Вісник Тернопільського національного технічного університету
https://doi.org/10.33108/visnyk_tntu
Scientific Journal of the Ternopil National Technical University
2022, № 3 (107) https://doi.org/10.33108/visnyk_tntu2022.03
*ISSN 2522-4433. Web: **visnyk.tntu.edu.ua***

UDC 539.3

TECHNOLOGY OF AUTOCLASSIFICATION OF CHANGES IN THE PROCESS OF MULTICOMPONENT SOFTWARE DEVELOPMENT

Andrii Boichuk[1]; Serhii Vashchyshak[1]; Taras Styslo[1]; Oleh Pashkevych[1]; Tetiana Boichuk[2]; Vitalii Vashchynskyi[3]

[1]King Danylo University, Ivano-Frankivsk, Ukraine
[2]Ivano-Frankivsk College of the Lviv National Environmental University, Ivano-Frankivsk, Ukraine
[3]Lviv Polytechnic National university, Lviv, Ukraine

Summary. *The paper proposes an automated method of classification of source code changes, which consists of two steps – clustering and comparison of clusters of classes. The currently existing methods of improving component software development are analyzed. Based on the analysis, it was established that the optimal method of increasing the productivity of the analysis of changes is the clustering of these changes. A method is proposed, according to which the distribution of changes by clusters is carried out automatically. Their comparison to classes is carried out by an expert. It is shown that the automation of the distribution of changes by clusters significantly reduces the time of examination of code changes, which makes it possible to use the obtained results to improve the quality of software during the development of complex software complexes. The results obtained in the course of the work provide an idea of possible data clustering algorithms with further analysis of the obtained set of clusters according to their parameters. Also, on the basis of the conducted research, the results of the comparison of the classifications of changes in the software system with open source code, performed using the proposed automated method and manually, are given. It is shown that the task of controlling changes that are undesirable at the current stage of development is solved significantly more effectively using the proposed method compared to a full examination of changes, as it allows identifying changes of classes prohibited at the current stage of development with less time spent. The application of the method in practice allows to improve the quality of the code due to the increase in the efficiency of the process of its examination. Using the approach proposed in the paper, the examination process under time constraints can be built more efficiently by selecting changes of the most important classes of changes. It has been proven that the method works perfectly if the same type of changes are analyzed, and when the changes combine heterogeneous code modifications, the quality of the automated classification deteriorates. The obtained results make it possible to extend the application of this method to other software complexes and systems, provided that differences in data types and their parameters are taken into account.*

Key words: *software, software quality, testing, clustering, automation.*

https://doi.org/10.33108/visnyk_tntu2022.03.099 *Received 09.06.2022*

Problem Setting. Today's software developers work with a very large amount of source code. The complexity of modern software complexes is the reason for processing a large amount of data, which makes it difficult to understand and analyze massive code, and, as a result, complicates its quality control. In the software quality control process, source code examination (Code Review) plays an important role. To simplify, code reviews are often limited only to the review of its changes in the development process, since code development usually occurs in an iterative way and is reduced to making changes (including new functionality). However, examination of changes is usually difficult due to the large number of such changes and the limitation of the expert's working time. Therefore, it is necessary to carry out a selective examination of changes. The criterion for selecting changes can be belonging to a certain class.

Corresponding author: Andrii Boichuk; e-mail: andrii.m.boichuk@ukd.edu.ua

This work is devoted to the development of algorithms and the description of the processes of automatic distribution of changes in the source code.

Analysis of notable research results. In the general implementation, the following methods of automating the classified change of source code are practical: a heuristic method of searching for characteristic words in the comments of changes, a heuristic method of searching and classifying refactorings based on the value of certain metrics; method of comparing syntactic trees of code versions; a method of analyzing the syntactic difference of a code version using built-in tags, a method based on the implementation of a version control system that stores abstract syntactic tree codes obtained on the basis of data from the development environment and a method of classifying changes based on the possible presence of errors in them. As a rule, the disadvantages of these methods are their algorithmic complexity, dependence on the programming language, and even lack of adaptability. The use of syntactic methods of classification of changes is justified only for the analysis of simple changes. Also, the disadvantage is the need for redundant storage and processing of a complete set of modifications in the version control system made by the user during development, since not all of them are stored in the final code.

This article proposes a method for classifying source code changes based on metrics clustering, which is free from the above-mentioned drawbacks, and which, unlike the above methods, can be applied to a wide range of applied software problems. Saving source code changes in the version control system opens up wide possibilities for analyzing the history of changes in the software system. Analysis of the history of the software system allows you to obtain information about the patterns of the development process of the software system in the past, which can be used to improve the development process in the future.

Description of results. The method of automated classification of software code changes proposed in this work allows to increase the productivity of development team members due to partial automation of examination of source code changes. Code reviews can be divided into two categories: formal inspections and informal reviews. Formal inspections are examinations carried out by specially trained people who look for defects according to a strictly defined process [1]. For example, Fegan's inspection [2] can be attributed to the formal methods of examination of the source code. Informal code review is not subject to a strictly defined process, and can be a component of the development process, for example, in the form of the following forms:

- code examination in pairs, when the author of the code and another person review the code together;
- examination of the code for a request that can be automatically sent, for example, by the change control system, when a new code change appears in it;
- pair programming [3], in which the entire process of writing code takes place in pairs;
- code testing using automated means, when software tools are used to check for common errors in the code.

There are many studies that confirm the fact that testing the source code, both in the form of inspections and informally, allows you to save significant money in the development of a software system by eliminating errors at the early stages of software product development [4, 5]. A team of programmers can generate a large number of changes, which can lead to significant time spent on their testing and verification.

Let's consider a certain software system P in the process of its development over time. The state of this system at each moment of time t is given by its current code S_t. For convenience, let's denote a set of unchanging states S_t during successive time intervals $t \in (t_{r-1}..t..._r)$ as S_r, r – integer, $1 \le r \le N$, N – the total number of different code states.

Changing the source code is called mapping δ_r, which translates the code of the previous state into a modified state. With the help of each code change, the developer achieves a certain goal of the development of the software system. When implementing software systems in practice, it becomes clear that often the goals achieved with the help of various changes have a lot in common with each other. To simplify the tasks of code examination, it is advisable to divide changes into classes according to the selected goals. Each change according to its introduction can be assigned to some class, where .

At the same time, it represents many classes of changes specific to the software system. The following classes of changes are most common: implementation of new functionality, correction of logic, refactoring, removal of redundant code, formatting of code.

It should be noted that the result of expert classification does not depend only on the analyzed software system and set of changes, but also on a specific expert. Therefore, it can be expected that according to the classification of some changes by different experts, a different distribution of changes by given classes is possible [6]. The method of clustering change metrics allows to level the subjectivity of classification, and, as shown in the thesis, partial automation of the classification of source code changes can be provided. Classification automation is performed taking into account the following hypothesis. In this paper, it is proposed to build the automation of the classification of source code changes based on the clustering of change metrics.

The scheme of method is shown in fig. 1.

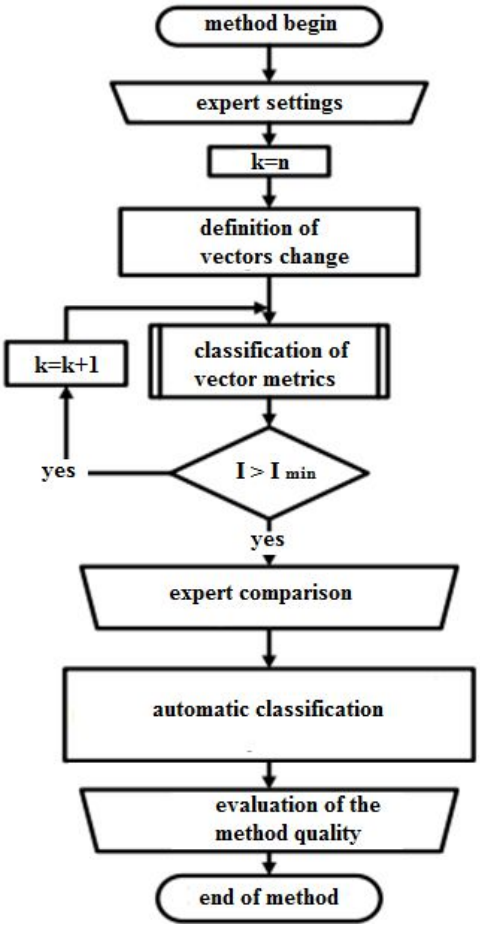

Figure 1. Algorithm of change classification method using clustering

Changes for classification, a set of expert classes, a set of change metrics and the number of clusters, which is initially set equal to the number of expert classes, are used as input data of the method. The output of the method is a set of classified changes. We will describe the stages of the method that correspond to the blocks of the scheme shown in Fig. 1. During the adjustment process, the expert selects a set of change metrics that allows to ensure the specified levels of the minimum and maximum classification quality criteria. For each change, a vector is formed from the values of the metrics selected at the method setup stage. Clustering will be performed using the k-means algorithm using the cosine measure of closeness of changes. The choice of the k-means clustering algorithm is explained by its ease of use with acceptable quality of the result. When using it, an assumption about the expected number of clusters is necessary. At this stage, each cluster is assigned an expert class. With this comparison, correspondence between clusters and classes of changes is established. In this work, it is proposed to compare clusters to classes on the basis of a selective expert classification of changes from each cluster. The automated change classification process can be considered complete if the valuesof the quality criteria calculated using the following formulas satisfy the specified levels.

Otherwise, it is necessary to repeat the stage of expert adjustment described in the first points of the algorithm. The result of the automatic classification of changes based on the mapping of the clusters to which they belong to the classes is a built classification of changes according to the given expert classes. CLUTO, a specialized software tool developed for data clustering, was used to evaluate the practical value of implementing the given algorithms. Its feature is the ability to set the parameter «number of trials» («ntrials»), which indicates to calculate a given number of cluster solutions in different ways and choose the best one based on the maximization of the value of the clustering quality functional. This makes it possible to improve the quality of clustering by reducing the probability of the functional value hitting a local minimum. In addition, the tool allows you to set different algorithms for clustering, as well as to choose the degree of proximity of clustering objects. Studies have shown that the method allows to reduce the time of classification of changes in comparison with «purely manual» examination. The hypothesis about the possibility of automated classification by the method of clustering of change metrics for the values of quality criteria for the distribution of changes by clusters was also confirmed.

Conclusions. The possibility of partially automating the classification of source code changes by means of metric clustering is substantiated. The choice of the k-means method with a measure of object proximity for clustering based on the cosine of the angle between vectors of change metrics is justified. A method of automated classification of source code changes based on clustering of change metrics has been developed, which allows to reduce the number of changes for manual classification.

References

1. International standard. ISO/IEC/IEEE 12207:2017 Systems and software engineering – Software life cycle processes. 2017. 145 p.
2. Andrashov A. A. Fasetno-iyerarkhicheskiye semanticheskiye struktury v zadachakh obespecheniya kachestva programmnogo obespecheniya. Integrirovannyye tekhnologii v mashinostroyenii "ÍKTM-2008": mater. Mizhnar. nauk.-tekhn. konf. (m. Kharkív, 2008.). Kharkív. 2008. T. 2. P. 204.
3. Gordieiev O., Kharchenko V., Fominykh N., Sklyar V. Evolution of software Quality Models in Context of the Standard ISO 25010: In proceedings of the International Conference on Dependability on Complex Systems DepCoS – RELCOMEX (DepCOS) (Brunow, Poland, June 30 July 4, 2014.). Brunow, 2014. P. 223–233. DOI: https://doi.org/10.1007/978-3-319-07013-1_21
4. Bouraou N., Tsybulnik S., Rupich S. (2017) Problems of Intellectualizing in SHM Systems: Estimation, Prediction, Multi-Class Recognition. Scientific Journal of TNTU (Tern.). Vol. 88. No. 4. P. 135–144. DOI: https://doi.org/10.33108/visnyk_tntu2017.04.135

5. Palamar A. Control system simulation by modular uninterruptible power supply unit with adaptive regulation function. Scientific Journal of TNTU (Tern.). Vol. 98. No. 2. 2020. P. 129–136. DOI: https://doi.org/10.33108/visnyk_tntu2020.02.129
6. Blizard W. The Development of Multiset Theory, Notre Dame J. of Formal Logic. Vol. 30. No. 1. 1989. P. 36–66. DOI: https://doi.org/10.1305/ndjfl/1093634995

Список використаних джерел

1. International standard. ISO/IEC/IEEE 12207:2017 Systems and software engineering – Software life cycle processes. 2017. 145 p.
2. Andrashov A. A. Taksonomicheskiye modeli profilirovaniya trebovaniy informatsionno-upravlyayushchikh sistem kriticheskogo primeneniya. Radíoyelektronni i komp'yuterni sistemi. 2010. № 7 (48). P. 104–108.
3. Gordieiev O., Kharchenko V., Fominykh N., Sklyar V. Evolution of software Quality Models in Context of the Standard ISO 25010: In proceedings of the International Conference on Dependability on Complex Systems DepCoS – RELCOMEX (DepCOS) (Brunow, Poland, June 30 July 4, 2014.). Brunow, 2014. P. 223–233. DOI: https://doi.org/10.1007/978-3-319-07013-1_21
4. Bouraou N., Tsybulnik S., Rupich S. (2017) Problems of Intellectualizing in SHM Systems: Estimation, Prediction, Multi-Class Recognition. Scientific Journal of TNTU (Tern.). Vol. 88. No. 4. P. 135–144. DOI: https://doi.org/10.33108/visnyk_tntu2017.04.135
5. Palamar A. Control system simulation by modular uninterruptible power supply unit with adaptive regulation function. Scientific Journal of TNTU (Tern.). Vol. 98. No. 2. 2020. P. 129–136. DOI: https://doi.org/10.33108/visnyk_tntu2020.02.129
6. Blizard W. The Development of Multiset Theory. Notre Dame J. of Formal Logic. Vol. 30. No. 1. 1989. P. 36–66. DOI: https://doi.org/10.1305/ndjfl/1093634995

УДК 539.3

ТЕХНОЛОГІЯ АВТОКЛАСИФІКАЦІЇ ЗМІН У ПРОЦЕСІ РОЗРОБКИ БАГАТОКОМПОНЕНТНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Андрій Бойчук[1]; Сергій Ващишак[1]; Тарас Стисло[1]; Олег Пашкевич[1]; Тетяна Бойчук[2]; Віталій Ващинський[3]

[1]ЗВО «Університет Короля Данила», Івано-Франківськ, Україна
[2]Івано-Франківський фаховий коледж Львівського національного університету природокористування, Івано-Франківськ, Україна
[3]Національний університет «Львівська політехніка», Львів, Україна

Резюме. *Запропоновано автоматизований метод класифікації змін вихідного коду, що складається з двох кроків – кластеризації та зіставлення кластерів класів. Проаналізовано існуючі на сьогодні методи удосконалення компонентного розроблення програмного забезпечення. На основі аналізу встановлено, що оптимальним методом підвищення продуктивності аналізу змін є їх кластеризація. Запропоновано метод, за яким розподіл змін за кластерами здійснюється автоматично. Зіставлення їх класам виконує експерт. Показано, що автоматизація розподілу змін за кластерами суттєво скорочує час експертизи змін коду, що дає можливість використовувати отримані результати для підвищення якості програмного забезпечення в ході розроблення складних програмних комплексів. Результати, отримані в ході виконання роботи, дають уявлення про можливі алгоритми кластеризації даних з подальшим аналізом отриманого набору кластерів за їх параметрами. Також на основі проведених досліджень наведено результати порівняння класифікацій*

змін у програмній системі з відкритим вихідним кодом, виконані з використанням запропонованого автоматизованого методу та вручну. Показано, що завдання контролю змін, небажаних на поточній стадії розроблення, вирішується суттєво ефективніше за допомогою запропонованого методу порівняно з повною експертизою змін, оскільки дозволяє виділяти зміни класів, заборонених на поточній стадії розроблення з меншимии витратами часу. Застосування методу на практиці дозволяє покращити якість коду завдяки підвищенню ефективності процесу його експертизи. Використовуючи пропонований у роботі підхід, процес експертизи в умовах обмеження часу можна будувати ефективніше за допомогою відбору змін найважливіших класів змін. Доведено, що метод відмінно працює, якщо аналізуються однотипні зміни, а коли у змінах поєднуються різнорідні модифікації коду, та якість автоматизованої класифікації погіршується. Отримані результати дають можливість розширення застосування даного методу для інших програмних комплексів та систем за умови врахування відмінностей у типах даних та їх параметрів.

Ключові слова: *програмне забезпечення, якість програмного забезпечення, тестування, кластеризація, автоматизація.*

https://doi.org/10.33108/visnyk_tntu2022.03.099 *Отримано 09.06.2022*