Do Software Categories Impact Coupling Metrics?

Lucas Batista Leite de Souza, Marcelo de Almeida Maia
Computer Science Department
Federal University of Uberlândia
Uberlândia, Brazil
lucas.facom.ufu@gmail.com, marcmaia@facom.ufu.br

***Abstract*—Software metrics is a valuable mechanism to assess the quality of software systems. Metrics can help the automated analysis of the growing data available in software repositories. Coupling metrics is a kind of software metrics that have been extensively used since the seventies to evaluate several software properties related to maintenance, evolution and reuse tasks. For example, several works have shown that we can use coupling metrics to assess the reusability of software artifacts available in repositories. However, thresholds for software metrics to indicate adequate coupling levels are still a matter of discussion. In this paper, we investigate the impact of software categories on the coupling level of software systems. We have found that different categories may have different levels of coupling, suggesting that we need special attention when comparing software systems in different categories and when using predefined thresholds already available in the literature.**

***Index Terms*—Software categories, coupling metrics, Java.**

I. Introduction

One important factor that should be considered during software design is module coupling, which measures the dependency level between two modules. According Martin [1], a design with highly interdependent modules tends to be rigid and difficult to reuse and maintain. Several studies in the literature have shown that module coupling directly impacts maintenance, evolution, and reuse of software components [2], [3], [4].

Open source software repositories play a major role in mining software repositories because they offer thousands of projects, within different domains, opening unprecedented opportunities for research. One of the most popular repositories is Sourceforge[1], accounting for almost 3.5 million registered users, almost 325,000 projects and approximately 5,000 commits/day[2]. In some repositories, projects are classified in different categories, which are related to different application domains. For example, in Sourceforge there are 10 major categories, each one with several subcategories.

Considering the huge amount of information in software repositories, software metrics have gained renewed interest because they can enable automated large-scale quality analysis of software. Several works have used generic thresholds for OO metrics to establish evaluation criteria of software, for example, to detect bad smells. These thresholds are generic in the sense they were established ignoring the software category [5], [6], [7], [8], [9]. However, the use of generic thresholds may not be adequate for the comparison of software systems belonging to different categories, i.e., a threshold that detect a bad smell in one category could not be same in other category.

In another work, Ferreira et al. [10] established thresholds for software metrics comparing software in different categories, but they did not find difference that could indicate necessity to consider different thresholds for different categories. This study was performed in relatively small scale with only 40 Java projects. Our hypothesis is that a larger scale study with a careful characterization of samples for different domains could indicate significant differences between metric values of different domains. Our scope in this study is limited only to coupling metrics. We carried out an experiment comparing Java projects from the 10 major categories of Sourceforge.

This paper is organized as follows. Section II presents the methodology used to conduct this study. Section III presents and discusses the results, and Section IV concludes the paper.

II. Methodology

In this section, we state our research question, review the coupling metrics used in this study, define the criteria to choose the sample of projects that will be analyzed, and define the data analysis strategy that will be performed to answer the research question.

A. Research Question

We state the following research question for this study:

- RQ1: Does the major category of a software is related to the coupling level of their modules?

B. Coupling Metrics

There are several coupling metrics for object-oriented software. In this study, we considered five metrics that were chosen because each of them is related to different aspects of the coupling notion. These metrics are described in the sequel.

CBO (Coupling Between Objects): this metric is part of the well-known Chidamber and Kemerer suite [11]. The coupling for a class is measured with the number of other classes to which it is coupled. A class X is coupled to a class Y if X uses one or more attributes of Y or if X invokes at least one method of Y.

[1] http://sourceforge.net
[2] http://sourceforge.net/blog/sourceforge-myths

Accepted for publication by IEEE. © 2013 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/ republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

CBO*: we defined a variant of CBO, which we call CBO*, where the coupling of a class is the number of attribute uses and method invocations from this class to other classes.

DAC (Data Abstraction Coupling): proposed by Li and Henry [12], defines the coupling for a class as the number of attributes of this class declared with types defined from other classes.

ATFD (Access to Foreign Data): proposed by Marinescu [5], [13], this metric consider the coupling of a class equal the number of external classes from which that class accesses attributes, directly or via accessor methods.

Afferent Coupling (Ca): proposed by Martin [1], this metric defines the coupling of a class with the number of other classes that refer to it.

In this work, we used iPlasma [14] to collect these metric values for the classes of selected Java projects.

C. Repository / Projects

The repository used in this work was Sourcerer [15], which contains 18,826 Java projects collected from the following original repositories: Apache, Java.net, Google Code and Sourceforge. Several selection criteria were applied in this repository. First, because our study considers that a software project should belong to just one category we decided to use Sourceforge projects because it was possible to retrieve the major category for each project, from 10 predefined categories. From the 18,826 Sourcerer projects, 9,969 were collected from Sourceforge. From these projects we excluded empty projects which contained no *.java* file, remaining 6,632 non-empty projects. From these projects, we excluded more 3,559 projects, which were in one of the following situations: the iPlasma tool could not completely collect the metric values; the project could not be classified in one of the 10 major Sourceforge categories; or the project belonged to more than one category, because those projects would interfere in our research question analysis.

The obtained list included 3,073 projects. The 10 major categories defined in Sourceforge and considered in this study are: *Science & Engineering* (SE), *Audio & Video* (AV), *Communications* (C), *Business & Enterprise* (BE), *Graphics* (Gr), *Games* (Ga), *Development* (D), *System Administration* (SA), *Security & Utilities* (Sec), *Home & Education* (HE). Each one defines sub-categories not considered in this study.

Table I shows, for each category, the percent number of projects in each range of size (number of classes). We can observe that for all categories, most projects contain between two and 200 classes. In order to consider a more homogeneous sample between the different categories, we decided to consider only projects in this range. Table II shows the absolute number of projects in each category, in the range *[2,200]*.

D. Data Analysis Strategy

We performed data exploration and defined two tests to answer the research question.

Data Exploration: The goal of this exploration is produce a descriptive analysis with boxplots of the mean project coupling for the 10 categories. We calculated the mean coupling metric value for each project. The five chosen metrics were considered. For example, the Mean CBO for project P is the mean of CBO of all classes of this project.

Test I: The goal of this test is to compare the mean coupling level of one category with the mean coupling level of the projects from all categories. So, we could observe if the mean coupling level of projects in one category differs from the coupling level of the universe of projects.

For each metric, the following procedure was conducted:

- For each project, we calculate the mean value of metric, as in the data exploration explained before;
- For each category, we calculate the median of the mean project values;
- For each category, we select the 75 projects whose means are nearest to the median. We defined this sample size because it is the number of projects in the smallest category (*Home & Education*), so the 10 categories have the same number of projects. This choice intends to provide fair sampling of the categories.
- For each category, we apply the Wilcoxon Rank Sum test, at a stringent significance level of 0.01, to compare the mean values of the 75 selected projects from that category with the mean values of all selected projects (75 projects were selected from each category and there are 10 categories, so there is a total of 750 selected projects).

The previous process was conducted for each coupling metric, i.e., five times.

Test II: The goal of this test is to compare mean coupling metric values of classes from different categories. While the previous test analyzes the mean metric values of the projects, this test consider in a category all classes from the 75 projects in that category. In other words, we assembled classes from different projects belonging to a same category instead of directly aggregating them on a project basis. We evaluate the mean value and standard deviation of the categories.

III. RESULTS

For the data exploration performed, Figures 1 to 5 show the boxplots of the mean values of each project, with each metric in a different plot. We can observe that there are differences between the categories in the maximum, median values and in the first and third quartiles as well. For all five metrics, the category *Games* has shown the highest medians. Considering the first and third quartiles, the category *Games* had the highest values, except for metric ATFD, where *Business* presented higher value. Concerning the maximum values, *Games* present the highest values for three from five metrics. On the other side, there are categories that stood out for lower values, especially *Development* and *Security*.

For Test I, Table III shows the results of the Wilcoxon Rank Sum test. We marked the cell with an `X´ where *p-value* < 0.01, i.e., there is a stringent difference in the medians of the two compared samples (for each cell, the two samples compared consisted of: 1) 75 projects selected from that category for a specific metric, and 2) 750 projects obtained from 75 projects from each category).

TABLE I. PERCENT OF PROJECTS BY SIZE RANGE

Category	[1,1]	[2,200]	[201,400]	[401,600]	[601,800]	[801,1000]	[1001,1200]	≥ 1201
Science & Engineering	0.01	0.78	0.11	0.04	0.03	0.02	0.00	0.01
Audio & Video	0.00	0.89	0.01	0.04	0.01	0.02	0.02	0.01
Communications	0.01	0.82	0.11	0.04	0.01	0.00	0.00	0.01
Business & Enterprise	0.01	0.73	0.14	0.05	0.04	0.00	0.01	0.02
Graphics	0.00	0.90	0.06	0.03	0.01	0.00	0.00	0.00
Games	0.01	0.82	0.13	0.03	0.01	0.00	0.00	0.00
Development	0.01	0.76	0.12	0.05	0.02	0.02	0.01	0.01
System Administration	0.01	0.93	0.04	0.02	0.00	0.00	0.00	0.00
Security & Utilities	0.00	0.94	0.04	0.00	0.01	0.00	0.00	0.01
Home & Education	0.01	0.88	0.06	0.02	0.02	0.01	0.00	0.00

TABLE II. NUMBER OF PROJECTS BY CATEGORY (SIZE [2,200])

Category	Number of Projects
Science & Engineering	179
Audio & Video	80
Communications	164
Business & Enterprise	87
Graphics	101
Games	222
Development	1254
System Administration	221
Security & Utilities	79
Home & Education	75
TOTAL	2462

We can observe that categories *Games* and *Development* had *p-value* < 0.01 for all coupling metrics. This result is coherent with boxplots presented in the data exploration because these categories also have a different characteristic concerning coupling metrics.

For Test II, Table IV shows, for each metric, the mean value and standard deviation considering all classes from projects of the same category. For each metric, the highest values were marked with `>´, while the lowest values were marked with `<´. The category *Games* presented the highest mean value in all five metrics and the category *Development* presented the lowest mean value for four of five metrics.

From the results of the data exploration and the two tests, we can see that the major category that a software project belongs can impact on module coupling metrics. Each one of the conducted analyses contributed with a specific view to get this answer: the data exploration enabled to compare visually the mean coupling level of the projects and the differences could be observed; Test I enabled to compare the coupling level of a category with the coupling level of the universe of categories. In this case, we could observe that the categories *Games* and *Development* presented the highest number of differences, indicating that these categories must be treated carefully when considering thresholds for quality analysis. Test II showed that the means of metric values vary between different categories, as well the standard deviation. This test was also consistent with the previous analyzes because the categories *Games* and *Development* had higher and lower mean values, respectively.

Considering that we are conducting an experimental study, we should consider the threats to validity. The set of chosen projects represents a threat to external validity. Although we have produced a reasonably fair set of projects to conduct the study, which also has a much larger scale compared to related work [10], the sample could not be the best representative of those categories.

Another limitation is that we used only Java projects and the results may not be applied to other languages. Nonetheless, the coupling metrics used in this project applies to any other OO language, so the study could also be a strong indicative that the results would be valid for languages such as C++ and C#, and the methodology could also be applied to repository of programs in these languages. Another threat is that our results rely on metric values extracted with the iPlasma tool, and we cannot assure that the results are completely accurate for all data points, although we have verified that the results were correct for a small sample used in a testing phase.

TABLE III. RESULTS OF WILCOXON RANK SUM TEST

Category	ATFD	CBO	DAC	Ca	CBO*
Science & Engineering					
Audio & Video					
Communications				X	
Business & Enterprise			X		
Graphics					
Games	X	X	X	X	X
Development	X	X	X	X	X
System Administration		X			
Security & Utilities	X				
Home & Education					

TABLE IV. MEAN AND STANDARD DEVIATION FOR CLASSES METRICS

Category	ATFD	CBO	DAC	Ca	CBO*
Science & Engineering	2.08 6.26	2.04 3.20	0.63 1.27	3 6.14	5.24 10.49
Audio &Video	1.77 4.80	1.96 3.18	0.6 1.41	2.76 5.47	4.88 10.06
Communications	1.77 5.23	1.91 3.26	0.55 1.14	2.8 5.21	4.76 10.31
Business & Enterprise	2.34 7.70	2.01 3.17	0.57 1.35	2.55 5.19	5.3 11.13
Graphics	1.97 5.33	2.08 3.39	0.59 1.22	2.78 5.48	5.42 11.21
Games	2.60> 6.65	2.36> 3.76	0.82> 1.61	3.17> 5.32	6.91> 15.98
Development	1.62< 4.69	1.75< 2.85	0.43< 1.00	2.51 5.58	4.09< 9.00
System Administration	1.94 6.20	1.88 2.91	0.64 1.72	2.36< 4.36	4.78 10.89
Security & Utilities	2.02 5.74	2.06 3.29	0.56 1.41	2.72 5.36	5.26 12.09
Home & Education	2.14 5.08	2.09 3.09	0.61 1.36	2.57 5.01	5.35 10.87

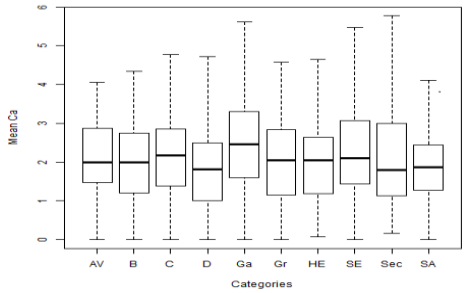

Fig. 1. Boxplots for mean Ca per project.

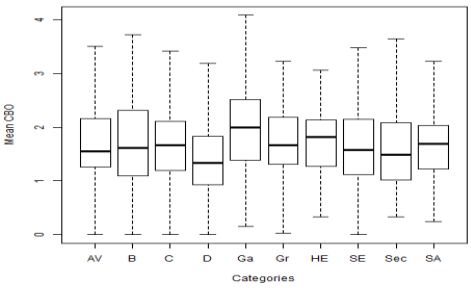

Fig. 2. Boxplots for mean CBO per project.

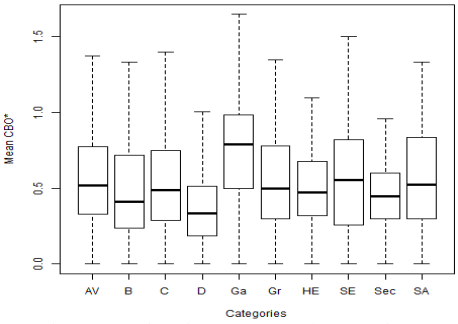

Fig. 3. Boxplots for mean CBO* per project.

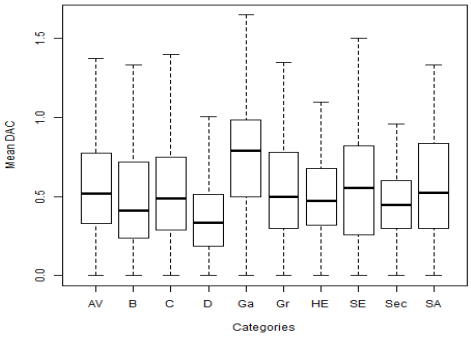

Fig. 4. Boxplots for mean DAC per project.

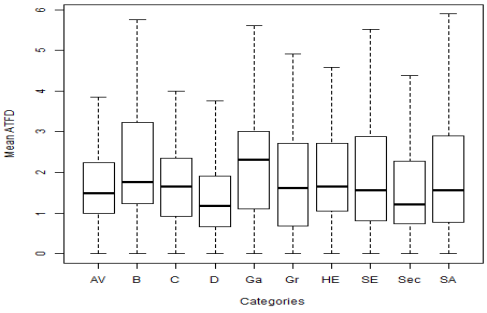

Fig. 5. Boxplots for mean ATFD per project.

IV. Conclusion

The goal of this study was to investigate if there were differences in the coupling level of projects in different categories of software. We selected Java projects from the Sourceforge belonging to 10 distinct categories. We assessed the coupling level using five different metrics. The results strongly suggest that there is different coupling level among different categories. Some categories may have higher coupling levels and other have lower levels. From the 10 categories analyzed, the category *Games* had a higher coupling level, while the category *Development* seems to be less coupled than the others. We conducted a stringent statistical test at significant level of 0.01 to provide higher confidence on this conclusion. This result strongly suggests that we need special attention when considering coupling thresholds when evaluating systems in the mentioned categories.

As future work, systems in different programming languages can be investigated, as well as other kind of metrics. Also, a qualitative analysis that explains the obtained results would enhance our comprehension on the characteristics of the categories that produce that difference.

Acknowledgements

This work was partially supported by FAPEMIG grant CEX-APQ-0286-11 and CNPQ grant 475519/2012-4.

References

[1] R. Martin, "OO Design Quality Metrics - An Analysis of Dependencies," in *Workshop Pragmatic and Theoretical Directions in Object-Oriented Software Metrics*, 1994.

[2] G. Gui and P. D. Scott, "Ranking reusability of software components using coupling metrics," *J. Syst. Softw.*, vol. 80, pp. 1450–1459, 2007.

[3] G. Kakarontzas, E. Constantinou, A. Ampatzoglou, and I. Stamelos, "Layer assessment of object-oriented software: A metric facilitating white-box reuse," *J. Syst. Softw.*, vol. 86, pp. 349–366, Feb. 2013.

[4] C. McMillan, N. Hariri, D. Poshyvanyk, J. Cleland-Huang, and B. Mobasher, "Recommending source code for use in rapid software prototypes," in *Proc. of the ICSE*, 2012, pp. 848–858.

[5] R. Marinescu, "Detecting Design Flaws via Metrics in Object-Oriented Systems" in *Proc. of the TOOLS39*, 2001, p. 173.

[6] F. Khomh, et al. "A Bayesian Approach for the Detection of Code and Design Smells," in *Proc of Intl. Conf Quality Softw.*, 2009, pp. 305–314.

[7] I. Macia, et al., "Are automatically-detected code anomalies relevant to architectural modularity?" in *Proc. of the AOSD*, 2012, pp. 167–178.

[8] S. M. Olbrich, D. S. Cruzes, and D. Sjoberg, "Are all code smells harmful? A study of God Classes and Brain Classes in the evolution of three open source systems," in *Proc. of the ICSM*, 2010, pp. 1–10.

[9] S. Olbrich, D. Cruzes, V. Basili, and N. Zazworka, "The evolution and impact of code smells: A case study of two open source systems," in *Proc. of the 3rd ESEM*, 2009, pp. 390–400.

[10] K. A. M. Ferreira, et al., "Identifying thresholds for object-oriented software metrics," *J. Syst. Softw.*, v. 85, pp. 244–257, 2012.

[11] S. R. Chidamber and C. F. Kemerer, "A Metrics Suite for Object Oriented Design," *IEEE Trans. Softw. Eng.*, v. 20, pp. 476–493, 1994.

[12] W. Li and S. Henry, "Object-oriented metrics that predict maintainability," *J. Syst. Softw.*, vol. 23, pp. 111–122, Nov. 1993.

[13] R. Marinescu, "Detection Strategies: Metrics-Based Rules for Detecting Design Flaws," in *Proc. the 20th ICSM*, 2004, pp. 350–359.

[14] M. Lanza, R. Marinescu, and S. Ducasse, *Object-Oriented Metrics in Practice*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[15] E. Linstead, et al., "Sourcerer: mining and searching internet-scale software repositories," *Data Min. Knowl. Discov.*, vol. 18, pp. 300–336, Apr. 2009.