Technoarete Transactions on Advances in Computer Applications
Vol-1, Issue-1, March 2022
e-ISSN: 2583-3472

Implication of Code Refactoring in Optimizing Software Security

Siddth Kumar Chhajer [1], Rudra Bhanu Satpathy [2]

[1] MBA, Marketing, St. Peter's University, Chennai.
[2] M. Tech, Electrical Electronics and Communications Engineering, St. Peter's University, Chennai.
*Corresponding Author Email: [1]siddth2011@gmail.com

Abstract

In this research article the researcher has assessed the significance of code refactoring in optimizing the software security. It is essential for the developers to ensure that the security attributes have been addressed while developing software. It can assist the developers to detect the problems within the software that may impose several burdens in future. Apart from that, it also assists in sabotaging the risk factors through providing excellent design which can prevent the threat of any such malware. Moreover, the study has incorporated a secondary method of data collection. In this context, the result of the study has highlighted that the inception of code refactoring is highly recommended in optimizing software security.

Keywords

Code refactoring, Computer science. developer, Malware, Program, security, Software, Bugs,

INTRODUCTION

Code refactoring has occupied an essential role for increasing and optimizing the security-awareness of software. In addition to that, enhancing the spreading of critical security classes within the design of the software can easily enhance its modularity resulting in decreasing the resilience of the software to prevent such attacks. Apart from that, an effective code refactoring process can easily enhance the readability and reduce the complexity of a code or software. Maintainability and extensibility can both be improved by making the underlying architecture or design model simpler, clearer or more expressive within the source code. However, in this research article the researcher has tried to detect the significance of code refactoring in optimizing software security. Based on this note, the further part of this study has been demonstrated.

Aim: *The research article has aimed to detect the significance of applying code refactoring methods in order to improve software security.*

The concept and significance of code refactoring in software security

Software's source code can be refectories without affecting the application's overall performance or functionality. As per the words of Kaur and Singh [5], code refactoring is intended to enhance the code's readability, complexity, maintainability, and extensibility through removing the unnecessary codes. On the other hand, few prior articles have showcased that code refactoring refers to making changes, mainly restructuring, tidying up and editing in an existing base of code. However, in order to modify the underlying structure of a code without hindering the internal structure an individual or any IT organization can adapt the process of Code refactoring. In addition to this, this complex method has aimed to enhance the modernizing of the software as well as reducing the potential hazards.

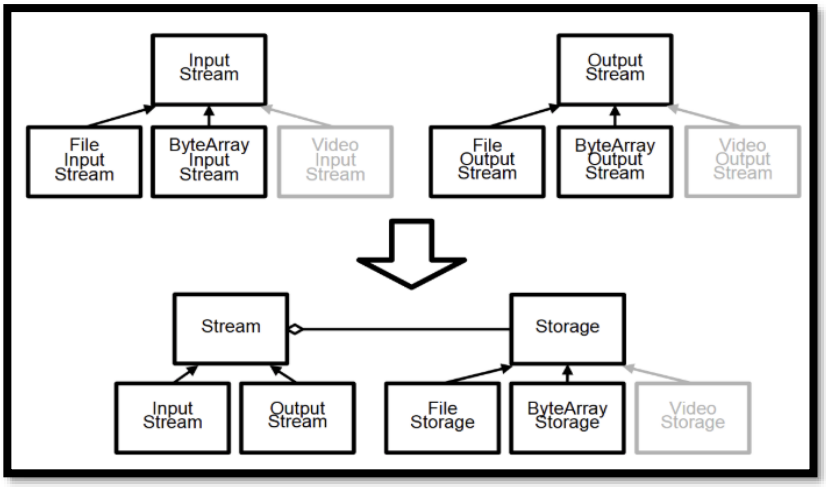

Figure 1: the process of Code Refactoring [6]

e-ISSN: 2583-3472

On the other hand, these methods are generally utilized in the context of improvising the maintainability, scalability, security and performance of an existing software or system. However, it has been mentioned above that code refactoring is a process that does not modify or evolve the external behaviour of a software or system. On this note, the following part the researcher has illustrated the significance of Code refactoring process to optimize security and enhance the performance of software.

- **Support and code changes have been made easier through the usage of code refactoring**

Updating and improvising clean code has become significantly easier through adapting the complex method of code refactoring. As denoted by Lacerda *et al.* [6], it is possible for developers to make new features accessible in a short period of time and the support budget will be saved as a result.

- **The application of code refactoring method can save money and time in long term**

The process of refactoring existing code makes it easier to incorporate new features and minimize the risk of future problems. However, developers do not have to spend time untangling tangled code or repairing detects [8]. Instead of that, they may begin immediately creating the desired functionality of a software or system.

- **The inception of code refactoring method has reduced the complexity of a software**

Adding a new employee or reorganizing the development team will help new developers to gain a better understanding of the code and implement changes more quickly.

- **It can also enhance the stability and extensibility**

In the time period of developing or redesigning software, programmers simply avoid making alterations to a few nonessential codes since the program does not gain a proper overview regarding the consequences the modification will lead to. However, the same can be said about the issue of increasing the system's capacity and that is another advantage of code refactoring as it removes the barrier of a complex code.

In the above part, the researcher has outlined the significance of code refactoring for enhancing the performance of a software or system. However, in the following part the researcher has outlined the significance of code refactoring in optimizing security of software.

Each year, **NIST** estimates that patching security vulnerabilities or dealing with the fallout from the security issues costs the US economy **$60 billion** in lost productivity. The degree to which a system is susceptible to these flaws is determined by the designer and programmer choices made during its development. As per the words of [1], programmer productivity and project failure could occur due to the poor quality of the code, since maintenance accounts for more than 70% of the lifetime expenditure for a software project.

According to [1] **ISO and IEC-25000 SQuaRE** is a standard for assessing the quality of a software product. In this regard, the inception of the code refactoring process can aid an individual or programmer to ensure the quality of the products that have been utilized in order to develop software. As a result, it may reduce the risk of software failure and bugs in a significant manner. On the other hand, as stated by [2], security is a novel attribute that measures the way a programme or software can withstand and prevent assaults and dangers from such hazards and bugs. Therefore, while developing a program it is essential that the attributes of security measurement have been taken under consideration. Apart from that, software quality metrics such as modularity are thought to have a favourable effect on the security, milking the design more robust and impervious to assaults. In addition to that, through the inception of the code refactoring process, developers and designers can overlook design fragments that include data and logic relevant to security qualities, making them overexposed while enhancing the quality of other elements of software. However, in order to make the code more reusable, a developer can establish a hierarchy inside a collection of classes. However, despite the super class includes key characteristics and methods these operations may widen the attractive surface in a more credible manner.

Nevertheless, a practical illustration of this is the effect of increased modularity on spreading security critical file dependencies to many other components of a system. On the other hand, a security critical file includes information and logic that might possibly be exploited to breach basic security quality ties such as confidentiality, availability and integrity of a program. According to the words of [9], many software systems are improved through the usage of code refactoring process to improvise their structure while maintaining their functionality. However, the influence of code refactoring on security is poorly understood and under researched to yet. Students based on definition have estimated the influence of several security metrics on some refactoring procedures, without empirical validation of these assumptions on actual software projects. In addition to that, no prior study has examined the relationship among security measurements and quality characteristics and no tool has been developed to offer refactoring in light of developer preferences and potential conflict from both quality and security viewpoints.

METHODS AND MATERIALS

Existing code refactoring strategies in the context of a literature review have been documented in both written and visual form. In addition, the researcher has utilized three steps of the review procedure; in this regard a basic framework for this review has been constructed. A literature review is constrained through the database to search and the phrases have been utilized based on the keywords. ACM digital Library, Google scholar and ProQuest have been utilized, as these are the most popular scientific libraries within the indexing systems in computer science to resolve

e-ISSN: 2583-3472

any query. In order to execute the aim of this research these datasets and indexing systems have been shown to be the most useful method.

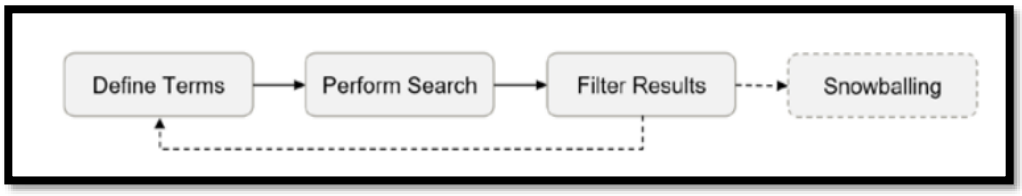

Figure 2: Strategy used for collecting and assessing the information of the study [3]

However, the researcher has determined that, in order to gather information from the previous literature, the sources have to be peer reviewed and published in the English language. In addition to that, the researcher has also ensured that the abstract of the studies had to clearly demonstrate a contribution to the research issues and the suggested strategy had to be substantiated [3]. On the other hand, utilizing a snowballing method has been also utilized within this research study in order to enhance the outcome of the study in a more credible manner. As a consequence of this "snowballing" action, the original set of search results has grown considerably. Finally, the quality of the research has been evaluated by looking at the technical depth, freshness, and applicability of the information provided.

RESULT AND DISCUSSION

The result of the study has outlined that the code refactoring to increase field security has the highest contribution with the average of the essential security criteria. Apart from that, frequently using this code refactoring type is anticipated to minimize access to characteristics. Hence, utilizing the code refactoring method can reduce the attractive surface that might be exploited by malicious code. Same holds true for increasing the security method, the code refactoring process creates a favourable link with security average and refactoring which assists in preventing such threats. According to the words of [4], encapsulate field, push down field and push method are all known to limit the chances of abstraction in order to safeguard fields and methods from being tampered with in the refactoring process. Refactoring has the primary advantage of removing unmaintained code and it also reduces the technical debt of a program. As a result, the code is simpler to comprehend and maintain both the original developer as well as future developers that may be working on developing a software project.

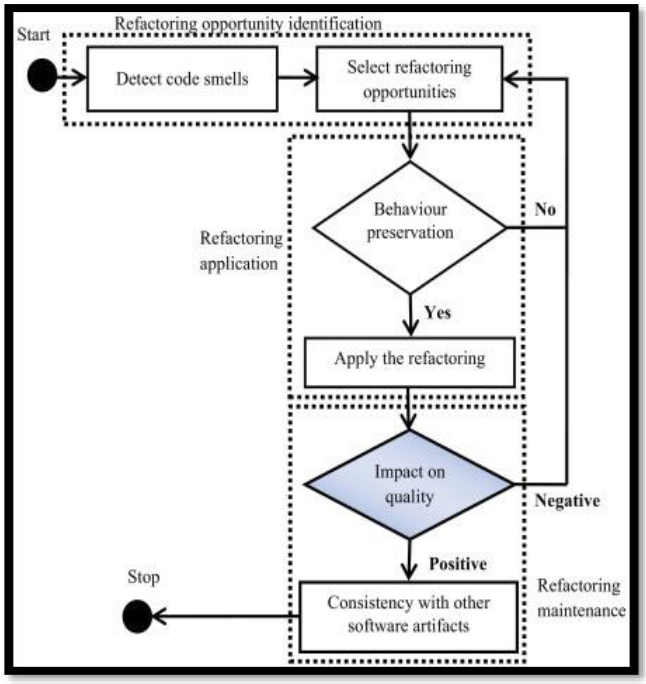

Figure 3: The process of Code Refactoring in optimizing software security [7]

The internal designing of a system or program can be more expressive unless the code is less difficult. In order to make things even better, a programme can be utilized as a template for other programmes that need to be written from scratch. As per the words of [7], code refactoring can also assist developers to gain a better understanding regarding the code and design choice they are making. On the other hand, once it comes to learning the way to write a code, beginners and

e-ISSN: 2583-3472

experts alike may benefit from understanding the way others have done it before them. Even, unless one developer is accountable for the code as a whole, this might help in fostering a feeling of communal responsibility. In addition to that, while it may seem as a low priority operation, refactoring modifies little sections of code that have no such immediate impact on the experience of the user. Changes such as these have a huge impact over time and can result in a more effective team and approach to programming. Apart from that, it also assists in reducing the risk factor through detecting and sabotaging the risks of software in a more efficient manner.

CONCLUSION

This research article is based on measuring the significance of code refactoring for optimizing the security of software. On this note, the researcher has discovered that the inception of code reformation is an essential method that can assist the developers to detect the potential hazards and bugs of software that may hinder the quality of the software. In addition to that, the inception of code refactoring can also assist the developers to ensure the quality of the materials that have been utilized in order to create a system or software. However, based on the findings of the study, it is fair to anticipate or conclude that the inception of Code refactoring is highly beneficial for reducing the security problems of software.

REFERENCES

[1] AlOmar, E.A., Mkaouer, M.W. and Ouni, A., 2021. Toward the automatic classification of self-affirmed refactoring. *Journal of Systems and Software*, *171*, p.110821.

[2] AlOmar, E.A., Peruma, A., Mkaouer, M.W., Newman, C., Ouni, A. and Kessentini, M., 2021. How we refactor and how we document it? On the use of supervised machine learning algorithms to classify refactoring documentation. *Expert Systems with Applications*, *167*, p.114176.

[3] Ampatzoglou, A., Mittas, N., Tsintzira, A.A., Ampatzoglou, A., Arvanitou, E.M., Chatzigeorgiou, A., Avgeriou, P. and Angelis, L., 2020. Exploring the relation between technical debt principal and interest: An empirical approach. *Information and Software Technology*, *128*, p.106391.

[4] García-Mireles, G.A., Moraga, M.Á., García, F., Calero, C. and Piattini, M., 2018. Interactions between environmental sustainability goals and software product quality: A mapping study. *Information and Software Technology*, *95*, pp.108-129.

[5] Kaur, S. and Singh, P., 2019. How does object-oriented code refactoring influence software quality? Research landscape and challenges. *Journal of Systems and Software*, *157*, p.110394.

[6] Lacerda, G., Petrillo, F., Pimenta, M. and Guéhéneuc, Y.G., 2020. Code smells and refactoring: A tertiary systematic review of challenges and observations. *Journal of Systems and Software*, *167*, p.110610.

[7] Lenarduzzi, V., Besker, T., Taibi, D., Martini, A. and Fontana, F.A., 2021. A systematic literature review on technical debt prioritization: Strategies, processes, factors, and tools. *Journal of Systems and Software*, *171*, p.110827.

[8] Palomba, F., Di Nucci, D., Panichella, A., Zaidman, A. and De Lucia, A., 2019. On the impact of code smells on the energy consumption of mobile applications. *Information and Software Technology*, *105*, pp.43-55.

[9] Smiari, P., Bibi, S., Ampatzoglou, A. and Arvanitou, E.M., 2022. Refactoring embedded software: A study in healthcare domain. *Information and Software Technology*, *143*, p.106760.