What Is Software Maintainability

dr. Adam Vanya

INTRODUCTION

Whether you are an owner, a manager or a developer of a software company, the extent to which you succeed in reaching your short and long term professional goals also depends on how maintainable the developed software is.

Since software maintainability is such an influential concept, we strongly believe that everyone should have a clear picture about what it exactly is, which factors are influencing it and how it can affect your goals set. In order to minimize the negative effects of a poorly maintainable system, one also needs to know what the current level of software maintainability is, evaluate if that level is sufficient and take the necessary corrective actions.

By publishing a series of blog posts, we intend to provide you with all the relevant information on software maintainability that is needed to understand and control it. This very first blog post introduces the subject of software maintainability.

SOFTWARE MAINTAINABILITY

Regardless of the software development methodology (agile, waterfall, etc.) applied, developers need to analyze, modify and test existing code. These activities are executed as part of the software maintenance process. The need to maintain a piece of software can come from errors or bugs to be fixed (corrective maintenance), changes to the different environments (organizational, infrastructural, legislative, etc.) of the software (adaptive maintenance), new and evolved end-user requirements (perfective maintenance), and foreseen future changes (preventive maintenance). The mentioned types of maintenance are further detailed in [1]. The time and valuable effort one needs to spend during the software maintenance process depends primarily on how maintainable the software is.

Maintainability is a complex concept. Professionals have been trying to define it for nearly a century. As a result, many definitions were created. One of the most commonly accepted and applied one is from the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). They define maintainability in their ISO/IEC 25010:2011 standard published in [2] as a “degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers”. In our series of blog posts on software maintainability we will refer to maintainability according to this definition.

Although software maintainability is a fundamental aspect of software product quality, it is not the only one. A software product needs to fulfill functional requirements, prevent unwanted access to core values,

be available according to the needs of the users, serve a prescribed amount of incoming requests within a specified amount of time, offer functionalities that can be discovered and used easily, etc. The ISO/IEC 25010:2011 standard identifies in total 8 quality aspects: maintainability, security, reliability, performance efficiency, portability, usability, compatibility and functional suitability. These quality aspects all contribute to the overall product quality.

A change or an extension to a software may have implications for multiple quality aspects. Some change may improve one quality aspect while negatively effecting another one at the same time. In those cases, trade-offs have to be made. From the perspective of maintainability this means that some maintainability related problems may not get fixed on purpose, to avoid the negative consequences for other quality aspects. Bass et al. in [3] further elaborates on the trade-offs mentioned.

The quality of a software is good if it fulfills the needs of its stakeholders. Each stakeholder has its own set of interest and these interests drive what is required from the software. A stakeholder can be amongst others a user who uses the software directly or indirectly to achieve a goal, a customer for whom a software is developed, a developer creating and modifying the software, an owner or investor financing the development activities. From the different needs of the stakeholders follow that software quality is subjective: a given software may be of excellent quality for one stakeholder while being completely disappointing for another one. In our next blog posts on maintainability, we focus on the maintainability quality aspect and the related stakeholder needs.

SUMMARY AND CONCLUSIONS

One way or another, software maintainability is effecting you. It can have an significant impact on your carrier if you are working for a software development company and it is a major factor to consider when you invest in such companies or buy custom software. Therefore, we believe maintainability has to be understood and explicitly be controlled to reach your goals.

To help you getting in control, we first elaborated on what software maintainability is. We also noted that maintainability is not the only quality aspect to be controlled. Based on this, we argued that control actions should be selected and executed carefully since they can have effects on the other quality aspects as well.

The next blog post in our series on software maintainability will elaborate further on the factors influencing maintainability. With these blog posts, we hope to increase the awareness about software maintainability, help you control this concept and start a healthy and constructive discussions about it.

In case there are certain things described in this blog post you do not or not completely agree with or if you think it should be extended in some ways, please feel free to contact us and tell us about your idea: adam.vanya@frontendart.com

REFERENCES

[1] ISO/IEC 14764 (2021). ISO/IEC 14764:2022, Software Engineering — Software Life Cycle Processes — Maintenance

[2] ISO/IEC 25010 (2011). ISO/IEC 25010:2011, Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models

[3] Bass, L., Clements, P., & Kazman, R. (2021). Software Architecture in Practice (Fourth). Addison-Wesley Professional.