A Framework for Improving Software Maintenance Using World Class Manufacturing Methodology

إطار عمل لتحسين صيانة البرامج باستخدام منهجية تصنيع عالمية المستوى

د. محمد علي سالم. أستاذ مساعد بقسم هندسة البرمجيات، كلية تقنية المعلومات، جامعة بنغازي
د. سلوى محمد العقيلي. أستاذ مساعد بقسم هندسة البرمجيات، كلية تقنية المعلومات، جامعة بنغازي
حنان رجب نجمة . طالبة دراسات عليا . كلية تقنية المعلومات. جامعة بنغازي

Dr: Muhammad . A. Salem. Assistant Professor, Department of Software Engineering, College of Information Technology, University of Benghazi
Email: Mohamed.hagal@uob.edu.ly.
Dr: Salwa . M. Al-Aqili. Assistant Professor, Department of Software Engineering, College of Information Technology, University of Benghazi
Email:salwa.elakeili@uob.edu.ly.
Hanan . R. najma. Postgraduate student. College of Information Technology. Benghazi University
Email: hanan.najma@uob.edu.ly.

تاريخ تسليم البحث 2022 / 3 / 22 تاريخ قبول البحث 2022 / 4 / 28

الملخص: أصبح تطوير البرمجيات أكثر تعقيدًا وضخامة، وبالتالي تواجه مشكلة كبيرة في صيانتها. خلال المراحل الأولية لتطوير البرمجيات أو أجزاء منها (المكونات البرمجية)، يصبح من الضروري تطويرها بالآلية التي يمكن بها سهولة تصحيح الأخطاء فيها أو زيادة أدائها أو تكيفها للعمل في بيئة متغيرة (أي بمعنى، سهولة صيانتها)، وهذا بدوره سيقلص تكلفة صيانتها لاحقا. وكما هو شائع، إن عملية تطوير البرمجيات تتم وفقا للمطلوب منها وبناء على الاحتياجات المحددة من الزبون ذو الشأن. ومن هنا، فإن تطوير تلك البرمجيات يجب أن يأخذ في الاعتبار أن المتطلبات قد تتغير بشكل متكرر. ووفقًا لبعض الاستطلاعات والدراسات، فإن عملية صيانة البرمجيات تمثل حوالي 80٪ من إجمالي تكاليف تطويرها. لذلك ، اقترحت هذه الورقة إطارًا لدمج تقنيات التصنيع على مستوى عالمي لإنشاء نموذج يمكن استخدامه لتحسين الصيانة وتقليل الوقت وتكلفة صيانة البرمجيات.

الكلمات المفتاحية: صيانة البرمجيات، منهجية تصنيع عالمية المستوى، نموذج معالجة البرمجية، تطوير البرمجيات

Abstract: The software that is being developed is becoming more complex and huge. The software industry has a huge problem with software maintenance. During the initial stages of software development, it becomes necessary to identify the ease with which a software system or component can be modified to correct faults, increase performance or adapt to a changing environment, i.e. software maintainability. This aids in the reduction of maintenance costs. A software is developed for some particular purpose, which may or may not change depending on the needs of the customer or the technological process. Hence, it leads to the evolution of the software to fit with the changing requirements. This non-trivial problem requires significant attention. According to some surveys, software maintenance accounts for about 80% of total software development costs. Therefore, this paper proposed a framework to integrate various World Class Manufacturing (WCM)-based technologies to create a model that can be utilized for improving maintenance, reduce time and cost of software maintenance.

Keywords: software maintenance, WCM, software process model, software development.

I. INTRODUCTION

Software development life cycle (SDLC) is one of the most important aspects of software engineering. It's because the structure of activities necessary to develop a software system has an impact on the product's quality [1, 2]. There are several approaches to software development[3]. A generic SDLC consists of five phases including requirements engineering, architecture, design, implementation, testing, software deployment, and maintenance. The maintenance phase follows the release of the product and keeps the software up to date with changes in the environment and changing user requirements. The initial stages should be completed so that the product can be easily maintained. The structure should be designed in such a way that it can be easily changed during the design

phase. Similarly, the implementation stage should produce code that is easy to read, comprehend, and modify. Maintenance can only be done effectively if the previous phases have been completed correctly[4].

1.1 **Software maintenance**

The IEEE 1219-1998 software standards document defines software maintenance as a "modification of a software product after delivery to improve performance, correct errors and different properties, or let the product adapt to a changing environment"[5]. The workload of the software maintenance process is enormous, and while maintenance costs vary significantly in different areas of application, on average, maintenance costs are relatively high. The cost of large-scale software maintenance is roughly four times that of development [6]. However, software maintenance has been identified as the most significant contribution to the software development life cycle (SDLC), and the time and costs associated with this phase of the SDLC are frequently underestimated[7].

Building software takes time, and this step of the software development process could take up to two years. The maintenance phase of the software development process can

last anywhere from one to ten years, depending on the type of program. This is when software is either stopped or replaced with new and

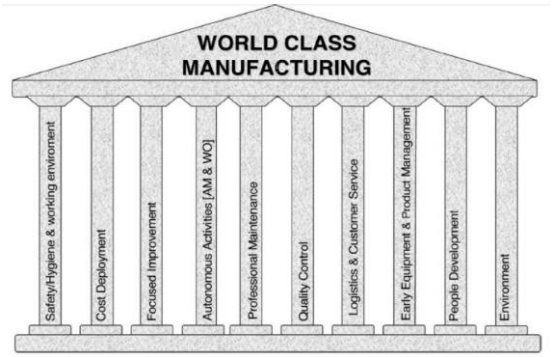

Fig. 1. pillars of WCM [14]

updated software.[8].

Over the last century, software maintenance tools have been developed in an attempt to improve the success rate of software systems[9]. The usage of a software maintenance tool can aid in the process of software maintenance. When all application, pass incident, and team information is kept in one place, for example, the time it takes to handle an issue is reduced and more effective[10].

World Class Manufacturing (WCM) was born with a joint venture of Fiat and the best European and Japanese experts with a goal to increase the production standards to recognized world standards. WCM is a structured and integrated manufacturing system that encompasses nearly all aspects of production, from safety to the environment, maintenance to logistics, and quality assurance. The main and most essential goal of this method is to include and encourage personnel who work in plants and firms to continuously improve production and gradually remove waste while ensuring quality and maximum flexibility in reacting to client needs. In the context of continuous improvement, WCM principles apply to all aspects of plant organization, from the quality system to maintain, cost control to logistics [11]. The purpose of this methodology is to eliminate all types of losses in the manufacturing process by following the “zero losses” policy: zero scraps, failures, and incidents. Its goal is to achieve zero waste.

The WCM is a continuous-improvement methodology. It is a robust strategy based on the economic consequences of quality, production, logistics, maintenance, and safety issues[12] . Fig 1 depicts the 10 WCM pillars and their descriptions.

Software development and software maintenance are distinctly different, concerning the nature of work and

execution of these activities. The work types, work sizes, workflow, required skills, and work stages are all different in Software Maintenance.

Software maintenance has become inefficient and ineffective due to this mismatch in execution models. As a result, it's vital to consider software maintenance's needs and develop models, processes, and practices that meet them[13]. Therefore, this research aims to develop a framework to improve the software engineering maintenance process.

The remainder of this paper is organized as follows. Section II gives problem statement. Section III gives aims of the study. Section IV explains motivation. Section V describes literature review. Section VI describes software maintenance models. Section VII illustrate the proposed framework. Finally, Section VI concludes the paper.

I. PROBLEM STATEMENT

Software development becomes increasingly complex and large over time. Hence, maintaining these programs remains a major challenge, due to the increasing development of their industry to keep pace with the developments of the environment in which they are located. Therefore, ease of modification of these software or software components, improving their performance or adapting them to a changing environment becomes essential [6].

Despite the fact that maintenance tasks are known to be complex and also costly, this phase has received far less attention than other phases of the software development lifecycle. However, many researchers are trying to develop models and tools to improve software maintenance, but the challenge of the appropriateness of these improvements and their acceptance by practitioners remains difficult in practice. In turn, practitioners are not always up to date with the proposals provided by the research community [15, 16]. This study investigates the gap between software maintenance techniques and tools proposed by the researcher community to improve the software maintenance process.

II. AIMS OF THE STUDY

Software maintenance is still an active area of research due to its importance in the software engineering field. Maintainable products save both time and money for customers and software engineering companies. Therefore, the aim of this research is to develop a framework to improve the software engineering maintenance process. To achieve that, a framework incorporating the strengths of both WCM methodology tools and scrum methodology would be established. To achieve the aim, a number of objectives have been defined.

1) Introduce a deep background and knowledge on software maintenance.
2) Establish a comparative literature review on software maintenance approaches and World Class Manufacturing (WCM).
3) Develop a framework for improving software maintenance activities by using World Class Manufacturing tools.

III. MOTIVATION

For many years, software maintenance has been one of the most contentious aspects of the software development process. It is known as the most expensive phase of the software development life cycle (SDLC) since it takes up the most time and money in the whole project [10].

A software project is delivered on time only if all parts of the software development process, including setup time, are completed within the estimated time frame. Various researchers have developed significant tools and strategies to improve software maintenance quality. At the same time, research is needed to improve software quality and lessen the obstacles of the maintenance phase[8].

Several execution approaches and tools, such as Agile, XP, and Scrum, have evolved over time to address these challenges. As a result, in this study we proposed a framework for improving software maintenance by using WCM methodology tools in this study.

IV. LITERATURE REVIEW

Software maintenance is a post-delivery activity whose major goal is to keep software's value over time. Once the software is given to the customer, any issues with it, from character enhancement to defect fixing, are addressed

as a maintenance job. It is a fact that the world is never static and perfect[17].

Lenarduzzi et al. [18] introduced a study that focuses on tools that aren't just research prototypes, but may also be employed in academic and industrial studies. Most of the tools utilized in industrial case studies were no longer available at the time of publication, preventing future academics or practitioners from quickly adopting the same methodologies and analyses utilizing already existing and tested tools. Furthermore, the majority of frequently used tools are commercial, and there are few open source communities capable of developing and supporting such products. This indicates that the research activities are not based on existing results and tools, but rather on the development of new ones.

This demonstrates the field's immaturity and the need for more research in this area to develop reusable models and tools that practitioners can use.

Murdoch et al. [19] proposed a generic method for software maintenance that uses structured event logs as a base that can be implemented on any software system. This method will aim to reduce the time spent on software maintenance, which can correlate to reducing the overall cost of a software system's life cycle. This method will ensure that a software system has enough information in its event log structure to be able to run analyses on the event logs for interpretation. Another advantage to the design of this method is that it will enable effective filtering and make the event logs available to software maintainers. However, this study also mentioned that the software maintenance phase is the largest contributor to costs and time involved in the software development life cycle. However, this approach is costly and consumes a huge amount of time to understand when dealing with a large number of maintenance problems.

Lacerda et al. [20] introduced a new tool called DR-Tools Suite, a suite of lightweight open-source tools that analyze and generate source code metrics and allow developers to examine the findings in various formats and graphs. They also establish a set of heuristics to help with code analysis. They conducted two case studies (one academic and the other one industrial) to gather comments on the tools suite, as well as insights into how we would modify the tools and develop new tools to assist developers in their everyday work to improve the software maintenance process. However, this tool concentrated on one part of the maintenance process is code analysis and intends to expand the DR-Tools Suite with new tools such as code smell detectors, refactoring recommenders, tools that support code review, and more.

The proposed maintenance model in [9] is concerned with how to maintain the software projects starting from the pretesting phases of the software project. The model will provide in its initial phase the maintenance method from the implementation phase, where this phase is one of the most important phases for software projects, developing and evaluating the requirements of the issues in various domains. These models have been proposed to determine the general or particular scopes of software products, and none of these quality models concerns the quality of software maintenance tools.

In addition to, quality models concern the quality of software maintenance tools. The proposed software quality maintenance models were developed based on the maintenance tools factor and the comparisons between the well-known quality models. These comparisons are the leakage of criteria based on distinct views and knowledge of maintenance tools requirements. The proposed technique applied to software maintenance tools technique demonstrates that the twelve factors must deem to increase the quality of software maintenance tools.

Stojanov [16] demonstrated the author's reflections on personal experience related to inquiring and improving software maintenance practice in a local micro software company. Since the selected software company spends the majority of its working time on maintenance activities, assessment and improvement of software maintenance processes which are essential for achieving better business performance and increased customer satisfaction. One of the identified improvement proposals was implemented as a technical solution in the company, which improved the time processing of maintenance requests. Process assessment and improvement facilitated the systematization of knowledge on maintenance practice, which is an additional benefit for the company. However, because this study focused on decreasing the processing time for repair requests in small companies, more research into large companies is needed.

Thomazinho et al. [15] presented a software maintenance approach that is used in small and medium-sized business (SMB) organizations in Brazil. The proposed approach indicates how SMB IT companies have improved their software maintenance processes. Multiple case studies were performed to validate this approach. The outcomes showed that strategies associated with managing users' knowledge and development/maintenance teams are relevant to increasing the maintenance process's effectiveness. This approach involves three aspects: users' knowledge management, maintenance team knowledge, and the management and maintenance process. This improvement includes reducing time and also minimizing the number of tickets. The response time for tickets resolution to the end-user has been reduced. In addition, IT organizations have minimized the effects associated with both staff and client turnovers. This approach also does not focus on decreasing the processing time for repair requests in large companies.

Masrat et al. [6] discussed some major problems that occur during software maintenance, particularly in terms of planning and comprehending architecture change, were software maintenance issues should be generated at different stages of the software life cycle and taken into account throughout the process. Software maintenance issues are costly, complex, and necessitate the services of skilled professionals. As a result, software maintenance must be considered at an early stage of the software development life cycle. This study examines various software approaches, models, and processes, as well as software after it has been delivered and is in the process of being retired between software maintenance activities. They also discuss the potential benefits of employing the suggested Model in software maintenance. These insights are useful for firms that need to know what software maintenance benefits to expect. It did, however, show that the maintenance phase is still a problem that needs to be addressed and that measures improve it are needed.

In summary, the previous studies show that software maintenance is still a challenge, and more study is needed to investigate more issues and develop a framework to address them.

V. SOFTWARE MAINTENANCE MODELS

A. *Co-Op Process Model*

According to [6], the author defined the Co-Op model as"an agile software maintenance approach centered on software maintenance and a supporting tool set that can be used in academic institutions". It encourages learning rather than focusing on the creation of a new system, allowing for the prioritization of early process phases. Fig 2 shows the CO - Op Process Model Stages.

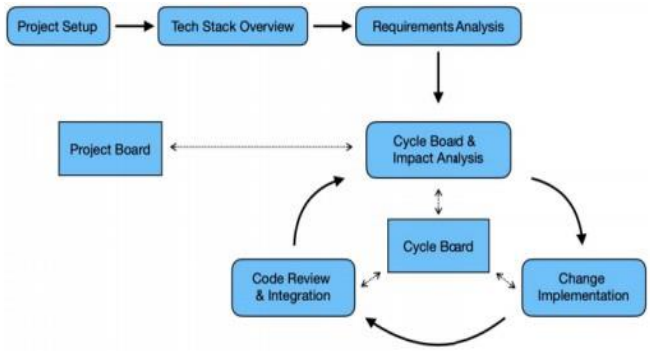

Fig. 2. Co-Op Process Model [6]

A. Scrum Model

Fig 3 illustrates scrum maintenance model. The model begins with the planning phase, which focuses on

controlling and tracking different types of maintenance changes. Priority will be given to corrective maintenance requests (critical work) as they are very important to the client's business. Non-corrective maintenance requests are scheduled. The current sprint is paused, work is saved in the version control system, and a new version is being produced for urgent requests [6]. After completing the urgent maintenance request(s), the model resumes the paused sprint status from where it was paused (last checkpoint). As a result, this model includes a concept that focuses on using version control to save the sprint state so that the critical work on the emergency sprint may be completed because it is more important to the client's business [21].

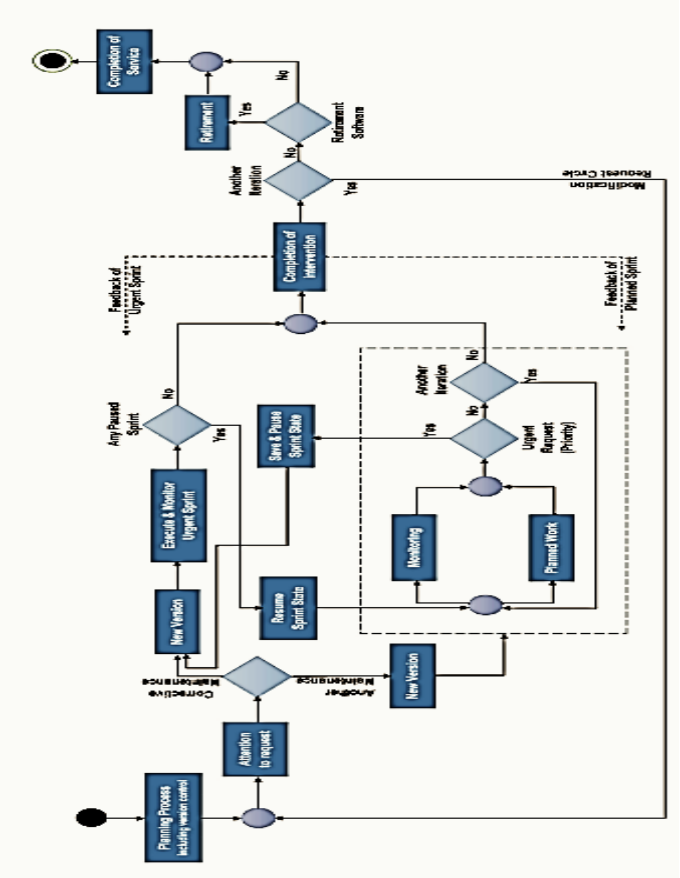

Fig. 3. Scrum Maintenance Model [6]

A. *Agile Maintenance Model*

The term "agile maintenance" refers to any agile methodology used in the maintenance process. Maintainers can respond to consumers quickly with agile maintenance, and it also aids in keeping track of customer requests. It also needs consumer involvement and interaction. As a result, agile approaches are utilized in software maintenance.

Adoption of this strategy has a number of drawbacks, including [6]:

- Sprint planning is disrupted by large amounts of user activity, resulting in delivery delays.
- A large number of developers cannot be added to the team, which has an impact on the performance of other teams.
- Unstable velocity due to group advancement has an impact on the next sprint's estimated velocity.

VI. THE PROPOSED FRAMEWORK

The purpose of suggesting this framework is to develop a plan for the software maintenance process after a sudden failure and reduce the time taken to correct the failure, as some of the tools in the WCM methodology were used to improve software maintenance.

The base of this framework is the integration and interaction of many WCM Methodology tools. Fig 4 depicts

how various tools are linked together.

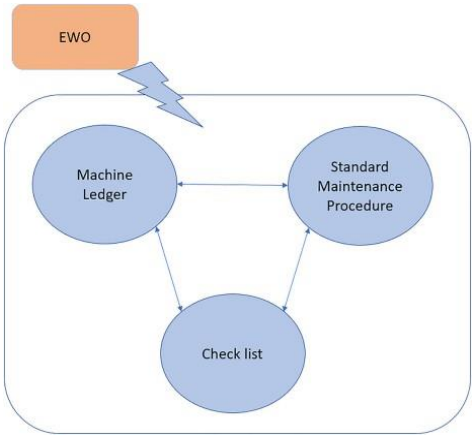

Fig. 4. Tools integration [22]

A. Framework components description

The first segment aims to describe the framework's main tools. Four tools has been proposed to improve software maintenance which are:

Machine ledger: It is a tool used to monitor and register all the maintenance activities in the time horizon taken as reference. It identifies and manages all the components where preventive maintenance should be implemented [22].

A machine ledger tool can be adopted in software maintenance and used as a reference to record and monitor all the activities that occurred during the maintenance phase while recording the time taken to carry out each activity. It also contains the time horizon to determine the maintenance operations carried out by the company during a specific period, for example, when a failure occurs during testing of the product as a whole the component was not tested as a unit testing before it is integrated with the all components, this failure is recorded and the time taken to correct the failure.

At the end of the time horizon, the Main Time Between Failures (MTBF) and the Main Time Between Repairs (MTBR) is calculated to be used in predicting the failures rate that we may encounter when building software. Then the moderators provide comments and suggestions for improvement and avoid such failures in the future that are made based on the information stored in the Machine Ledger. Where MTBF= total time of correct operation in a period/number of failures and MTTR=total hours of downtime caused by system failures/number of failures.

- **Standard maintenance procedure (SMP)**: A SMP structure, under a WCM perspective, is a detailed set of steps that outlines how to complete a maintenance task and serves as a documented standard by which the task should be completed. In this way, the same SMP can be applied to all projects that appear to have the same failures and require maintenance. All repetitive maintenance tasks should be covered by SMP, regardless of who performs them.
- For example, maintenance plans are developed so that each problem that may happen during the programming testing or after the product is delivered has the appropriate solution for it.
- ***Check-list:*** The check-list is generated any time a maintenance intervention is scheduled on the Machine Ledger of a particular Project or Product. Following the WCM technique, the structure of the check-list can be used as follows in software maintenance:
  The first section is the heading, which contains all of the basic information, such as the person in charge of the intervention, the type of maintenance, the date, and a brief description of the maintenance operations.

The operational part is in the second section, and it includes all of the fields that the maintainer must complete throughout the task's execution. In addition to the start and end dates of the work. The importance of this check-list is based on its interaction and interdependence with the machine ledger and SMP.

- ***Emergency work order(EWO):*** The EWO is divided into different sections. in software maintenance can be used when need to specify the nature of the failure of the product or project.

The most important aspect of this section is the row that contains the full time component required to complete the maintenance process. These periods include wait times to repair, problem-solving times, and startup times, to name a few. By combining all of these times, the Time To Repair (TTR) is calculated. Having all the times separated allows performing analyses and identifying areas of improvement in order to reduce the time to repair.

The second section involves an analysis of the core cause of the failure in order to prevent it from happening again. The five whys are a common approach for determining the fundamental cause of a failure. The failure is then investigated using the Five Whys tool as shown in Figure 5.

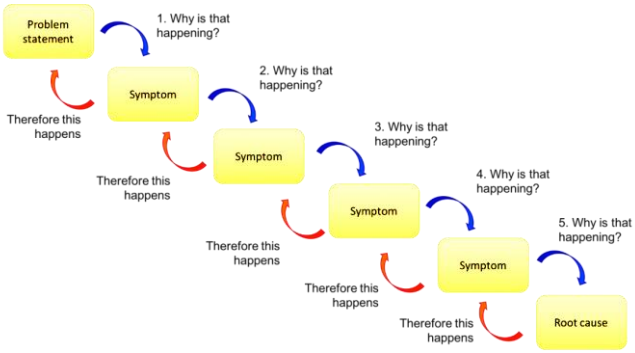

Figure 5. Five whys tool

A. Framework Implementation

- *Corrective maintenance*: When a failure occurs, the maintenance team is contacted to help. The procedure to follow in this case is suggested by the WCM methodology: at the end of the intervention, the maintainer is asked to fill out the EWO module in order to determine the root cause of the failure. Figure 6 depicts the corrective maintenance procedure. Every time a corrective maintenance intervention is performed, it must be reported on the product's own machine ledger. The data that must be filled in are the intervention time, the correspondent EWO code number, and the failure root cause that has been found. Both frequency and duration of failures have an impact on the creation of the new machine ledger at the end of the year. The data collected allows for the creation of new maintenance intervals and maintenance strategies for the products.

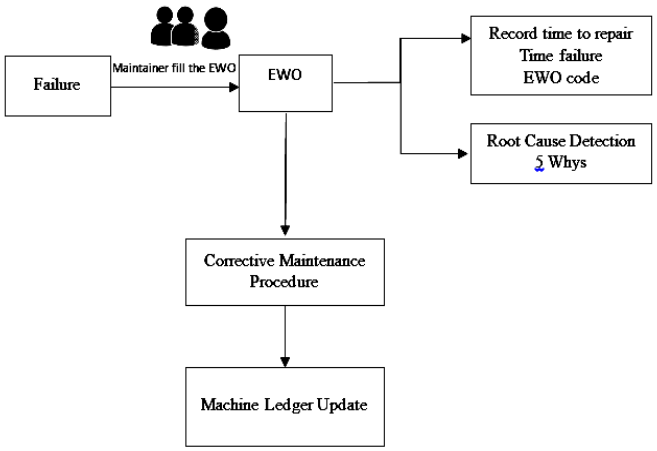

Fig. 6. Corrective maintenance process

- **Preventive maintenance (PM)**: PM is accomplished by a series of scheduled interventions. The PM process is depicted in Fig 7. Once the product is tested and there is a problem, it must be addressed by the maintainer's team. The checklists, as well as the work order and SMPs, must be brought to the intervention site by the team. If the team discovers problems when implementing the product, they must also carry out a corrective maintenance operation in accordance with the related flow.
  The team, on the other hand, must review the relevant SMPs, check the check-list activities, and record the values obtained when carrying out the checklist activities.
  Both the data from PM and corrective maintenance are registered on the machine ledger tool, in order to improve it.

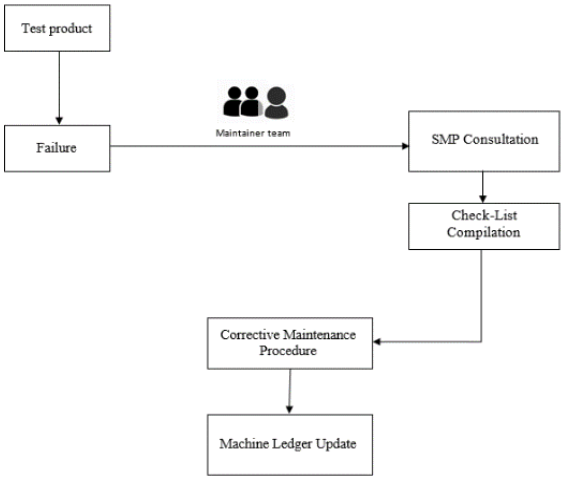

Fig. 7. Preventive maintenance process

VII. CONCLUSION

With the development of the IT industry, software products are being developed largely, and so the maintenance problemsthough non-trivial have become increasingly noticeable.

This paper discussed software maintenance procedures and models that have significant practical and theoretical implications for improving software maintenance processes, guiding maintenance operations, and improving software maintenance quality utilizing through the WCM Methodology. The proposed framework used to show how the WCM technique may beused to improve software maintenance processes. Also, the potential benefits of employing WCM in software maintenance is discussed. This framework can be useful for organizations that need to know what software maintenance benefits to expect. As a scope of future work, the proposed work can be applied in many software projects to improve them maintenance, and the framework will be refined through the recommended remarks on it.

REFERENCES

- I. Sommerville, *Software engineering*. Pearson Education India, 2004.
-R. Kneuper, "Sixty years of software development life cycle models," *IEEE Annals of the History of Computing*, vol. 39, no. 3, pp. 41-54, 2017.
-K. Saeedi and A. Visvizi, "Software Development Methodologies, HEIs, and the Digital Economy," *Education Sciences*, vol. 11, no. 2, p. 73, 2021. [Online]. Available: https://www.mdpi.com/2227-7102/11/2/73.
-K. Erdil, E. Finn, K. Keating, J. Meattle, S. Park, and D. Yoon, "Software maintenance as part of the software life cycle," *Comp180: Software Engineering Project*, vol. 1, pp. 1-49, 2003.
-I. S. C. Committee, "IEEE standard glossary of software engineering terminology (IEEE Std 610.12-1990). Los Alamitos," *CA: IEEE Computer Society*, vol. 169, p. 132, 1990.
-A. Masrat, "Software Maintenance Models and Processes: An Overview," *Available at SSRN 3838444*, 2021.
-T. Mens, A. Serebrenik, and A. Cleve, *Evolving Software Systems*. Springer, 2014.
-A. Gupta and S. Sharma, "Software maintenance: Challenges and issues," *Issues*, vol. 1, no. 1, pp. 23-25, 2015.

-H. Fawareh, "Software quality model for maintenance software purposes," *International Journal of Engineering Research and Technology, 13 (1),* pp. 158-162, 2020.
-S. Abdullah, M. Subramaniam, and S. Anuar, "Improving the Governance of Software Maintenance Process for Agile Software Development Team," *International Journal of Engineering and Technology*, vol. 7, no. 4.31, 2018.
- Q. Riasat, "World Class Manufacturing and Its applications (Lean Manufacturing)," Politecnico di Torino, 2020.
- M. Rossini, F. Audino, F. Costa, F. D. Cifone, K. Kundu, and A. Portioli-Staudacher, "Extending lean frontiers: A kaizen case study in an Italian MTO manufacturing company," *The International Journal of Advanced Manufacturing Technology*, vol. 104, no. 5, pp. 1869-1888, 2019.
-N. Rao, "Scrumban Software Maintenance," ed: CreateSpace Independent Publishing Platform, 2017.
-E. B. SARI, "World Class Manufacturing (WCM) Model and Operational Performance Indicators: Comparison Between WCM Firms," *Dokuz Eylül Üniversitesi İşletme Fakültesi Dergisi,* vol. 19, no. 2, pp. 249-269, 2018.
- A. L'Erario, H. C. S. Thomazinho, and J. A. Fabri, "An approach to software maintenance: A case study in small and medium-sized businesses it organizations," *International Journal of Software Engineering and Knowledge Engineering,* vol. 30, no. 05, pp. 603-630, 2020.
-Z. Stojanov, "Software maintenance improvement in small software companies: Reflections on experiences," 2021.
-S. Christa, V. Madhusudhan, V. Suma, and J. J. Rao, "Software maintenance: From the perspective of effort and cost requirement," in *Proceedings of the International Conference on Data Engineering and Communication Technology*, 2017: Springer, pp. 759-768.
- V. Lenarduzzi, A. Sillitti, and D. Taibi, "A survey on code analysis tools for software maintenance prediction," in *International Conference in Software Engineering for Defence Applications*, 2018: Springer, pp. 165-175.
- R. Murdoch, "Development of a method for software maintenance using event logs," North-West University (South Africa), 2020.
-G. Lacerda, F. Petrillo, and M. S. Pimenta, "DR-Tools: a suite of lightweight open-source tools to measure and visualize Java source code," in *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 2020: IEEE, pp. 802-805.
-F. ur Rehman, B. Maqbool, M. Q. Riaz, U. Qamar, and M. Abbas, "Scrum software maintenance model: Efficient software maintenance in agile methodology," in *2018 21st Saudi Computer Society National Computer Conference (NCC)*, 2018: IEEE, pp. 1-5.
-K. Kundu, F. Cifone, F. Costa, A. Portioli-Staudacher, and M. Rossini, "An evaluation of preventive maintenance framework in an Italian manufacturing company," *Journal of Quality in Maintenance Engineering*, 2020.