DOI: https://doi.org/10.15276/hait.05.2022.13
UDC 004.412:519.237.5

A statistical estimation of the coupling between object metric for open-source apps developed in Java

Sergiy B. Prykhodko[1)]
ORCID: https://orcid.org/0000-0002-2325-018X; sergiy.prykhodko@nuos.edu.ua. Scopus Author ID: 55225622100
Kateryna S. Prykhodko[1)]
ORCID: https://orcid.org/0000-0003-0310-5724; kateryna.s.prykhodko@gmail.com. Scopus Author ID: 57200139991
Tetiana G. Smykodub[1)]
ORCID: http://orcid.org/0000-0002-4254-5477?lang=en; tgsmyk@gmail.com. Scopus Author ID: 57200139871
[1)] Admiral Makarov National University of Shipbuilding, 9, Heroes of Ukraine Ave. Mykolaiv, 54007, Ukraine

ABSTRACT

The coupling between objects along with other metrics, is used for evaluating the faults, vulnerabilities, and other quality indicators in software systems, including open-source ones. It is known, that a coupling between objects value between one and four is good. However, there are apps in Java for which the coupling between objects metric value at an app level is greater than four. That is why, in our opinion, the above interval for coupling between objects needs to be clarified for the app level. To find the recommended values for the coupling between objects mean of an app we have proposed to apply the confidence and prediction intervals. A coupling between objects mean value of an app from the confidence interval is good since this interval indicates how reliable the estimate is for all apps. A coupling between objects mean value higher than an upper bound of the prediction interval may indicate that some classes are too tightly coupled with other ones in the app. We have estimated the confidence and prediction intervals of the coupling between objects mean using normalizing transformations for the data sample from one hundred open-source apps developed in Java hosted on GitHub. Comparison with the coupling between objects mean values of three popular open-source apps developed in Java illustrate the applicability of the proposed quality indicators in the form of the confidence and prediction intervals of the coupling between objects mean.

Keywords: Statistical estimation; software metric; coupling between objects; open-source application; Java

***For citation*: Prykhodko S. B., Prykhodko K. S., Smykodub T. G. "A statistical estimation of the coupling between objects metric for open-source apps developed in Java". *Herald of Advanced Information Technology.* 2022; Vol. 5 No. 3: 175–184. DOI: https://doi.org/10.15276//hait.05.2022.13**

INTRODUCTION

The coupling between objects (CBO) metric was firstly defined by Chidamber and Kemerer in [1]. The CBO metric, along with others, is used for evaluating the faults [2, 3], vulnerabilities [4], maintenance [5, 6], complexity [7, 8], and other quality indicators [9] of software systems, including open-source apps [10, 11]. This metric indicates the required effort to test and maintain a class [9]. At a class level, the coupling between objects metric is the number of classes coupled to a given class. At an app level, this metric provides the average number of classes used per class. Coupling between objects is required for an app to do useful work, but excessive coupling makes the app more difficult to maintain and reuse. It is known [12], a CBO value between 1 and 4 is good since it indicates that the class is loosely coupled. A value higher than this may indicate that the class is too tightly coupled with other classes in the app, which would complicate testing and modification, and limit the possibilities of reuse [12]. Also, a high coupling hasbeen found to indicate fault-proneness [13].

According to [14], a coupling between objects value greater than 14 is too high. However, there are apps in Java, in which the coupling between objects values for some classes are significantly greater than 4 and even 14. For example, in the TuxGuitar app version 1.5.2, the software metrics of which are analyzed in [9], there are the org.herac.tuxguitar.app.action.impl.system. TGDisposeAction, org.herac.tuxguitar.app.action. installer TGActionConfigMap, and org.herac. tuxguitar.app.action.installer.TGActionInstaller classes where CBO values equal 31, 52, and 252, respectively. Note that TuxGuitar is a fairly large app with 207810 lines of code, which includes 2381 classes. The total CBO for all classes is 20189.

In this regard, the need arises to evaluate the object-oriented design (OOD) of the entire app, and not its classes, from the point of view of the

© Prykhodko S., Prykhodko K., Smykodub T., 2022

This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/deed.uk)

coupling between objects, if the CBO values of some classes are significantly greater than the maximum recommended CBO value of 4.

ANALYSIS OF LITERARY DATA

There are known applying various software metrics at an app level [15, 16], including CBO [9, 15], in some aspects of object-oriented design. However, the above papers present only point estimates of the metrics. Such according to [15], the CBO average values of 2.48 and 1.25 indicate the low and high quality of the software system in Java, respectively. At an app level, similar interval estimates for the CBO metric are not known. Only in [17] it was proposed to apply the confidence and prediction intervals to find interval estimates for the DIT metric at an app level. According to [17], a DIT average of an app from the confidence interval is good since this interval indicates how reliable the estimate is for the DIT average values of all apps used for estimating the interval. A DIT average higher than an upper bound of the prediction interval may indicate that some classes have a large number of inheritance levels from the object hierarchy top.

As we know, well-known statistical methods to estimate the confidence [18, 19] and prediction intervals [20, 21] of random variables and their estimates by data samples are used under the assumption that the data is generated by a Gaussian distribution. However, distributions of software metric data, including CBO, are not Gaussian. The empirical distributions of software metric data in the form of histograms, for example, in [1, 9] suggest that. That is why in [17] the confidence and prediction intervals of the DIT metric at an app level were estimated by two techniques based on the normalizing transformations.

Similarly to [17], we apply normalizing transformations to evaluate the confidence and prediction intervals of the CBO mean by appropriate techniques.

FORMULATION OF THE PROBLEM

Suppose given the original sample of coupling between objects random variable *X*, which distribution is not Gaussian. Suppose that there is a normalizing transformation of variable *X* to Gaussian random variable *Z* as $Z = \psi(X)$, which has the corresponding inverse transformation $X = \psi^{-1}(Z)$.

Need to estimate the confidence and prediction intervals of the CBO mean by the CBO sample using normalizing transformation $Z = \psi(X)$ and inverse transformation $X = \psi^{-1}(Z)$.

OBJECTIVES OF THE STUDY

The work aims to find the confidence and prediction intervals of the coupling between objects mean for open-source apps developed in Java for a 0.05 significance level.

We chose open-source apps developed in Java for two reasons. First, because of the ability to collect data for open-source apps. Second, Java is a popular programming language that is used practically everywhere from laptops to datacenters, game consoles to mainframes, and cell phones to the Internet.

MATERIALS AND RESEARCH METHODS

Similarly to [17], we apply normalizing transformations to evaluate the confidence and prediction intervals of the coupling between objects mean by appropriate techniques. The technique for estimating the prediction intervals is based on normalizing transformations and Grubb's test [22]. The technique is as follows [17]. At first, we normalize non-Gaussian data by the bijective normalizer transformation. Then we transform the sample mean of the non-Gaussian data using the normalizing transformation and calculate the deviation from the sample mean for normalized data (approximately with Gaussian distribution) in Grubb's test. After that, we define the bounds of the interval for normalized data by respectively subtracting and adding the calculated deviation from the sample mean. Finally, we detect the boundaries of the prediction interval for non-Gaussian data by transforming the bounds of the prediction interval for normalized data by the transformation inversed to normalizing one. The technique for estimating the confidence intervals of the sample mean is similar to the previous one, with the only difference that we calculate the deviation from the sample mean for normalized data by student's *t*-distribution [17].

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

ESTIMATING THE CONFIDENCE AND PREDICTION INTERVALS OF THE CBO MEAN FOR OPEN-SOURCE APPS DEVELOPED IN JAVA

We have estimated the confidence and prediction intervals of the CBO mean using normalizing transformations for the data sample from 100 open-source apps developed in Java hosted on GitHub (https://github.com/). The data sample was obtained using the CK tool [23]. The minimum and maximum values of the CBO mean equal 3.307 and 22.417, respectively. The average and sample standard deviation values of the CBO mean are 8.925 and 3.964, respectively. Table 1 contains the observed n_j and expected Np_j frequencies of type j of the CBO mean values for seven intervals from 100 open-source apps developed in Java. Here N is the data sample size and p_j is the probability of type j. We used the Gaussian and Johnson distributions for calculating the expected (theoretical) frequencies of the CBO mean.

Note, when the model is fully specified, computing the degrees of freedom number of the statistic is easy: it is equal to the number of cells (or observation intervals) minus one [18]. For testing the goodness of fit, the degrees of freedom number is reduced by the number of fitted parameters in the distribution, that for the Gaussian distribution equals two.

Table 1. **The observed and expected frequencies of CBO mean values**

j	Actual interval		Observed frequencies, n_j	Expected frequencies, Np_j	
	LB	UB		Gaussian	Johnson
1	3.308	6.038	27	15.49	26.10
2	6.038	8.767	29	25.10	30.69
3	8.767	11.50	23	25.77	20.24
4	11.497	14.23	8	16.77	11.77
5	14.227	16.96	7	6.91	6.29
6	16.957	19.69	5	1.81	3.01
7	19.687	22.42	1	0.30	1.20

Source: **compiled by the authors**

We reject the null hypothesis that the distribution of the CBO mean values is the same as the normal distribution and accept the alternative hypothesis H_1 (there is a difference between the distributions) since the chi-squared test statistic χ^2 value equals 21.34 is higher than the critical value of the chi-square, which equals to 9.49 for 4 degrees of freedom and 0.05 significance level. For the distribution of the coupling between objects mean values, estimators of skewness and kurtosis equal 1.07 and 3.74, respectively. These values also indicate to us that the data of CBO mean are not Gaussian, since the skewness and kurtosis of the Gaussian distribution are equal to 0 and 3, respectively. Also, Table 1 contains the expected frequencies of CBO mean values obtained by the Gaussian distribution, lower (LB), and upper (UB) bounds of actual intervals of the CBO mean values.

Therefore, we apply normalizing transformations to estimate the confidence and prediction intervals of the CBO mean. As in [17], for normalizing the data sample, we use the Johnson translation system [24] that set up a transformation of a continuous random variable X to a standard Gaussian variable Z and is defined by

$$Z = \gamma + \eta h(X, \varphi, \lambda) \sim N(0,1), \qquad (1)$$

where γ, η, φ, and λ are parameters of transformation (1), $-\infty < \gamma < \infty$, $\eta > 0$, $-\infty < \varphi < \infty$ $\lambda > 0$, the translation function h takes four possible forms

$$h = \begin{cases} \ln(y), & \text{for } S_L \text{ (log normal) family;} \\ \ln[y/(1-y)], & \text{for } S_B \text{ (bounded) family;} \\ \text{Arsh}(y), & \text{for } S_U \text{ (unbounded) family;} \\ y & \text{for } S_N \text{ (normal) family.} \end{cases} \qquad (2)$$

Here $y = (X - \varphi)/\lambda$, $\text{Arsh}(y) = \ln\left(y + \sqrt{y^2 + 1}\right)$.

We use the Johnson translation function h for S_B family (2) since the skewness and kurtosis estimators for the observed frequency distribution of the CBO mean values equal 1.07 and 3.74, respectively.

The Johnson transformation (1) for the S_B family from (2) is also defined by

$$Z = \gamma + \eta \ln \frac{X - \varphi}{\varphi + \lambda - X}, \qquad (3)$$

where $\varphi < X < \varphi + \lambda$, $\eta > 0$, $-\infty < \varphi < \infty$, $\lambda > 0$.

The Johnson probability density function (pdf) for the S_B family is obtained using transformation (3) and has the form [24]

$$f_B(X) = \frac{\eta\lambda}{\sqrt{2\pi}(X - \varphi)(\lambda + \varphi - X)} \times$$

$$\times \exp\left\{-\frac{1}{2}\left[\gamma+\eta\ln\left(\frac{X-\varphi}{\lambda+\varphi-X}\right)\right]^2\right\}, \quad (4)$$

where $\varphi < X < \varphi+\lambda$, $-\infty<\gamma<\infty$, $\eta>0$, $-\infty<\varphi<\infty$, $\lambda>0$.

The parameter vector $\boldsymbol{\theta}=\{\gamma,\eta,\varphi,\lambda\}$ of the Johnson transformation (3) and pdf (4) for the S_B family is estimated by the maximum likelihood method (MLM) [17]

$$\hat{\boldsymbol{\theta}}=\arg\max_{\boldsymbol{\theta}} l(X,\boldsymbol{\theta}), \quad (5)$$

where the log-likelihood function is

$$l(X,\boldsymbol{\theta})=N\ln(\eta\lambda)-\frac{N\ln(2\pi)}{2}-\sum_{i=1}^{N}\ln(x_i-\varphi)-$$

$$-\sum_{i=1}^{N}\ln(\varphi+\lambda-x_i)-\frac{1}{2}\sum_{i=1}^{N}\left[\gamma+\eta\ln\frac{x_i-\varphi}{\varphi+\lambda-x_i}\right]^2, \quad (6)$$

where x_i is the i-value of X from transformation (3).

Estimators for parameters of the Johnson transformation (3) and pdf (4) for S_B family by MLM (4) are: $\hat{\gamma}=1.560865$, $\hat{\eta}=1.10777$, $\hat{\varphi}=2.77482$, and $\hat{\lambda}=26{,}840$. The value of the log-likelihood function (6) equals -263.1.

Table 1 contains expected frequencies of CBO mean values obtained by the Johnson distribution for S_B family (4) with the above estimates of parameters. Table 2 contains the observed and expected frequencies of normalized CBO mean values using the Johnson transformation for S_B family (3) with the above estimates of parameters.

Table 2. **The observed and expected frequencies of normalized coupling between objects mean values by the Johnson transformation for the S_B family**

j	LB	UB	n_j	Np_j
1	-2.759	-1.983	2	2.08
2	-1.983	-1.207	8	9.01
3	-1.207	-0.431	25	21.95
4	-0.431	0.345	29	30.17
5	0.345	1.121	23	23.39
6	1.121	1.897	10	10.22
7	1.897	2.673	3	2.52

Source: **compiled by the authors**

We used the Pearson chi-squared test to check the normality of normalized data based on the Johnson transformation for the S_B family. The Chi-square test allows us to confirm the null hypothesis H_0 that is compatible with the assumption of normality in normalized data based on the Johnson transformation for the S_B family, since the χ^2 value, which is equal to 0.688, is less than the critical value of the chi-square for 0.05 significance level and 4 degrees of freedom. The skewness and kurtosis estimators for the distribution of normalized data based on the Johnson transformation for the S_B family equal 0.036 and 3.035, respectively. These values also indicate the normality of the normalized data.

Also to normalize the data sample, we use the Box-Cox transformation [25]

$$Z=x(\lambda)=\begin{cases}\dfrac{X^{\lambda}-1}{\lambda}, & \text{if} \quad \lambda\neq 0;\\ \ln(X), & \text{if} \quad \lambda=0.\end{cases} \quad (7)$$

Here λ is a parameter of the Box-Cox transformation (7).

The parameter λ of the Box-Cox transformation (7) was estimated by the MLM [26]

$$\hat{\lambda}=\arg\max_{\lambda} l(X,\lambda), \quad (8)$$

where the log-likelihood function is

$$l(X,\lambda)=C-\frac{N}{2}\ln\sum_{i=1}^{N}\frac{[x_i(\lambda)-\bar{x}(\lambda)]^2}{N}+$$

$$+(\lambda-1)\sum_{i=1}^{N}\ln(x_i), \quad (9)$$

Here C is a constant, which is determined from the normalization condition; $\bar{x}(\lambda)=\sum_{i=1}^{N}x_i(\lambda)/N$; $x_i(\lambda)$ is the i-value of $x(\lambda)$ or Z from (6).

The parameter estimator $\hat{\lambda}$ is -0.21357 by the MLM (8). The value of the log-likelihood function (9) equals -123.1.

Table 3 contains the observed and expected frequencies of normalized CBO mean values using the Box-Cox transformation with the above estimate $\hat{\lambda}$.

A comparison of the observed and expected frequencies of normalized CBO mean values from Table 2 and Table 3 indicates the best data normalization by the Johnson transformation for the S_B family besides the Box-Cox transformation.

We have evaluated the confidence and prediction intervals of the CBO mean for a significance level of 0.05 according to [17] using the Johnson transformation for the S_B family and the Box-Cox transformation.

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

Table 3. **The observed and expected frequencies of normalized CBO mean values by the Box-Cox transformation**

j	LB	UB	n_j	Np_j
1	1.056	1.229	2	3.68
2	1.229	1.403	16	10.51
3	1.403	1.577	19	20.00
4	1.577	1.751	22	25.36
5	1.751	1.925	24	21.45
6	1.925	2.098	10	12.10
7	2.098	2.272	7	4.55

Source: **compiled by the authors**

The bounds of the confidence interval for non-Gaussian data are defined by transforming the bounds of the confidence interval for normalized data

$$\left[\bar{z}_{\bar{x}} - \Delta_z, \bar{z}_{\bar{x}} + \Delta_z\right], \tag{10}$$

using the inverse transformation according to [16]

$$\left[\psi^{-1}\left(\bar{z}_{\bar{x}} - \Delta_z\right), \psi^{-1}\left(\bar{z}_{\bar{x}} + \Delta_z\right)\right], \tag{11}$$

where ψ is a function of normalizing transformation $z = \psi(x)$, $\bar{z}_{\bar{x}} = \psi(\bar{x})$, $\bar{x}$ is the sample mean of the non-Gaussian data, Δ_z is the deviation from the sample mean for normalized data by student's *t*-distribution

$$\Delta_z = t_{\alpha/2,N-1} S_z / \sqrt{N}. \tag{12}$$

In (12) $t_{\alpha/2,N-1}$ is a quantile of students' *t*-distribution with $N-1$ degrees of freedom and $\alpha/2$ significance level, N is the data size, S_z is the sample standard deviation of the normalized data.

The bounds of the prediction interval for non-Gaussian data are calculated analogously (11) with the only difference that instead of formula (11) for Δ_z should be used

$$\Delta_z = S_z \frac{N-1}{\sqrt{N}} \sqrt{\frac{t^2_{\alpha/(2N),N-2}}{N-2+t^2_{\alpha/(2N),N-2}}}, \tag{13}$$

where $t_{\alpha/(2N),N-2}$ is a quantile of students' *t*-distribution with $N-2$ degrees of freedom and $\alpha/(2N)$ significance level.

The Johnson transformation for S_B family (3) has the following inverse transformation

$$X = \varphi + \lambda h^{-1}(Z, \gamma, \eta), \tag{14}$$

where the inverse translation function $h^{-1}(z, \gamma, \eta)$ takes the form

$$h^{-1} = 1/\left(1 + e^{-\zeta}\right).$$

Here $\zeta = (Z - \gamma)/\eta$, $\eta > 0$, $-\infty < \gamma < \infty$, $\lambda > 0$, $-\infty < \varphi < \infty$.

The Box-Cox transformation (7) has the following inverse transformation

$$X = (\lambda Z + 1)^{1/\lambda}. \tag{15}$$

Here λ is a parameter of the Box-Cox transformation (7).

The confidence intervals of the CBO mean in the case of applying the Johnson transformation for S_B family (3) with its inverse transformation (14) and the Box-Cox transformation (7) with its inverse transformation (15) are from 8.12 to 9.81 and from 8.20 to 9.72, respectively. The prediction intervals of the CBO mean in the case of using the Johnson transformation for S_B family (3) with its inverse transformation (14) and the Box-Cox transformation (7) with its inverse transformation (15) are from 3.15 to 25.94 and from 2.52 to 50.54, respectively.

Also, we have estimated the confidence and prediction intervals of the CBO mean for a significance level of 0.05 by (10) without the normalization. The only difference is that we substitute the sample mean $\bar{x}$ instead of $\bar{z}_{\bar{x}}$ in (10), and for calculating Δ_z by (12) and (13) we use the sample standard deviation value of the CBO mean instead S_z. In this case, when we consider our data to be Gaussian, the confidence and prediction intervals of the CBO mean for a significance level of 0.05 are from 8.14 to 9.71 and from -4.49 to 22.34, respectively. According to Grubb's test [27], the upper bound equaled to 22.34 indicates there is one outlier in the data sample, for which the CBO mean value is 22.417. In the case of applying the normalizing transformations [16, 28], both the Johnson transformation for the S_B family and the Box-Cox transformation, there are no outliers in the data sample.

Easy to notice, the values of confidence interval bounds are similar. However, the values of prediction interval bounds differ significantly. In this case, when we consider our data to be Gaussian, the lower bound of the prediction interval is negative, which does not correspond to real values. The lower bounds of the prediction interval calculated by the normalizing transformations are positive. The width of the prediction interval based on the Johnson transformation for the S_B family is less than after the

Box-Cox transformation and smaller than without one. This result may be explained best normalization of the non-Gaussian data set by the Johnson transformation for the S_B family as evidenced by the χ^2 value, which is about 9 times smaller than in the case of the Box-Cox transformation.

Therefore, we have selected the confidence and prediction intervals of the CBO mean for a significance level of 0.05 based on the Johnson transformation for the S_B family. A CBO mean value of an open-source app developed in Java between confidence interval bounds from 8.12 to 9.81 is good with a 0.05 significance level. A value from this interval indicates that the classes are loosely coupled on average according to software development practices of open-source apps developed in Java. A CBO mean value of an open-source app developed in Java between interval bounds from 3.15 to 8.12 is also good. That is why we might consider that a CBO mean value of an open-source app between interval bounds from 3.15 to 9.81 indicates its high quality.

A CBO mean value of an open-source app developed in Java from 9.81 to 25.94 is acceptable. But in this case, we might consider that a CBO mean value of an open-source app from the above interval (from 9.81 to 25.94) indicates its medium quality.

And finally, a CBO mean value higher than 25.94 may indicate that some classes are too tightly coupled with other ones in an open-source app developed in Java. In this case, we might consider that a CBO mean value of an open-source app, which is greater than 25.94, indicates its low quality.

The above lower and upper bounds are correlated with the result given in [15] that the higher the CBO, the lower the quality of the software system.

We have compared our results with the CBO mean values of three popular open-source apps developed in Java: FreeMind, TuxGuitar, and jEdit having over 465k, 131k, and 56k downloads in 2019, respectively [9]. The CBO mean values of FreeMind, TuxGuitar, and jEdit equal 5.36, 7.32, and 4.67, respectively [9, 29]. According to [30], the CBO mean value of TuxGuitar is 6.71. These values are good since they are in the calculated range from 3.15 to 9.81. The above CBO mean values of three open-source apps indicate their high quality. Also, the same results we have for various versions of the above apps, for which CBO mean values are given in [9].

Note, we detected the confidence and prediction intervals of the CBO mean of open-source apps developed in Java based on the 100 values of the CBO mean from the range of 3.308 to 22.42 (see Table 1). That is why we cannot use the CBO mean values of apps whose averages are out of the above range to detect software quality. For example, the like apps include three software systems considered in [15], simple board-based software games, such as the game called “X and O” [31].

Also, we used the CBO mean value of the Metro_systems app, which is 9.93 to detect software quality. The Metro_systems app is a Core Java project on managing metro systems and the introduction of smartcards for daily users (https://github.com/Sparsh6496/Metro_systems). The above CBO mean value indicates the medium quality of this project.

The CBO mean values of FreeMind, TuxGuitar, jEdit, and Metro_systems indicate confidence in the proposed bounds of the CBO mean intervals as the quality indicators of open-source apps developed in Java from point of view of an OOD.

In the future, to validate conclusions derived from data analysis based on the confidence and prediction intervals of the CBO mean, further study needs to be carried out for other software metrics (for example, RFC) and data sets. Also, in the future, it is planned to find confidence and prediction regions of admissible values of other metrics using the transformed ellipsoids [32] for multivariate non-Gaussian data.

DISCUSSION AND FUTURE RESEARCH

The paper is founded on the assumption that the CBO mean could be viewed as an app-level metric. This assumption is based on the papers of other authors, in particular [9, 15]. The study by NASA [15] analyzed three software systems and classified their quality depending on the average values for Chidamber & Kemerer metrics, including CBO one. The average values of the CBO metric from [15] suggest that the higher the CBO, the lower the software system quality. In [9] mean values of 16 software metrics (including the CBO) were considered to check their consistency across three popular open-source apps developed in Java and their versions.

We emphasize that the papers [9, 15] considered point estimates of mean values. The recommended interval estimates for the CBO mean of an app are not known. To find the interval estimates for the CBO mean of an app we have proposed to use the confidence and prediction intervals. A CBO mean value of an app from the confidence interval is good since this interval indicates how reliable the estimate is for all apps

ISSN 2663-0176 (Print)
ISSN 2663-7731 (Online)

used for estimating the interval with a certain significance level. We used a 0.05 significance level as the appointed one usually, although this value may be discussed. A CBO mean value between the lower bounds of the confidence and prediction intervals, or between the upper bounds of the confidence and prediction intervals is also acceptable.

We apply normalizing transformations to estimate the confidence and prediction intervals of the CBO mean by appropriate techniques [17] since the distribution of the CBO mean values is not Gaussian what the chi-squared test result and the values of estimators of skewness and kurtosis indicate. As stated earlier, estimators of skewness and kurtosis are equal to 1.07 and 3.74, respectively for the distribution of the CBO mean values. Moreover, the distributions of the CBO values at the class level for all reviewed apps are not Gaussian what the chi-squared test results and the values of estimators of skewness and kurtosis indicate too.

Concerning the proposed values of the bounds of the confidence and prediction intervals of the CBO mean as the quality indicators of software apps from the point of view of OOD two limitations should be acknowledged and addressed concerning the data sample from 100 open-source apps developed in Java. The first limitation concerns the estimation of the data sample for open-source apps developed in Java only. The evaluation of other data samples, for example, for industrial systems in Java [15], may affect the bounds of the confidence and prediction intervals of the CBO mean. In such cases, the proposed values of the bounds of the confidence and prediction intervals of the CBO mean remain to be confirmed or changed.

The second limitation concerns the sample size which equals 100. This value can be unambiguously considered as the lower size limit of the large sample. Larger sample sizes may lead to a reduction in the widths of the confidence and prediction intervals.

It should also be noted that the quality of software apps from the point of view of OOD may depend on other metrics at the app level, for example, RFC (response for a class) and WMC (weighted methods per class) [1, 15], the influence of which needs to be investigated.

CONCLUSIONS

1. We have discovered that for evaluating the confidence and prediction intervals of the CBO mean is need to use normalizing transformations. In this case, we have used two transformations, both the Johnson transformation for the S_B family and Box-Cox one. The best normalization of the data sample has been implemented by the Johnson transformation for the S_B family than the Box-Cox one. We have proposed to use the Johnson probability density function for the S_B family as a probabilistic model of the CBO mean of open-source apps developed in Java.

2. The confidence and prediction intervals of the CBO mean using normalizing transformations for the data sample from 100 open-source apps developed in Java hosted on GitHub are estimated.

The width of the prediction interval based on the Johnson transformation for the S_B family is less than after the Box-Cox transformation and smaller than without one. The values of confidence interval bounds are similar in all cases.

3. We have found the following intervals of the CBO mean for open-source apps developed in Java for a 0.05 significance level. A CBO mean value of an open-source app developed in Java from 3.15 to 9.81 is good. A CBO mean value of an open-source app developed in Java from 9.81 to 25.94 is also acceptable. A CBO mean value higher than 25.94 may indicate that some classes are too tightly coupled with other ones in an open-source app developed in Java.

4. We have proposed to apply the confidence and prediction intervals of the CBO mean as the quality indicators of software apps from point of view of an OOD. A CBO mean value of an open-source app between interval bounds from 3.15 to 9.81 indicates its high quality. A CBO mean value of an open-source app from the range of 9.81 to 25.94 indicates its medium quality. A CBO mean value of an open-source app, which is greater than 25.94, indicates its low quality.

Comparison with the CBO mean values of three popular open-source apps developed in Java illustrate the applicability of the proposed quality indicators in the form of the confidence and prediction intervals of the CBO mean.

5. In the future, to validate strong conclusions derived from data analysis based on the CBO mean intervals, further research needs to be carried out for other software metrics and data sets. Also, in the future, it is planned to find confidence and prediction regions of admissible values of other metrics using the transformed ellipsoids for multivariate non-Gaussian data. The influence of the CBO metric on the possible maintenance of software also needs further research.

REFERENCES

1. Chidamber, S. R. & Kemerer, C. F. “A metrics suite for object oriented design”. *IEEE Transactions on Software Engineering*. 1994; Vol. 20 Issue 6: 476–493. DOI: https://doi.org/10.1109/32.295895.

2. Shatnawi, R. “Empirical study of fault prediction for open-source systems using the Chidamber and Kemerer metrics”. *IET Software*. 2014; 8 (3): 113–119. https://www.scopus.com/record/display.uri?eid=2-s2.0-84901913588&origin=resultslist&sort=plf-f. DOI: http://dx.doi.org/10.1049/iet-sen.2013.0008.

3. Rathore, S. S. & Kumar, S. “A study on software fault prediction techniques”. *Artificial Intelligence Review*. 2019; 51 (2): 255–327. https://www.scopus.com/record/display.uri?eid=2-s2.0-85019755092&origin=resultslist&sort=plf-f . DOI: https://doi.org/10.1007/s10462-017-9563-5.

4. Chowdhury, I. & Zulkernine, M. “Using complexity, coupling, and cohesion metrics as early indicators of vulnerabilities”. *Journal of Systems Architecture*. 2011; 57 (3): 294–313. https://www.scopus.com/record/display.uri?eid=2-s2.0-79952574726&origin=resultslist&sort=plf-f. DOI: https://doi.org/10.1016/j.sysarc.2010.06.003.

5. Ajienka, N., Capiluppi, A. & Counsell, S. [Journal First] “An empirical study on the interplay between semantic coupling and Co-change of software classes”. *IEEE/ACM 40th International Conference on Software Engineering (ICSE). Conference Proceedings.* 2018. p. 432. DOI: https://doi.org/10.1145/3180155.3190833.

6. Ajienka, N., Capiluppi, A. & Counsell, S. “An empirical study on the interplay between semantic coupling and Co-change of software classes”. *Empirical Software Engineering*, 2018; 23 (3): 1791–1825. https://www.scopus.com/record/display.uri?eid=2-s2.0-85034604579&origin=resultslist &sort=plf-f. DOI: https://doi.org/10.1007/s10664-017-9569-2.

7. “Software quality metrics for object oriented system environments”. – Available from: https://qualazur.pagesperso-orange.fr/$swmesu2.htm. – [Accessed: Sep. 2021].

8. Rathore, N. P. S. & Gupta, R. “A novel coupling metrics measure difference between inheritance and interface to find better OOP paradigm using C#”. *World Congress on Information and Communication Technologies, Conference Proceedings.* 2011. p. 467–472. https://www.scopus.com/record/display.uri?eid=2-s2.0-84857183826&origin=resultslist&sort=plf-f. DOI: https://doi.org/10.1109/WICT.2011.6141290.

9. Molnar, A. J., Neamţu, A. & Motogna, S. “Evaluation of software product quality metrics”. In: Damiani E., Spanoudakis G., Maciaszek L. (eds) “Evaluation of novel approaches to software Engineering”. *ENASE 2019*. *Communications in Computer and Information Science*. 2020; 1172: 163–187. *Springer. Cham.* https://www.scopus.com/record/display.uri?eid=2-s2.0-85080917482&origin=resultslist&sort=plf-f. DOI: https://doi.org/10.1007/978-3-030-40223-5_8.

10. Foucault, M., Teyton, C., Lo, D., Blanc, X. & Falleri, J.-R. “On the usefulness of ownership metrics in open-source software projects”. *Information and Software Technology*. 2015; 64: 102–112. https://www.scopus.com/record/display.uri?eid=2-s2.0-84929024229&origin=resultslist&sort=plf-f. DOI: https://doi.org/10.1016/j.infsof.2015.01.013.

11. Bouktif, S., Sahraoui, H. & Ahmed, F. “Predicting stability of open-source software systems using combination of Bayesian classifiers”. *ACM Transactions on Management Information Systems*. 2014; 5 (1): 1–26. DOI: https://doi.org/10.1145/2555596.

12. “Coupling between object classes (CBO)”. – Available from: https://www.cachequality.com/docs/metrics/coupling-between-object-classes-cbo. – [Accessed: Sep. 2021].

13. “Chidamber & Kemerer object-oriented metrics suite”. – Available from: https://www.aivosto.com/project/help/pm-oo-ck.html. – [Accessed: Sep, 2021].

14. Sahraoui, H. A., Godin, R. & Miceli, T. “Can metrics help to bridge the gap between the improvement of OO design quality and its automation?” *International Conference on Software Maintenance, Conference Proceedings.* 2000. p. 154–162. https://www.scopus.com/record/display.uri?eid=2-s2.0-0034497430&origin=resultslist&sort=plf-f. DOI: https://doi.org/10.1109/ICSM.2000.883034.

15. Laing, V. & Coleman, C. “Principal components of orthogonal object-oriented metrics”. *White Paper Analyzing Results of NASA Object-Oriented Data*. SATC, NASA. 2001. 18 p.

16. Mishra, D. “New inheritance complexity metrics for object-oriented software systems: An evaluation with weyuker's properties”. *Computing and Informatics*. 2011; 30 (2): 267–293.

17. Prykhodko, S., Prykhodko, N. & Smykodub, T. “A statistical evaluation of the depth of inheritance tree metric for open-source applications developed in Java”. *Foundations of Computing and Decision*

Sciences. 2021; 46 (2): 159–172. https://www.scopus.com/record/display.uri?eid=2-s2.0-85108561565&origin=resultslist&featureToggles=FEATURE_NEW_METRICS_SECTION:1, https://www.webofscience.com/wos/woscc/full-record/WOS:000663561000003?SID=EUW1ED0CAAnUqtZTfXPOEuQmHPedM&state=%7B%7D. DOI: https://doi.org/10.2478/fcds-2021-0011.

18. Moore, D. S., McCabe, G. P. & Craig, B. A. "Introduction to the practice of statistics". Ninth Edition. *W. H. Freeman*. 2016. 725 p.

19. Kreyszig, E. "Advanced engineering mathematics". Tenth Edition. *John Wiley & Sons, Inc.* 2011. 1280 p.

20. Evans, M. J. & Rosenthal, J. S. "Probability and statistics: The science of uncertainty". [redacted] *W. H. Freeman*, 2009. 638 p.

21. Montgomery, D. C., Runger, G. C. & Hubele, N. F. "Engineering statistics". Fifth Edition. *John Wiley & Sons, Inc.* 2011. 544 p.

22. Prykhodko, S. B. "Statistical anomaly detection techniques based on normalizing transformations for non-Gaussian data". *The International Conference on Computational Intelligence (Results, Problems and Perspectives). Conference Proceedings.* Kyiv-Cherkasy: Ukraine. 2015. p. 286–287.

23. "Mauricioaniche/ck". – Available from: https://github.com/mauricioaniche/ck/tree/ck-0.5.2. – [Accessed: Feb. 2021].

24. Kendall, M. G. & Stuart, A. "The advanced theory of statistics". Distribution Theory. 2nd edn. *Charles Griffin & Company Limited, London*. 1963; Vol. 1: 433 p.

25. Box, G. E. P. & Cox, D. R. "An analysis of transformations". *Journal of the Royal Statistical Society, Series B (Methodological)*. 1964; 26 (2): 211–252.

26. Johnson, R. A. & Wichern, R. A. "Applied multivariate statistical analysis". *Pearson Prentice Hall*. 2007. 773 p.

27. Grubbs, F. "Procedures for detecting outlying observations in samples". *Technometric*s. 1969; 11 (1): 1–21.

28. Prykhodko, S., Prykhodko, N., Makarova, L. & Pugachenko, K. "Detecting outliers in multivariate non-Gaussian data on the basis of normalizing transformations". *The 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON), Conference Proceedings.* Kyiv: Ukraine. 2017. p. 846–849. https://www.webofscience.com/wos/woscc/full-record/WOS:000426985500171.
DOI: https://doi.org/10.1109/ UKRCON.2017.8100366.

29. Molnar, A., Neamtu, A. & Motogna, S. "Longitudinal evaluation of software quality metrics in open-source applications". *The 14th International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE. Conference Proceedings. INSTICC. SciTePress.* 2019; 1: 80–91. DOI: https://doi.org/10.5220/0007725600800091.

30. Barkmann, H., Lincke, R. & Lowe, W. "Quantitative evaluation of software quality metrics in open-source projects". *2009 International Conference on Advanced Information Networking and Applications Workshops, Conference Proceedings.* 2009. p. 1067–1072. https://doi.org/10.1109/WAINA.2009.190.

31. Sabahat, N., Afzal Malik, A. & Azam, F. "Utility of CK Metrics in Predicting Size of Board-Based Software Games". *Mehran University Research Journal of Engineering and Technology*. 2017; 36 (4): 975–986.

32. Prykhodko, S., Makarova, L., Prykhodko, K. & Pukhalevych, A. "Application of transformed prediction ellipsoids for outlier detection in multivariate non-gaussian data". *IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET). Conference Proceedings.* 2020. p. 359–362. https://www.scopus.com/record/display.uri?eid=2-s2.0-85086308714&origin=resultslist&sort=plf-f&src=s&sid=8088a14120bf4e71589512bb7ce82a8b&sot=autdocs&sdt=autdocs&sl=18&s=AU-ID%2855225622100%29&relpos=0&citeCnt=0&searchTerm=, https://www.webofscience.com/wos/woscc/full-record/WOS:000578041000075.
DOI: https://doi.org/10.1109/TCSET49122.2020.235454.

Conflicts of Interest: the authors declare no conflict of interest

Received 07.09.2022
Received after revision 12.10.2022
Accepted 19.10.2022

DOI: https://doi.org/10.15276/hait.05.2022.13
УДК 004.412:519.237.5

Статистичне оцінювання метрики зв’язування між об’єктами для застосунків із відкритим кодом розроблених на Java

Приходько Сергій Борисович[1)]
ORCID: https://orcid.org/0000-0002-2325-018X; sergiy.prykhodko@nuos.edu.ua. Scopus Author ID: 55225622100
Приходько Катерина Сергіївна[1)]
ORCID: https://orcid.org/0000-0003-0310-5724; kateryna.s.prykhodko@gmail.com. Scopus Author ID: 57200139991
Смикодуб Тетяна Георгіївна[1)]
ORCID: http://orcid.org/0000-0002-4254-5477?lang=en; tgsmyk@gmail.com. Scopus Author ID: 57200139871
[1)] Національний університет кораблебудування імені адмірала Макарова, пр. Героїв України, 9. Миколаїв, 54007, Україна

АНОТАЦІЯ

Зв'язування між об'єктами (ЗМО) разом з іншими метриками використовується для оцінювання помилок, уразливостей та інших показників якості програмних систем, у тому числі з відкритим кодом. На рівні класу метрика зв'язування між об'єктами – це кількість класів, пов’язаних із даним класом. На рівні застосунку цей показник визначає середню кількість класів, використаних на клас. Відомо, що значення зв'язування між об'єктами від одного до чотирьох є добрим. Однак існують застосунки на Java, для яких значення метрики зв'язування між об'єктами на рівні застосунку перевищує чотири, наприклад, три популярні програми з відкритим кодом, розроблені на Java: FreeMind, jEdit і TuxGuitar. Тому, на нашу думку, наведений вище інтервал для зв'язування між об'єктами потребує уточнення для рівня застосунку. Щоб знайти рекомендовані значення для середнього зв'язування між об'єктами застосунку, ми запропонували застосувати довірчі та прогнозні інтервали. Середнє значення зв'язування між об'єктами застосунку з довірчого інтервалу є добрим, оскільки цей інтервал вказує на те, наскільки достовірною є оцінка для всіх застосунків. Середнє значення зв'язування між об'єктами вище верхньої межі інтервалу прогнозування може означати, що деякі класи надто тісно пов’язані з іншими в застосунку. Ми оцінили довірчі та прогнозні інтервали середнього зв'язування між об'єктами за допомогою нормалізуючих перетворень для вибірки даних зі ста застосунків з відкритим кодом, розроблених на Java, розміщених на GitHub. Порівняння із середніми значеннями зв'язування між об'єктами трьох популярних додатків з відкритим кодом, розроблених на Java, ілюструє застосовність запропонованих індикаторів якості у формі довірчих інтервалів і інтервалів прогнозування середнього зв'язування між об'єктами.

Ключові слова: статистичне оцінювання; програмна метрика; зв'язування між об'єктами; застосунок з відкритим кодом; Java

ABOUT THE AUTHORS



Sergiy B. Prykhodko - Doctor of Science (Eng), Professor, Head of Department of Software for Automated Systems. Admiral Makarov National University of Shipbuilding. 9, Heroes of Ukraine Ave. Mykolaiv, 54007, Ukraine
ORCID: https://orcid.org/0000-0002-2325-018X; sergiy.prykhodko@nuos.edu.ua. Scopus Author ID: 55225622100
***Research field*:** Mathematical modeling; computer simulation; stochastic processes; multivariate statistical analysis; system identification; parameter estimation, nonlinear stochastic differential equations; software metrics estimation

Приходько Сергій Борисович - доктор технічних наук, професор, завідувач кафедри Програмного забезпечення автоматизованих систем Національного університету кораблебудування імені адмірала Макарова, пр. Героїв України, 9. Миколаїв, 54007, Україна



Kateryna S. Prykhodko - PhD, Associate Professor of Department of Information Control Systems and Technology. Admiral Makarov National University of Shipbuilding. 9, Heroes of Ukraine Ave. Mykolaiv, 54007, Ukraine
ORCID: https://orcid.org/0000-0003-0310-5724; kateryna.s.prykhodko@gmail.com. Scopus Author ID: 57200139991
Research field: Mathematical modeling and computer simulation of random variables and stochastic processes; multivariate statistical analysis

Приходько Катерина Сергіївна - кандидат технічних наук, доцент кафедри Інформаційних управляючих систем та технологій Національного університету кораблебудування імені адмірала Макарова, пр. Героїв України, 9. Миколаїв, 54007, Україна



Tetiana G. Smykodub - Lecture of Department of Software for Automated Systems. Admiral Makarov National University of Shipbuilding. 9, Heroes of Ukraine Ave. Mykolaiv, 54007, Ukraine
ORCID: http://orcid.org/0000-0002-4254-5477?lang=en; tgsmyk@gmail.com. Scopus Author ID: 57200139871
Research field: Mathematical modeling and computer simulation of random variables; software metrics estimation

Смикодуб Тетяна Георгіївна - старший викладач кафедри Програмного забезпечення автоматизованих систем Національного університету кораблебудування імені адмірала Макарова, пр. Героїв України, 9. Миколаїв, 54007, Україна