.7.1	117	12774	1.4.1	38	1794
0.8.0	140	16103	1.5.0	44	1519
0.8.1	147	17004	1.5.1	49	2173
0.8.2	154	20141	1.6.0	53	2902
0.9.0	191	22334	2.0.0	24	852
0.9.1	190	12140			

It can be observed that the size of the Stellarium system is bigger than the LabPlot system in terms of the number of classes and lines of code (LOC). Moreover, the number of classes and LOC are increasing from one release to the next in both systems, from the first release to the release before the last one, which is most likely due to the addition of new features and requirements. However, there is a significant drop in the number of classes in the LabPlot system, and a significant drop in LOC in both systems in the last release. This suggests that a major refactoring was performed. The percentages of changed classes that are provided in Figure 1 and Figure 2 include deleted classes. The differences in the percentages of changed classes between releases and between the two systems will be helpful in evaluating the accuracy of the prediction models and preventing biased results.

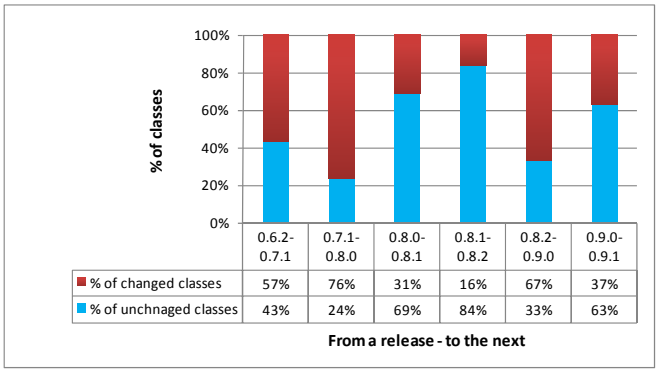


Figure 1. Percentage of changed and unchanged classes from one release to the next in Stellarium system

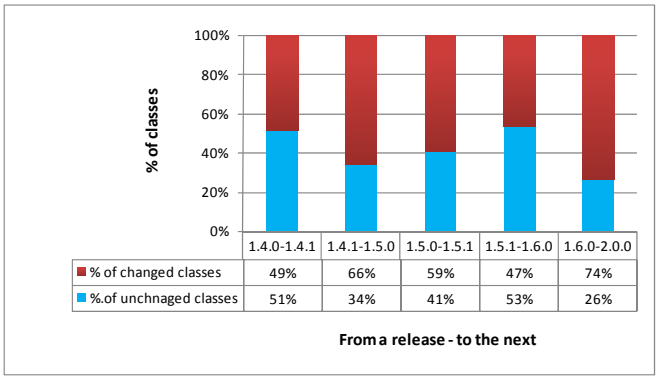


Figure 2. Percentage of changed and unchanged classes from one release to the next in LabPlot system

3.4 Data Collection

Columbus[3] tool [11] was used in this study to collect all independent variables, i.e., coupling, cohesion and C&K metrics. ExamDiff Pro[4] tool was used to collect the dependent variable by comparing classes from one release to the next. Comment and blank lines were excluded in class comparison. Our approach is release by release prediction where we train the prediction model using all releases from the first release to release i-1 and then test it using release i. Since there are seven releases in the Stellarium system and six in the LabPlot system, five and four pairs of training and testing datasets were produced from each system respectively; no training dataset for the first release and no testing dataset for the last release.

3.5 Descriptive Statistics

Table 2 and Table 3 provide descriptive statistics of the 22 coupling metrics under investigation, which were collected from Stellarium and LabPlot systems, respectively. In general, the LabPlot system has lower coupling than the Stellarium system. Only coupling metrics that have more than five non-zero values were considered for further analysis since those metrics that have less than five non-zero values have low variance (almost zero) and they may mislead the analysis. This strategy was also applied in the previous studies [4, 8]. Metrics with less than five non-zero values are highlighted in both tables.

3.6 Correlation Analysis

We performed Spearman's rank-order correlation analysis, at 99% confidence level, between the dependent variable (CHANGE) and each of the investigated coupling metrics. Table 4 reports the results over all the releases of Stellarium and LabPlot systems, respectively, in terms of the correlations coefficients and p-values. All metrics in both systems, except OCAEC in Stellarium, were found to be significantly correlated (p-value < 0.01) with CHANGE. However, OCAEC metric in Stellarium was significantly correlated with CHANGE but at 95% confidence level (p-value < 0.05). Hypothesis 1 is therefore accepted.

It can be observed that all coupling metrics, except IH-ICP (information-flow-based inheritance coupling) metric, are positively correlated with class change-proneness. This indicates that the more the coupling of a class with the rest of the system the higher the probability that the class will change. In case of IH-ICP metric, the negative correlation between it and class change-proneness suggests that the more the coupling of a class with its ancestors through methods invocations the less likely the class will change. This observation can be explained; ancestor classes are expected to be highly stable [10], and thus change propagations from them to their descendant classes are less likely to occur.

[3] http://www.frontendart.com
[4] http://www.prestosoft.com/edp_examdiffpro.asp

Table 2. Descriptive statistics of coupling metrics in Stellarium system

Metric	Mean	Std Dev.	Max	Min
CBO	2.24	3.91	47	0
CBO1	1.63	3.63	45	0
RFC	16.4	37.49	447	0
RFC1	14.85	34.02	424	0
MPC	16.38	74.08	1309	0
DAC	2.19	12.04	165	0
DAC1	1.09	3.22	34	0
ICP	29.28	141.74	2357	0
IH-ICP	3.13	10.57	109	0
NIH-ICP	26.15	141.51	2357	0
ACAIC	0	0	1	0
DCAEC	0	0	1	0
ACMIC	0	0	2	0
DCMEC	0	0	3	0
AMMIC	0	0	0	0
DMMEC	0	0	0	0
OMMIC	0	0	0	0
OMMEC	0	0	0	0
OCAIC	2.11	11.36	165	0
OCAEC	1.89	5.8	117	0
OCMIC	1.93	4.05	32	0
OCMEC	1.81	7.45	71	0

Table 3. Descriptive statistics of coupling metrics in LabPlot system

Metric	Mean	Std Dev.	Max	Min
CBO	0.32	0.75	4	0
CBO1	0.31	0.75	4	0
RFC	9.31	17.18	124	0
RFC1	9.31	17.18	124	0
MPC	4.13	15.45	124	0
DAC	0.97	2.21	13	0
DAC1	0.73	1.43	7	0
ICP	4	18.29	168	0
IH-ICP	0	0	0	0
NIH-ICP	4	18.29	168	0
ACAIC	0	0	0	0
DCAEC	0	0	0	0
ACMIC	0	0	0	0
DCMEC	0	0	0	0
AMMIC	0	0	0	0
DMMEC	0	0	0	0
OMMIC	0	0	0	0
OMMEC	0	0	0	0
OCAIC	0.97	2.21	13	0
OCAEC	0.95	2.37	12	0
OCMIC	0.97	2.28	14	0
OCMEC	0.93	2.59	18	0

Table 4. Spearman correlation results

Metric	Stellarium		LabPlot	
	Corr. Coef.	p-value	Corr. Coef.	p-value
CBO	0.14	<0.01	0.26	<0.01
CBO1	0.20	<0.01	0.24	<0.01
RFC	0.19	<0.01	0.48	<0.01
RFC1	0.22	<0.01	0.48	<0.01
MPC	0.20	<0.01	0.21	<0.01
DAC	0.27	<0.01	0.43	<0.01
DAC1	0.27	<0.01	0.44	<0.01
ICP	0.18	<0.01	0.24	<0.01
IH-ICP	-0.16	<0.01	---	---
NIH-ICP	0.21	<0.01	0.24	<0.01
OCAIC	0.28	<0.01	0.43	<0.01
OCAEC	0.07	0.03	0.30	<0.01
OCMIC	0.26	<0.01	0.43	<0.01
OCMEC	0.17	<0.01	0.28	<0.01

3.7 Prediction Models

Different logistic regression models, standard models based on maximum likelihood estimation, were constructed in this study for predicting change-prone classes from one software release to the next. In the following subsections, we compare the accuracy (correct classification rate) of coupling-based model vs. cohesion-based model; import coupling-based model vs. export coupling-based model; and among the models that are based on each metric in the C&K suite.

3.7.1 Coupling-based Model vs. Cohesion-based Model

In a modular design, each individual class should have high cohesion within the class and low coupling with other classes. It is interesting to investigate which one of these two class characteristics (coupling and cohesion) are better predictors of its change-proneness. Accordingly, two prediction models were constructed for identifying change-prone classes for each system (Stellarium and LabPlot). One model was based on the coupling metrics as independent variables; and the other was based on the cohesion metrics as independent variables. The prediction was performed release by release where the models were trained using all releases from the first release to release i-1 and then tested using release i. The accuracy (correct classification rate) of each model for each release was calculated as well as the average accuracy over all the releases. Figure 3 and Figure 4 illustrate the accuracy curve of these models on Stellarium and LabPlot systems respectively. The horizontal axis represents the release number and the vertical axis represents the accuracy for identifying change-prone classes in that release.

Out of the five releases of Stellarium system, the accuracy of the coupling-based model was better than the accuracy of the cohesion-based model in two releases (0.8.0 and 0.8.1), and the accuracy of the cohesion-based model was better in two releases as well (0.7.1 and 0.8.2). In release 0.9.0, both models have almost the same accuracy. However, the best achieved accuracy by the coupling-based model throughout the releases was 70.1% compared to 68.6% which was achieved by the cohesion-based model. In LabPlot System, the coupling-based model had better accuracy than the cohesion-based model in three releases, and in release 1.5.1 in which the accuracy of the cohesion-based model was better, the difference was only 2%. The highest accuracy of the coupling-based model reached 79.5% in release 1.5.0, while the highest accuracy of the cohesion-based model was 75.5%.

It was observed that the models based on coupling metrics outperform the models based on cohesion metrics in both systems in terms of the average accuracy across the releases. In Stellarium system, the average accuracy of the coupling-based model was 62.2%, whereas it was 58.8% by the cohesion-based model. In LabPlot System, the average accuracy of the coupling-based model was 72.2%, whereas it was 68% by the cohesion-based model. These results suggest the acceptance of hypothesis 2.

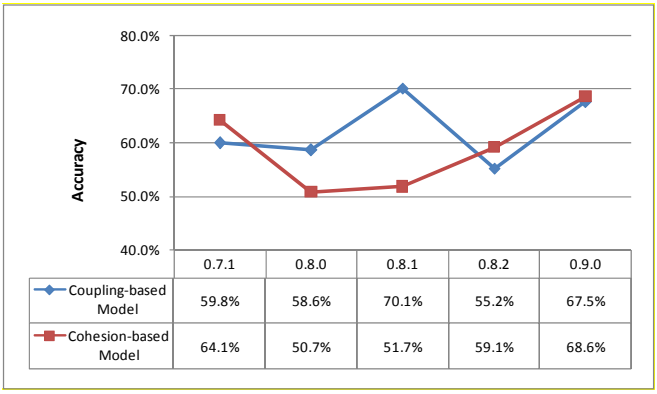


Figure 3. Accuracy of coupling-based model and cohesion-based model for Stellarium system

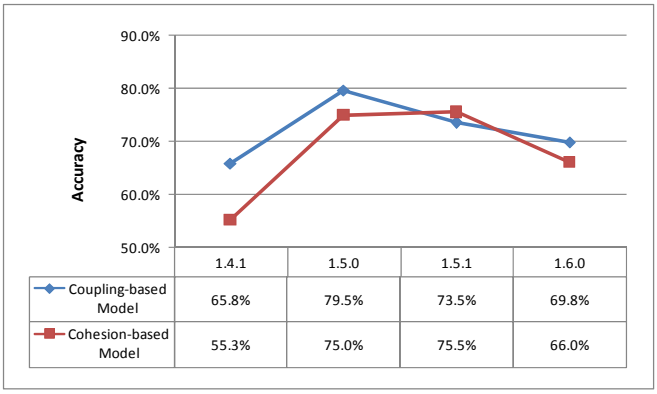


Figure 4. Accuracy of coupling-based model and cohesion-based model for LabPlot system

3.7.2 Import Coupling-based Model vs. Export Coupling-based Model

A class could depend on some other classes, and some classes could depend on it. Import and export coupling metrics measure these dependencies respectively. Table 5 lists import and export coupling metrics. We investigated which one of these two types of coupling metrics (import and export) is better predictor of class change-proneness. Two prediction models were constructed for identifying change-prone classes for each system (Stellarium and LabPlot). One model was based on the import coupling metrics as independent variables; and the other was based on the export coupling metrics as independent variables. The prediction was also performed release by release where the models were trained using all releases from the first release to release i-1 and then tested using release i. The accuracy (correct classification rate) of each model for each release was calculated as well as the average accuracy over all the releases. Figure 5 and Figure 6 illustrate the accuracy curve of these models on Stellarium and LabPlot systems respectively.

Table 5. Import and export coupling metrics

Import Coupling Metrics	CBO, CBO1, RFC, RFC1, MPC, DAC, DAC1, ICP, IH-ICP, NIH-ICP, OCAIC, OCMIC
Export Coupling Metrics	CBO, CBO1, OCAEC, OCMEC

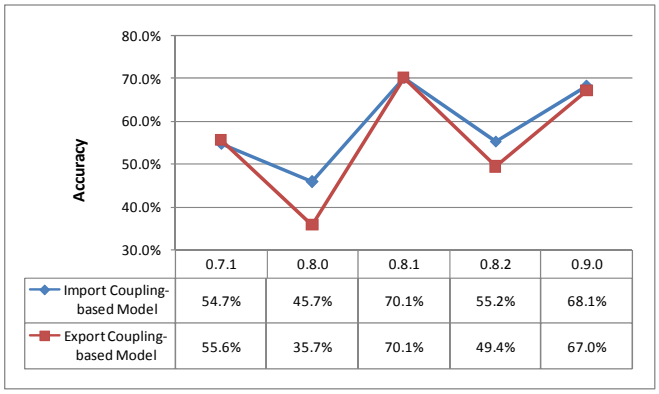


Figure 5. Accuracy of import coupling-based model and export coupling-based model for Stellarium system

In Stellarium system, the import coupling-based model outperformed the export coupling-based model in three releases (0.8.0, 0.8.2 and 0.9.0). In release 0.8.1, both models had the same accuracy (70.1%), and in release 0.7.1, the accuracy of the export coupling-based model was better than the accuracy of the import coupling-based model by 1% only. The average accuracy of the import coupling-based model was 58.7%, whereas it was 55.5% by the export coupling-based model. In LabPlot system, the import coupling-based model outperformed the export coupling-based model in all releases. The highest accuracy of the import coupling-based model was 81.8%. The average accuracy of the import coupling-based model was 71.9%, which is very high compared to the average accuracy of the export coupling-based model (55.5%).

These results indicate that the models based on import coupling metrics outperform the models based on export coupling metrics in both systems in terms of the average accuracy. We therefore accept hypothesis 3.

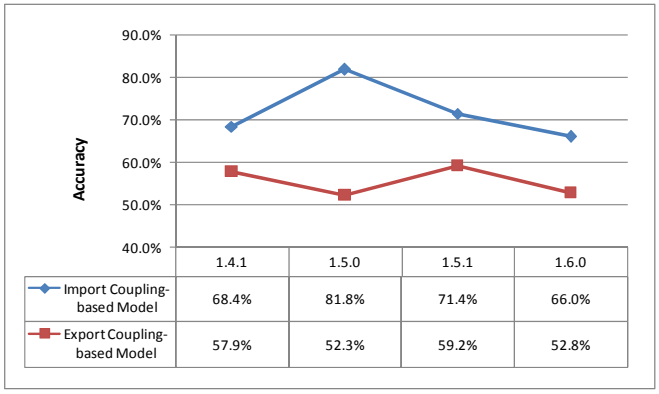


Figure 6. Accuracy of import coupling-based model and export coupling-based model for LabPlot system

3.7.3 Models based on Each Metric in C&K Suite

The C&K suite of metrics consists of six metrics [6]: CBO (coupling between object classes), RFC (response set for class), LCOM (lack of cohesion in methods), WMC (weighted methods per class), DIT (depth of inheritance tree), NOC (number of children). Two of them are coupling metrics

which are CBO and RFC. The following analysis aimed to compare the prediction performance of each of these two metrics in identifying change-prone classes against each of the other four metrics in the C&K suite.

For Stellarium system, six prediction models were built; each based on one of the six metrics in the C&K suite. For LabPlot system, only four models were built since there is no inheritance in LabPlot system and thus DIT and NOC metrics were excluded. Figure 7 and Figure 8 illustrate the accuracy curve of these models on Stellarium and LabPlot systems respectively.

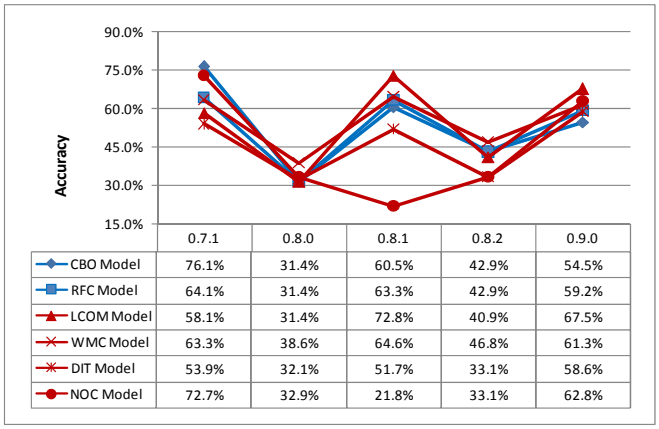
We now compare the accuracy of the coupling metrics (CBO and RFC) against the other four metrics (LCOM, WMC, DIT, and NOC) in the C&K suite. Out of the five releases of Stellarium system, the LCOM model was the best in two releases, the WMC model was best in two other releases, and the CBO model was the best in one release. The average accuracy achieved by each of the CBO, RFC, LCOM, WMC, DIT and NOC models was 53.1%, 52.2%, 54.2%, 54.9%, 45.9% and 44.6% respectively. It can be observed that the average accuracy achieved by each of the CBO, RFC, LCOM, WMC models is competitive (less than 3% differences). However, the performance of the DIT and NOC models is noticeably lower than the other metrics. This suggests that inheritance-based metrics are not good indicators of change-prone classes. In LabPlot system, the average accuracy achieved by each of the CBO, RFC, LCOM and WMC models was 57.4%, 72.6%, 67.5% and 61.6% respectively. The RFC model outperformed all other three models in all release except the last release where it was outperformed by the WMC model. The CBO model, however, has the lowest accuracy on average. These results suggest the rejection of hypothesis 4.

3.8 Confounding Effect of Class Size

We also investigated the potential confounding effect of class size on the validity of the investigated coupling metrics with respect to their relationship with change-proneness. This helps to determine if the relationship between these metrics and change-proneness of classes is real regardless of the class size. We followed the same approach suggested by El Emam et al. [7] to determine whether there is a confounding effect of class size. The idea is to include a size metric (LOC in this study) as another independent variable, in addition to a coupling metric, in a prediction model for change-proneness. If there is a statistically significant difference in the results with and without the size metric, then this indicates a confounding effect of size on the validity of that coupling metric.

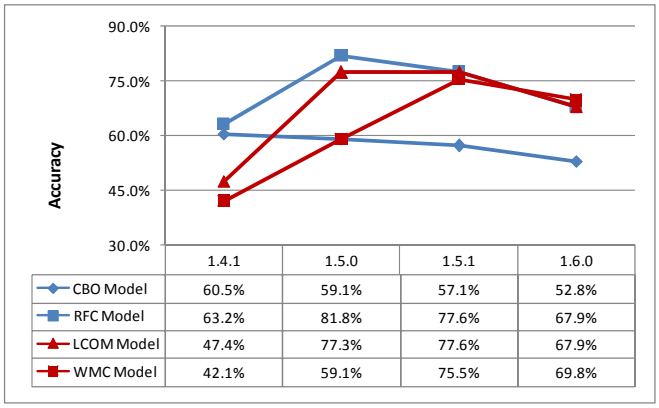
In order to examine the confounding effect of size on each coupling metric Ci, two prediction models for class change-proneness were built: (i) one based on the coupling metric Ci only and (ii) one based on the Ci and LOC. Wilcoxon nonparametric test was then performed, at 95% confidence level, to evaluate the significance difference between the results obtained from each model. Table 6 reports the Z statistic values and p-values for the Wilcoxon test results. If the p-value is less than 0.05, then there is a confounding effect of size on the validity of the corresponding coupling metrics.

It can be observed that, in LabPlot system, there is no confounding effect of class size on the validity of all of the coupling metrics. However, in Stellarium system, the associations between some coupling metrics (i.e. CBO, CBO1, RFC, ICP, IH-ICP, OCAEC, and OCMEC) and change-proneness of classes disappear after controlling for class confounder.



	0.7.1	0.8.0	0.8.1	0.8.2	0.9.0
CBO Model	76.1%	31.4%	60.5%	42.9%	54.5%
RFC Model	64.1%	31.4%	63.3%	42.9%	59.2%
LCOM Model	58.1%	31.4%	72.8%	40.9%	67.5%
WMC Model	63.3%	38.6%	64.6%	46.8%	61.3%
DIT Model	53.9%	32.1%	51.7%	33.1%	58.6%
NOC Model	72.7%	32.9%	21.8%	33.1%	62.8%

Figure 7. Accuracy of models based on each metric in C&K suite for Stellarium system



	1.4.1	1.5.0	1.5.1	1.6.0
CBO Model	60.5%	59.1%	57.1%	52.8%
RFC Model	63.2%	81.8%	77.6%	67.9%
LCOM Model	47.4%	77.3%	77.6%	67.9%
WMC Model	42.1%	59.1%	75.5%	69.8%

Figure 8. Accuracy of models based on each metric in C&K suite for LabPlot system

3.9 Limitations

The results of this study were obtained by analyzing two C++ open source systems. Although these systems are normal open source systems and representing different size and application domains, more studies should be conducted to further support the results and to accumulate knowledge.

This study focused on static coupling metrics. It did not explore the relationship between dynamic coupling metrics and change-processes of classes. This study was also focused on change-proneness of classes, i.e., whether or not a class was changed from one software release to the next release. The capability of coupling metrics in estimating change size and density was not evaluated.

In addition, this study was a regression and correlation study. Association between most of the investigated coupling metrics and change-proneness of classes was observed but causality of the association cannot be claimed. In other words, claims cannot be made as altering a class to reduce its coupling would necessarily decrease its likelihood of being changed.

Table 6. Wilcoxon test for confounding effect of class size

Metric	Stellarium		LabPlot	
	Z	p-value	Z	p-value
CBO	3.29	<0.05	0.67	0.50
CBO1	2.58	<0.05	1.80	0.07
RFC	3.51	<0.05	0.53	0.59
RFC1	1.62	0.11	0.53	0.59
MPC	1.61	0.11	0	1
DAC	1.22	0.22	0	1
DAC1	1.60	0.11	0	1
ICP	3.00	<0.05	0	1
IH-ICP	5.17	<0.05	---	---
NIH-ICP	1.87	0.06	0	1
OCAIC	0.39	0.69	0	1
OCAEC	4.95	<0.05	1.23	0.22
77OCMIC	1.30	0.19	0	1
OCMEC	4.17	<0.05	0.67	0.50

4 Concluding Remarks

In this study, empirical evaluation of coupling metrics was performed to explore their capability to identify change-prone classes in evolving object-oriented software systems. Coupling metrics were found to be statistically correlated with change-proneness of classes. Different logistic regression models were constructed for predicting change-prone classes from one software release to the next. The results indicate that a prediction model based on coupling metrics is generally more accurate than a model based on cohesion metrics. Furthermore, a prediction model based on import coupling metrics is more accurate than a model based on export coupling metrics. We also found that the coupling metrics in the C&K suite are not necessarily more accurate than other metrics in the suite in identifying change-prone classes in evolving object-oriented software. Moreover, there is no confounding effect of class size in the validity of some of the investigated coupling metrics.

There are several directions for future work. One direction is to conduct more studies that include software systems written in different programming languages (Java, C#, etc.), and also proprietary software systems and compare the results. Another direction is to empirically evaluate dynamic coupling metrics and compare them against static coupling metrics. In addition, it would be interesting to evaluate the capability of coupling metrics in estimating change size and density, and to explore the relationship between coupling metrics and other software quality attributes. Finally, a comparative study of different computational intelligence models for identifying change-prone classes could be also conducted.

5 References

[1] L. Briand, J. Daly, and J. Wust, "A Unified Framework for Coupling Measurement in Object-Oriented Systems," *IEEE Transactions on Software Engineering,* vol. 25, no. 1, pp. 91-121, 1999.

[2] L. Briand, J. Daly, and J. Wüst, "A Unified Framework for Cohesion Measurement in Object-Oriented Systems," *Empirical Software Engineering,* vol. 3, no. 1, pp. 65-117, 1998.

[3] L. Briand, P. Devanbu, and W. Melo, "An Investigation into Coupling Measures for C++," in *19th Int'l Conf. Software Eng., ICSE'97*, pp. 412-421, 1997.

[4] L. Briand, J. Wüst, J. Daly, and V. Porter, "Exploring the Relationships between Design Measures and Software Quality in Object-Oriented Systems," *Journal of Systems and Software,* vol. 51, no. 3, pp. 245-273, 2000.

[5] M. Bruntink and A. Deursen, "An empirical study into class testability," *Journal of Systems and Software,* vol. 79, pp. 1219-1232, 2006.

[6] S. Chidamber and C. Kemerer, "A Metrics Suite for Object Oriented Design," *IEEE Transactions on Software Engineering,* vol. 20, no. 6, pp. 476-493, 1994.

[7] K. El-Emam, S. Benlarbi, N. Goel, and S. Rai, "The Confounding Effect of Class Size on the Validity of Object-Oriented Metrics," *IEEE Transactions on Software Engineering,* vol. 27, pp. 630-650, 2001.

[8] K. El-Emam, W. Melo, and J. Machado, "The Prediction of Faulty Classes Using Object-Oriented Design Metrics," *Journal of Systems and Software,* pp. 63-75, 2001.

[9] M. Elish and M. Al-Khiaty, "A suite of metrics for quantifying historical changes to predict future change-prone classes in object-oriented software," *Journal of Software: Evolution and Process,* vol. 25, no. 5, pp. 407-437, 2013.

[10] M. Elish and D. Rine, "Investigation of Metrics for Object-Oriented Design Logical Stability," in *7th IEEE European Conference on Software Maintenance and Reengineering,* pp. 193-200, 2003.

[11] R. Ferenc, A. Besze´des, M. Tarkiainen, and T. Gyimo´thy, "Columbus—reverse engineering tool and schema for C++," in *IEEE International Conference on Software Maintenance,* pp. 172–181, 2002.

[12] G. Gui and P. Scott, "Ranking reusability of software components using coupling metrics," *Journal of Systems and Software,* vol. 80, pp. 1450-1459, 2007.

[13] T. Gyimothy, R. Ferenc, and I. Siket, "Empirical Validation of Object-Oriented Metrics on Open Source Software for Fault Prediction," *IEEE Transactions on Software Engineering,* vol. 31, no. 10, pp. 897-910, Oct. 2005 2005.

[14] A. Koru and H. Liu, "Identifying and characterizing change-prone classes in two large-scale open-source products," *Journal of Systems and Software,* vol. 80, pp. 63-73, 2007.

[15] W. Li and S. Henry, "Object-Oriented Metrics that Predict Maintainability," *Journal of Systems and Software,* vol. 23, no. 2, pp. 111-122, 1993.

[16] H. Olague, L. Etzkorn, S. Gholston, and S. Quattlebaum, "Empirical Validation of Three Software Metrics Suites to Predict Fault-Proneness of Object-Oriented Classes Developed Using Highly Iterative or Agile Software Development Processes," *IEEE Transactions on Software Engineering,* vol. 33, no. 6, pp. 402-419, 2007.

[17] N. Tsantalis, A. Chatzigeorgiou, and G. Stephanides, "Predicting the Probability of Change in Object-Oriented Systems," *IEEE Transactions on Software Engineering,* vol. 31, no. 7, pp. 601-614, 2005.

[18] F. Wilkie and B. Kitchenham, "An Investigation of Coupling, reuse and Maintenance in a Commercial C++ Application," *Information and Software Technology,* vol. 43, pp. 801-812, 2001.