Proceedings in Manufacturing Systems, Volume 17, Issue 4, 2022, 129-134 ISSN 2067-9238

CAPABILITY MATURITY MODEL AND TESTABILITY MATURITY MODEL – A COMPARATIVE APPROACH

Crina DUTA[1], Nicoleta CARUTASU[2,*], Iulia STANICA[3]

[1)] PhD Student, Doctoral School of Industrial Engineering and Robotics, University "Politehnica" of Bucharest, Bucharest, Romania
[2] Prof., PhD, Department of Robots and Manufacturing Systems, University "Politehnica" of Bucharest, Romania
[3)] Assistant Prof., PhD Department of Engineering in Foreign Languages, University "Politehnica" of Bucharest, Romania

Abstract: *Inmost fields of activity, devices are used, and their key element is the software used to meet the needs of the users. Since users consider these devices to be the necessary tools to use to support them in the way they carry out their activity, the emphasis of software product developer organizations is placed on the quality that the product has when it becomes available on the market. During the development life cycle of a software product, the gap between the stage of practice and the stage of theory was observed, and thus measures were taken to improve the development processes. In order to improve the development processes of software products as well as their testing, optimization models such as Capability Maturity Model and Testability Maturity Model were created. The main similarity between these two methods is due to the fact that the Testability Maturity Model is complementary to the Capability Maturity Model and both work on five maturity levels. These models bring with them a series of benefits to the organizations that follow them and put them into practice, but at the same time, they also have weak points for which coverage solutions must be found. The aim of the paper is to carry out a comparative analysis between these two models and to identify the weak points of these models.*

Key words: *software processes, management, testing process, quality improvement, market.*

1. INTRODUCTION

Regardless of the device used, whether it is a phone, an e-commerce application, a robot or a device used in the industrial manufacturing process, it works based on a software application that needs to be developed. The software development process is a complex process consisting of several stages. The software process also contains the activities methods and practices used by an organization to develop and maintain a software product. The first stage in a software development process is research. After all the details have been established, the development begins, meaning the part of writing and maintaining the source code followed by the integration and verification stage, consisting of several steps. The first stage in a software development process is research. After all the details have been established, the development begins, meaning the part of writing and maintaining the source code followed by the integration and verification stage. The quality of a software product depends on the way the development process is organized. This is the reason why software process improvement models were created. The purpose of these models is to evaluate and improve software development processes in order to guarantee the quality of a software product. Two essential elements are analyzed when it comes to the quality of a software product: software process capability and software process performance. The capability of a software product describes the possible results expected to be achieved following a software process while the performance of a software product shows the actual results obtained through a software process. Between the possible results and actual results may be differences that show that somewhere, during the development of the product, were some changes or deviations from the development process that impacted the quality of the product.

The present article address the problem of the necessity to improve software processes. The second section of the article presents a brief the state of the art followed by the third section in which the two models, Capability Maturity Model and Testability Maturity Model will be compared. The fourth section presents the emphasis placed by the authors on the identified weak points of these models. At the end of the article some conclusions will be drawn.

2. STATE OF THE ART

The Capability Maturity Model and the Testability Maturity Model were identified and chosen to be analyzed following a research on software process optimization models. The most important model that represented an important source of inspiration in the development of other models is Capability Maturity Model or CMM. A dedicated group of people who spent many hours discussing the model and its features initially created its first version. The main purpose of this model is to provide software organizations with advice and guidance in controlling the development and

[*] Corresponding author: Splaiul Independenței 313, București 060042, Romania,
Tel.: 004021 402 9369,
E-mail address: *ahohenkendl@univ.com* (C. Carutasu).

maintenance processes of software products in order to achieve an evolution towards a culture of software engineering and management excellence. Identifying the most critical aspects for the quality of software products and determining the current level of maturity of the processes are the methods by which CMM helps organizations determine their improvement strategies. CMM describes the process maturity framework of five maturity levels so that organizations can refer to one of them and set strategies to reach the level they want. The CMM is useful because it provides information regarding the understanding of the key practices that are part of effective processes for developing and that are needed to be reach the next maturity level in the CMM, the understanding of the risk of having a particular software organization perform the work of a contract and also by preparing teams to perform software capability evaluations. Figure 1 shows the maturity levels addressed by CMM [1].

Positioned as being complementary to the CMM, the Testability Maturity Model or TMM is a model for test process improvement. According to its reference, TMM uses the concept of maturity levels that help organizations to evaluate processes and to establish their improvement strategies. The difference between TMM and CMM is that TMM focuses especially on testing processes while CMM approaches the entire development process of a software product, not sufficiently addressing the testing part. One of the advantages of TMM is that it can be applied to both static and dynamic testing. This model identified a number of process areas, maturity goals and key practices. According to the TMM, the testing phases start from a debugging and detection-oriented period to a prevention-oriented period. Figure 2 shows the maturity levels addressed by TMM [2].

TIM is another well-known test improvement model and was developed based on CMM and TMM. As the name suggests, the test improvement model helps the organization to improve the tests used in the testing process by focusing on cost-effectiveness and risk management. The test improvement model consists of two components: a scale of levels and key areas and an evaluation procedure [3].

Figure 3 shows that each of the four key areas of TIM represents a scale of improvement for a certain area of importance for testing. We also observe that at the level of the key areas, the objectives are set, and their achievement is done through specific activities and control points [4].

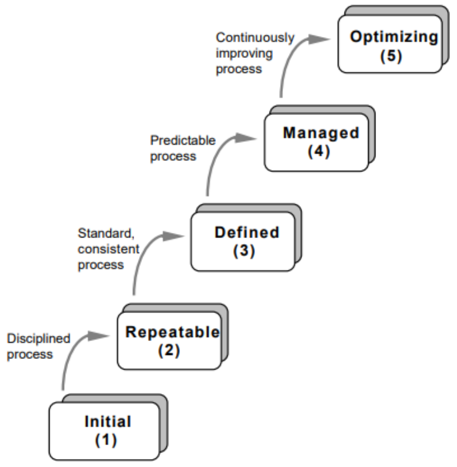

Fig.1. CMM Levels [1].

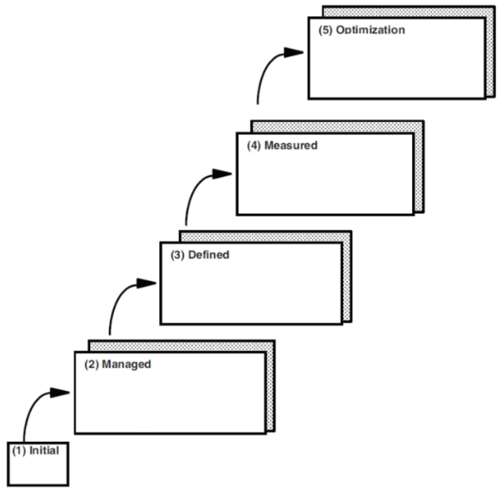

Fig. 2. TMM Levels [2].

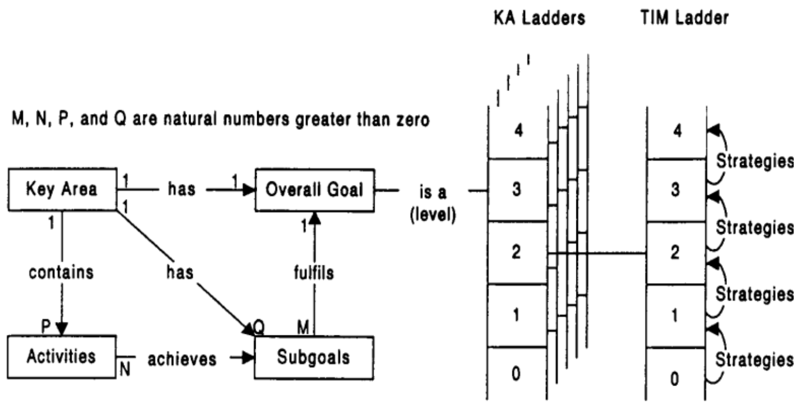

Fig. 3. TIM Structure [3].

The development of the present article, which proposes a comparative approach of the CMM and TMM models, during the research stage required to analyze also works that aimed to compare two or more models to be improved. Based on the analyzed works, the article acquired the appropriate structure to highlight the strengths and weaknesses of the CMM and TMM models and to identify a number of other models developed with the aim of improving processes in a software organization. The research methods applied for this article are analytical and comparative research.

3. THE COMPARATIVE APPROACH

Since the models to be compared are based on maturity levels, it is important to mention from the beginning how to differentiate a mature organization from an immature one. Knowing this difference, the evolution of an organization from a low level of maturity to a higher one can be easily observed.

An immature organization operates with improvised software processes, where attention is directed towards combating incidents more than preventing them. An important aspect related to product quality, in an immature organization, if the deadline is exceeded, the product quality testing stage will be abandoned or reduced.

On the other hand, in a mature organization the software process is well understood by the entire organization, the product quality occupies an important place in the organization's objectives, and the activity of monitoring and improving the processes is continuous. Referring to a level of maturity reached by an organization, we must keep in mind that the level of maturity represents a well-defined evolutionary plateau toward achieving a mature software process [1].

The analysis of these two models begins with the first level of maturity, level 1 or the initial level both in the case of CMM and TMM. According to CMM, the first level at which an organization can be is the one in which there is not yet a stable environment for the development of software applications. The capability of a software product developed by an organization at the initial level of maturity is unpredictable because constant changes are brought to the development processes.

Focusing on the maturity levels of the TMM, the initial level can be described similar to the initial level of the CMM because the testing activity is a chaotic one, often considered the debugging part. The main objective of testing at this level is to verify that the software product works. Defects will not be considered if they are not major and do not completely prevent the operation of the product [2].

The second level of maturity of CMM is called the repeatable level, because the planning and management of new projects is carried out according to the experience of previous similar projects. The implementation of an effective management processes that can later be used in the development of other software products is one of the objectives that organizations must fulfill in order to reach the second level of maturity. Level two is the first level we can discuss about several key process areas which are helping the organization to identify the focus areas in

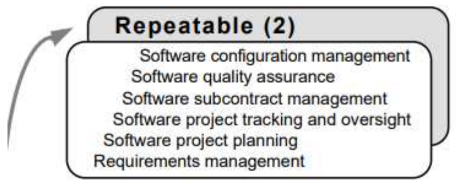
Repeatable (2)
Software configuration management
Software quality assurance
Software subcontract management
Software project tracking and oversight
Software project planning
Requirements management

Fig. 4. Key process Level 2 – CMM [1].

order to improve its software process. Summarily, a key process areas identifies the necessary actions that an organization must perform in order to achieve the objectives of a maturity level and to be able to reach the desired maturity level. Figure 4 shows the key process areas of the second level of the CMM [1].

Analyzing in parallel the second level of TMM, we will identify some similarities. First of all, the second level of TM is called definition level. For organizations whose testing process is at the second level of maturity, the testing process is well delimited from the debugging part. In addition, as in the case of the second level of the CMM, at this level some test plans and a test strategy are established which can later be adapted and reused in the case of a new product. An aspect that differentiates the first level of TMM from the second level is the fact that the focus of the testing is shifted to the fulfillment of the specified requirements.

Figure 5 shows the key process areas of the second level of the TMM [3].

The third level of the CMM, the defined level, requires the documentation of the development and maintenance processes and their introduction into a standard software process for the organization. Also, at this level, employee training programs will be carried out so that they have the necessary knowledge for the position they hold. Management activities as well as software engineering activities are stable and repeatable. These activities are integrated into a coherent and well-defined software process that is based on the business environment and the technical requirements of the project. At this level, the peer reviews appear, and their goal is to eliminate the defects of a software product as early as possible in the development life cycle. For the third level of maturity of the CMM, new key process areas identified and shown in Fig. 6 [1].

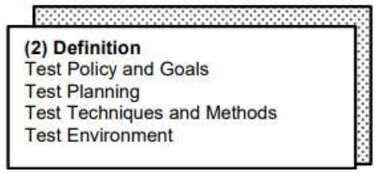

Fig. 5. Key process Level 2 – TMM [3].

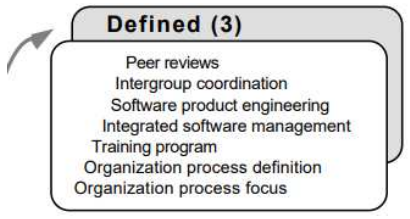
Defined (3)
Peer reviews
Intergroup coordination
Software product engineering
Integrated software management
Training program
Organization process definition
Organization process focus

Fig. 6. Key process Level 3 – CMM [1].

The third level of TMM is the integration level in which testing is fully integrated into the software development life cycle. As in the case of CMM, at this level there is a training program for the people who perform the testing part. In addition, the reviews are present at this level, but they are not carried out according to predefined procedures. This time, the testing focus is on invalid testing. The key process areas of this level can be seen in Fig. 7 [2].

Advancing to the fourth level of the CMM, the managed level it is noticed that the organizations that have reached this level set their quantitative quality goals. The productivity and quality are measured as part of an organizational measurement program. The foundation for evaluating the projects' software processes is this organizational measurement program. At this level, some limits are established in which the performance of the software product must be. Thus, through these set intervals, level four can be considered a predictable level. In addition, once a lower level of the set limits is reached, the organization will be able to act immediately to correct the problem so that the quality of the final product reaches a higher level. Figure 8 shows the key process areas of the fourth level of the CMM.

At the management and measurement level of TMM, meaning at the fourth level, the testing process is a measurable process. Criteria such as usability, reliability and maintainability are used to evaluate the software products. Testing at this level is a key element in the organization and takes place throughout the product development period. Even this level is not without key process areas, shown in the Fig. 9 [2].

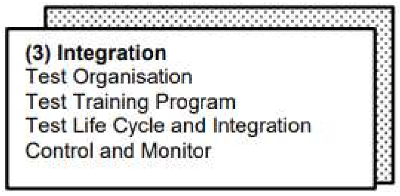

Fig. 7. Key process Level 3 – TMM [2].

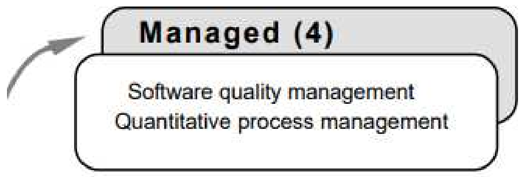

Fig. 8. Key process Level 4 – CMM [1].

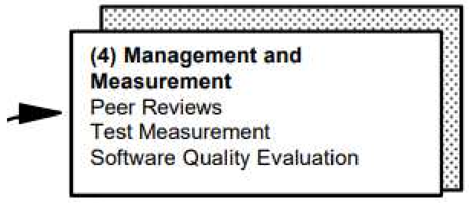

Fig. 9. Key process Level 4 – TMM [2].

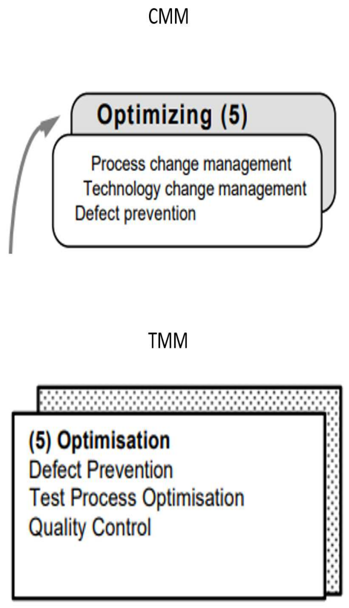

Fig. 10. Key process Level 5 - CMM&TMM [1, 2].

For both CMM and TMM the fifth level is the last level of maturity that an organization can reach. In the case of CMM, at this level the entire organization is focused on the continuous improvement of processes. The fifth level, the optimizing level, means that organizations are able to identify the weak points of their processes and act in such a way as to prevent the occurrence of defects. If defects occur, organizations with maturity level five may be able to determine the cause of the defects.

For TMM, the optimization level, means that testing is the process that aims to prevent defects. As in the case of CMM, a software organization that reaches the fifth maturity level of the testing process aims to improve continuously the testing process. The key process areas for both models can be seen in Fig. 10 [1, 2].

4. WEAKNESSES OF CMM AND TMM

The comparative analysis of those two models, Capability Maturity Model and Testability Maturity Model, carried out in the previous section rather described the strengths of these models and the similarities between them.
In addition to the strong points that they have, both CMM and TMM also present weak points that will be addressed in this chapter and on which some of the possible improvements will be presented.

The weak points that the CMM presents are related to the lack of addressing the problems that may arise from the point of view of strategic influences, human and cultural problems, the lack of establishing a procedure to consider the product's launch time on the market and also insufficient approach to the testing process to ensure product quality.

For all these problems, we will further describe some suggestions for improvement.

Regarding the appearance of strategic influences issues, in addition to the key process area regarding the employee-training program to train them in relation to the positions they hold, the CMM should mention as a key process area the implementation of a sustainable training for software project leaders to teach them to think and act strategically. It is important that in the development team there are leaders who know how to act strategically because this means that no matter how complex a situation is, they have the ability to make common sense of it. To think strategically means to understand the complex relationship between your organization and its environment. Using that knowledge, you can then make decisions that facilitate your organization's enduring success. In terms of strategic action, this means taking decisive action consistent with the strategic direction of your organization despite ambiguity, complexity, and chaos. Leaders who know how to think and to act strategically are constantly learning and acting.

To prevent human and cultural problems, first the factors that can cause these problems in a work environment must be understood. This is why the CMM could bring these type of issues to organizations and mention some factors that can bring problems such as religion, ethnicity, education and also generation. An organization can prevent these problems by ensuring effective communication so that the messages sent do not affect any of the employees in any way, and by providing all employees with the necessary tools to report and follow up any incident. To achieve the objectives and create a work environment as suitable as possible for employees, organizations must also take into account the development of team activities such as team building through which employees can learn how to respect their colleagues and understanding their environment.

The lack of establishing a procedure to consider the product's launch time on the market can be prevented by setting some objectives of the organization in achieving some limit dates by which the product or parts of the product should be delivered.

Regarding the insufficient approach to testing process to ensure product quality the application of the testability maturity model, which is the complement of the CMM, and which directly addresses the testing process, can be provided as a solution.

Even though it is the complement of capability maturity model, the TMM also has weak points such as the lack of addressing the establishment of a testing competence and even more important not taking into account the deployment process which can be considered part of the testing process.

Depending on the complexity of the developed software product, it is important for the members of the testing team to have almost the same level of knowledge and for this, it is necessary to implement a suitable training system. The TMM could add as key process areas the setting of a standard level of knowledge for team members by acquiring standard certifications.

When it comes to the deployment process, it should be considered as a small separate process. This process can be done either by a tester or by a developer. When it is done by the developer, there should be a procedure to follow so that all the changes that go live are first tested on that intermediate testing environment. If the tester is the one who is also responsible for the deployment, a procedure must be carried out that addresses this process separately. For example, in a software organization that works agile and whose deployment is the responsibility of the testers, they created a procedure to follow for this process with well-defined stages. Preparations for a deployment will begin by establishing the activities and the days on which they will be carried out. Figure 11 shows the main activities and the days they are carried out [5].

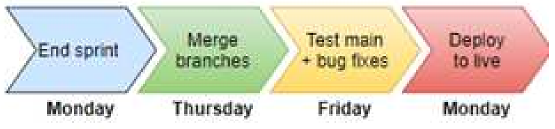

Fig. 11. Deploy process activities.

Branch management is another activity that must be taken into account in the implementation of the deployment if the development team works with branches. Working with branches means that the team diverges from the main line of development and continue to do work without interfering with that main line. At the beginning of the sprint the team decides which branches are needed based on the specifications and bugs that are in the sprint. Without delaying deploys the teams should try to find a balance with combining specifications on a branch where it is possible. Figure 12 shows the life cycle of a branch [5].

In order to complement the fact that the deployment process should be approached and treated as a separate process, in Fig. 13 a synthesis of all the activities that take place both during the testing and development process and whose objective is to complete the deployment. It is also important to take into account the number of servers with which the organization works and on which the deployment must be carried out and tested [6].

5. CONCLUSIONS

The main conclusion is that this study is very important as a reference for the organization in preparing software capability roadmap in software process and software testing.

The capability maturity model provides a conceptual structure for improving the management and development of software products in a disciplined and consistent way. It does not guarantee that software products will be successfully built or that all problems in software engineering will be solved.

At the same time, the testability maturity model is focused on the test process. It has been developed to support software organizations at evaluating and improving their test process.

The basic difference between these two models is that the CMM analyze the maturity of the software processes of an organization while the TMM describes the testing process and supports software quality monitoring but both work on process maturity levels.

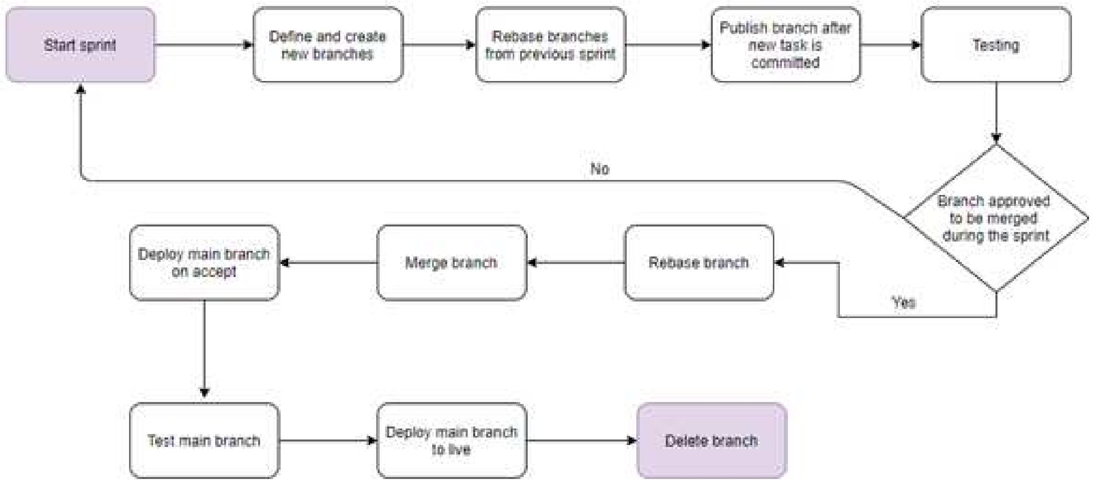
Start sprint
Define and create new branches
Rebase branches from previous sprint
Publish branch after new task is committed
Testing
No
Branch approved to be merged during the sprint
Yes
Rebase branch
Merge branch
Deploy main branch on accept
Test main branch
Deploy main branch to live
Delete branch

Fig. 12. Life cycle of a branch.

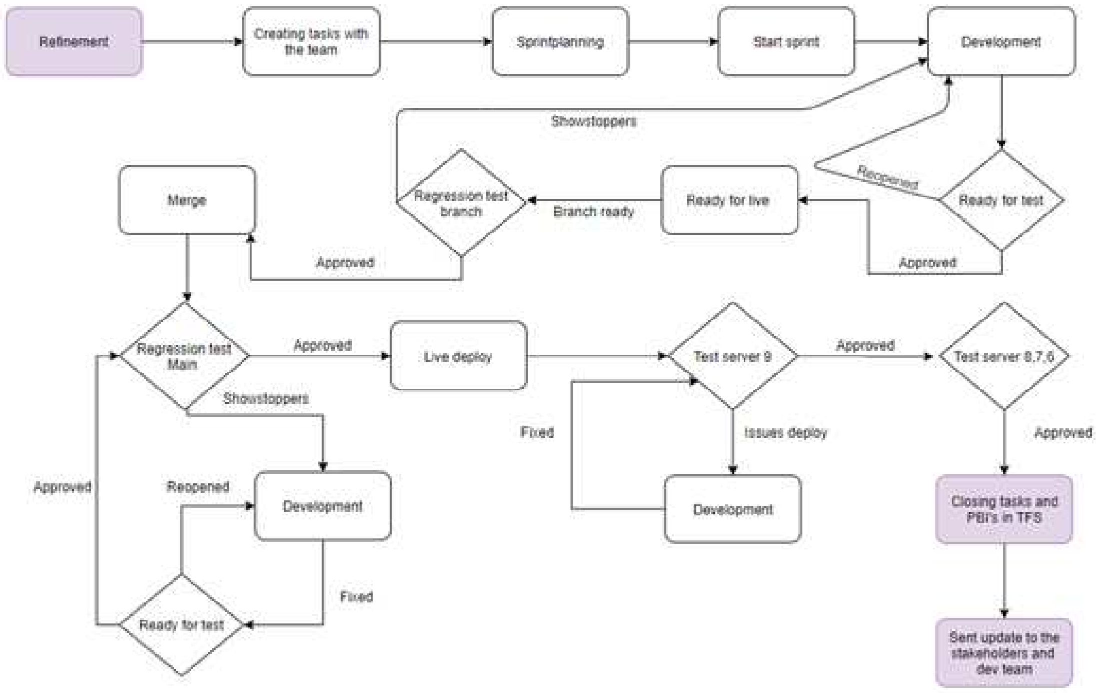

Fig. 13. Testing activities.

REFERENCES

[1] M.C. Paulk, C.V. Weber, S.M. Garcia, M.B. Chrissis, & M. Bush, *Key practices of the capability maturity model, version 1.1*, Carnegie-Mellon Univ. Pittsburgh Pa Software Engineering Inst., 1993.

[2] V. Veenendaal, R. Swinkels, *Guideline for testing maturity: Part* 1: *The TMM model*, Professional Tester, vol. 3, No. 1, 2002, pp.11–15.

[3] T. Ericson, A. Subotic, & S. Ursing, *TIM—a test improvement model*, Software Testing, Verification and Reliability, vol. 7, no. 4, 1997, pp. 229–246.

[4] Erik van Veenendaal, *Guidelines for Testing Maturity – Part2: Test Maturity Model level 2*, Professional Tester, vol. 3, no. 2, 2002, pp. 21–24.

[5] R. Black, Advanced Software Testing – *Vol. 2: Guide to the Istqb Advanced Certification as an Advanced Test Manager*, Rocky Nook, Inc., 2014.

[6] R. Marselis, B. Veenendaal, D. Geurts, W. Ruigrok, *Quality for DevOps*, Vianen Netherlands Sogeti, 2022.