International Journal of Technoinformatics Engineering
Vol: 01; **Issue**: 01; Jul-Dec 2024; **Pg**: 020-025

Emerging Technologies and Cost Estimation Models for Software Maintenance: A Comprehensive Analysis

Ahamed Masih Uddin Siddiqi
Department of Computer Science Engineering, Mangalayatan University (MU), Aligarh, Uttar Pradesh, India

Received: 21th August 2024 / Accepted: 09th October 2024 / Published: 06th November 2024

Abstract:

The concept of software maintenance encompasses a wide range of activities necessary for the effective operation and upkeep of software, with a focus on cost performance. In the system life cycle, the maintenance stage is when a software product performs all constructive activities, including corrective maintenance like fixes and bug fixes, adaptive maintenance like adjustments to new environments or requirements, perfective maintenance like enhancements and optimizations, and preventive maintenance like measures to anticipate problems. This broad scope assures that the software is effective, efficient, and meets changing user and technical needs. The primary objective of this study is to conduct research on the emerging technologies and cost estimation models pertaining to software maintenance. The present study will employ a qualitative research methodology. The study's findings indicate that the use of developing technologies has a substantial influence on the cost dynamics of software maintenance, hence requiring the development of more precise and flexible estimating methods. The proposed model adeptly tackles these issues by integrating pertinent elements such as technological advancement, intricacy, and allocation of resources. The validation results illustrated the theoretical capacity of the model to offer enhanced accuracy in cost projections, hence facilitating improved budgetary planning and resource allocation.
Keywords: *Cost Estimation Model; Software Maintenance; Software Development; Lifecycle; Technology.*

INTRODUCTION

In recent years, software has emerged as the most financially burdensome element of computer computing initiatives. In the realm of software development, the primary source of costs is attributed to human labour, with the bulk of estimates methodologies expressing maintenance costs in terms of Person-Months [1]. "Person-Months" is used in project management and cost estimating to measure one person's monthly effort. It is used to estimate and allocate software development resources, particularly human labor, the main cost driver. A "Person-Month" estimates project time and manpower by capturing the effort needed for development, testing, debugging, and maintenance.

The generation of precise software cost estimates holds significant importance for both developers and clients. They serve a valuable purpose in the development of proposals, contract negotiations, scheduling, monitoring, and control [2]. The assessment of software costs is a complex task that is influenced by a range of elements, including human, technological, environmental, and political concerns. These factors collectively affect the ultimate cost and the resources required for optimal maintenance [3,4]. Certain elements exhibit more visibility than others. In order to effectively estimate software maintenance expenses, it is necessary to identify and assign weights to various factors. When referring to the full collection of actions that are necessary to run and maintain software in a manner that is efficient with regard to costs, the phrase "software maintenance" is used. "Software maintenance" is a term that explains the actions that take place following the distribution of software artefacts to a consumer [5]. The figure below illustrates the predominant cost estimation process in detail.

Numerous estimating models have been proposed and implemented throughout the years. A comprehensive examination of prominent estimating models that have been developed The primary objective of this study is to undertake a research investigation on emerging technologies and cost estimation models pertaining to software maintenance.

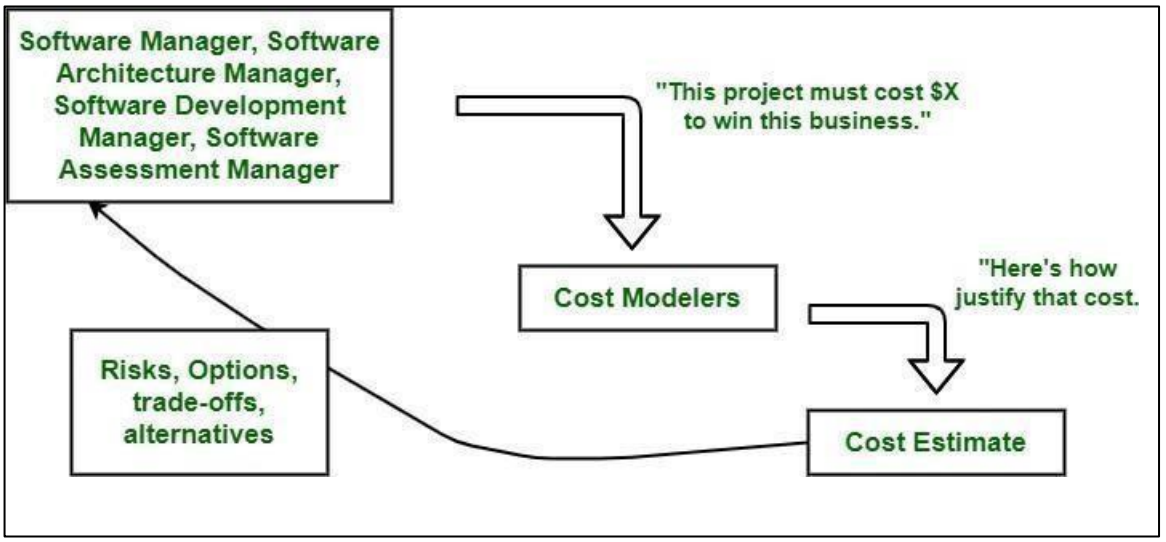

Figure 1. An Overview of Cost Estimation Process.

The present study aims to tackle these difficulties through the creation and implementation of a novel software framework with the objective of improving HCI. This framework will be based on a thorough examination of current HCI designs, finding significant areas where current solutions are inadequate and suggesting novel techniques to address these shortcomings. The study aims to investigate multiple dimensions of HCI, encompassing usability, accessibility, user engagement, and adaptation, with the objective of developing a framework that possesses both resilience and versatility. Through the utilization of advanced technologies such as machine learning, natural language processing, and gesture recognition, the proposed framework endeavors to provide a user experience that is tailored and adaptable, to accommodate the unique requirements and preferences of individual users. Furthermore, the study will integrate perspectives from scholarly literature and industry methodologies, guaranteeing that the framework is not only theoretically robust but also practically relevant across diverse fields. The primary objective of this study is to provide a valuable contribution to the field of HCI by introducing an innovative framework that enhances the quality of interactions, increases user happiness, and lays the foundation for future advancements in this area.

LITERATURE REVIEW

The subsequent part provides an in-depth examination of previous scholarly works pertaining to the field of emerging technologies and cost estimation models for software maintenance.

Table 1. Related Works.

Authors and Years	Methodology	Findings
Jimenez et al., (2020)	This comprehensive survey reviewed diagnostics and prognostics trends, focussing on multi-model approaches, obstacles, and research possibilities.	Single-model approaches cannot handle all complicated system diagnostics and prognostics duties, hence consulted research generally suggested additional models to solve model weaknesses.
van Dinter et al., (2022)	A systematic literature review (SLR) utilising an active learning technique analyses 42 primary publications on predictive maintenance using Digital Twins.	This studyis the first Digital Twin SLR in predictive maintenance. This cited study answered critical issues for creating a successful Digital Twin-based predictive maintenance model. It concluded that computational burden, data diversity, and

		model, asset, or component complexity remain the biggest hurdles in model design.
Almogahed et al., (2023)	Developed a unique framework for classifying refactoring techniques based on their measurable influence on internal quality criteria.	This suggested framework is a helpful guide for developers since it consists of three main components: a methodology for applying refactoring techniques, the Quality Model for Object-Oriented Design (QMOOD), and a classification scheme for refactoring approaches.
Almashhadani et al., (2023)	In this study, it analysed the issues that arise when attempting to assess the influence that agile methodologies have in terms of the quality aspects that are involved.	findings revealed that there are challenges that are onlysomewhat effective in terms of manageability, scalability, communication, collaboration, and transparency.
Heričko & Šumak (2023)	Comprehensive literature assessment of supervised-learning-based models for the classification of commits into maintenance activities.	Software needs adjustments to survive. Fixing faults, security vulnerabilities, meeting new functional needs, and improving software performance are examples.

Research Gap

Existing scholarly literature extensively examines several facets of software maintenance; yet, a conspicuous deficiency exists in the incorporation of developing technologies into cost estimating models. The aforementioned gap underscores the necessity for further investigation in order to construct and authenticate models that effectively capture the dynamic intricacies brought about by these technologies in the realm of software maintenance management.

METHODOLOGY

This study utilizes a qualitative research technique, specifically employing secondary data gathering methods, to investigate the convergence of developing technologies and cost assessment models in the context of software maintenance. A thorough examination of the existing body of literature was undertaken, encompassing peer-reviewed publications, industry reports, and case studies that were published between the timeframe of 2018 to 2024. In order to detect trends, patterns, and gaps in current maintenance management practices and cost estimation models driven by technological improvements, the collected data is subjected to a systematic evaluation and classification process based on thematic significance.

RESULTS AND DISUSSION

The extensive utilisation of data processing resources for software maintenance has resulted in a limitation on the speed of new software development. One potential resolution to this issue has been the adoption of fourth generation programming languages, which facilitate increased efficiency in software development compared to alternative approaches [4]. As a result of this change, there has been a rise in the quantity of software that need maintenance. Certain organisations that possess expertise in fourth-generation programming languages have deemed it financially prudent to contemplate the process of system rewriting as opposed to the ongoing maintenance and patching of their current software [1,11]. In the realm of software maintenance, the adoption of fourth generation languages might yield several effects.

- One potential solution to mitigate simple hidden faults is the implementation of a fourth generation language, which possesses the capability to automatically handle specific system features. For instance, this language can accurately identify the first and last entries within the system.
- Numerous languages belonging to the fourth generation exhibit seamless integration with data management systems that incorporate inherent data dictionaries. In programming, it is imperative that the data is accurately represented and that variables are declared correctly.
- Numerous languages belonging to the fourth generation exhibit self-documentation. The lack of comprehensive documentation is a probable factor contributing to challenges in maintaining third generation languages.
- Fourth-generation programming languages enhance the comprehensibility of a program, hence facilitating its maintenance by third-party individuals.
- Numerous fourth generation programming languages prohibit the utilization of ill-structured program constructs, hence potentially hindering future functionality.

The swift advancement of nascent technologies has profoundly transformed the domain of software maintenance, hence requiring the creation of increasingly intricate cost assessment models. Conventional models, which predominantly emphasise fixed variables such as code complexity, size, and historical data, frequently fail to adequately consider the dynamic characteristics of contemporary software environments. The advent of emerging technologies such as Artificial Intelligence (AI), Machine Learning (ML), Internet of Things (IoT), and DevOps practices has brought about novel factors and intricacies in the realm of maintenance. These include the requirement for timely updates, ongoing integration and deployment, and heightened security protocols. These technological improvements necessitate a paradigm shift in the estimation of maintenance costs, wherein models are re-evaluated to accommodate the dynamic nature of these components and offer enhanced precision and dependability in projections.

Integration of emerging technologies into established cost estimation frameworks poses a significant problem in the dynamic landscape of this expanding field. For several decades, conventional models such as COCOMO (Constructive Cost Model) and Function Point Analysis (FPA) have been extensively employed [12]. However, these models are encountering growing limitations in effectively addressing the complexities brought about by contemporary technologies. The aforementioned models frequently neglect to take into account several elements such as the iterative nature of development driven by artificial intelligence, the real-time data processing requirements of Internet of Things (IoT) systems, and the collaborative workflows facilitated by DevOps. Consequently, there is an increasing demand for novel or modified cost estimating models that may integrate these components, thereby offering a comprehensive perspective on software maintenance expenses within the framework of evolving technologies.

Table 2. Identified Gaps in Current Literatures [1,4,10,11,12].

Research Area	Identified Gap	Suggested Future Research
AI/ML Integration	Constrained models that integrate AI-driven development processes	Creating flexible pricing models for artificial intelligence and machine learning settings
IoT System Maintenance	Insufficient attention to real-time data processing criteria	Development of models tailored to meet the needs of maintenance in the Internet of Things (IoT)
DevOps Impact on Cost	Lack of adequate use of DevOps methodologies in the estimation process	Evaluation of cost estimation approaches related to DevOps
Security Costs in Emerging Tech	Lack of adequate depiction of cybersecurity expenses in estimation	The use of sophisticated security protocols inside cost models
Agile and Hybrid Methodologies	Limited emphasis on difficulties related to estimating costs particular to agile methodologies.	Development of cost estimation frameworks that are compatible with agile methodologies

Proposed Software Maintenance Cost Estimation Model

COCOMO (Constructive Cost approach) is a foundational approach for software project cost estimation. Barry W. Boehm created and published this model in 1981 using data from 63 projects [12]. A study utilised COCOMO II to present a systematic approach for estimating software maintenance costs in a fourth generation language environment. This model considers three parameters: SMCE with Fourth Generation Language Environment, ACT (Annual Change Traffic), and technical and non-technical factors affecting maintenance costs. Our proposed model is shown in figure 2 below. The figure shows the ACEM (Adaptive Cost Estimation Model) in software maintenance, which integrates project features, technical and non-technical factor weights, and a 4GL environment to refine cost estimates using model adjustments and History Table data. This adaptive model estimates ACT and maintenance costs continuously to adapt to project details and environmental conditions.

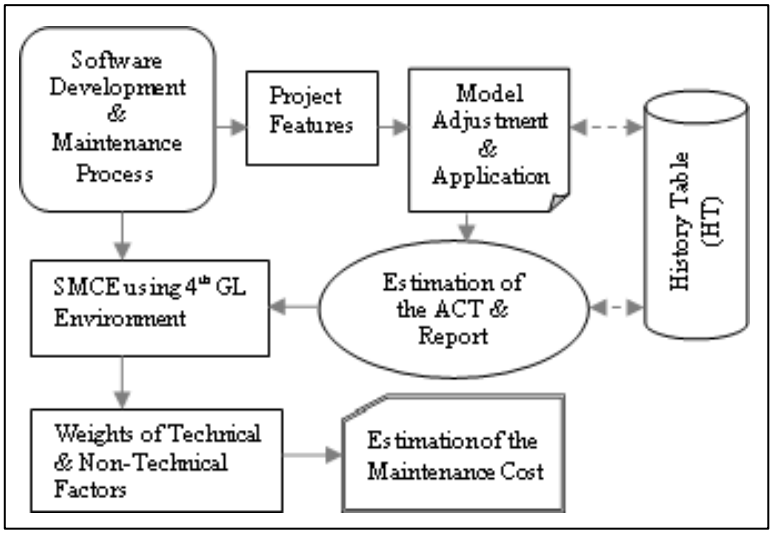

Figure 2. Proposed Model.

In addition, the Adaptive Cost Estimation approach (ACEM) is identified as the most efficacious approach for evaluating software maintenance costs within the framework of evolving technologies. ACEM is specifically engineered to effectively integrate the intricacies brought about by artificial intelligence (AI), the Internet of Things (IoT), DevOps, and improved cybersecurity tools, hence providing a versatile and adaptable methodology for cost estimating [4]. The proposed methodology utilises real-time data inputs and continuous feedback loops to adapt estimates in response to changing project conditions, hence maintaining the accuracy of cost projections during the software's entire lifecycle. The ACEM framework has modular evaluation tools that calculate the influence of certain technologies on maintenance tasks, including factors such as code complexity, update frequency, and security requirements. This enables a customised estimation procedure that takes into consideration the distinct demands of each technology. Through its emphasis on adaptability and precision, ACEM offers a holistic solution that improves decision-making, optimises resource allocation, and facilitates the sustainable administration of software systems in dynamic technological landscapes [4].

By integrating the insights derived from existing literature, this work presents a modified cost assessment model that effectively incorporates the distinctive attributes of new technologies. This model places significant emphasis on the qualities of flexibility, adaptability, and the capacity to accommodate the dynamic nature of contemporary software maintenance settings. The validation of the model through case studies or real-world application is crucial in order to ascertain its practical significance and efficacy in properly projecting costs. This, in turn, enables software maintenance professionals to optimize resource allocation, budgeting, and strategic planning.

The findings of this study hold substantial ramifications for both the academic and industrial sectors. The highlighted gaps in the academic literature underscore the need for more research in the development and validation of novel models that more accurately capture the challenges associated with software maintenance in an era characterised by the prevalence of AI, the IoT, and other emerging technologies. The suggested model presents a valuable tool for industry professionals, as it has the potential to refine decision-making processes, decrease maintenance expenses, and ultimately boost the sustainability and efficiency of software systems within a fiercely competitive technological environment. The study's

extensive scope, coupled with its emphasis on contemporary trends and issues in software maintenance, renders it a significant asset for industry stakeholders, encompassing scholars, practitioners, and policymakers alike.

CONCLUSION

In conclusion, this research highlights the imperative requirement for the development of software maintenance cost estimation models that can effectively incorporate upcoming technologies such as AI, IoT, and DevOps. The issues at hand are effectively tackled by the suggested COCOMO and ACEM methodologies, which incorporate real-time data, modular assessments, and continuous feedback. These mechanisms are designed to guarantee precise and flexible cost forecasts over the whole software lifespan. This technique not only improves the process of decision-making and allocation of resources, but also establishes a sustainable framework for effectively managing software maintenance in a progressively intricate technical environment. As a result, it offers substantial benefits to stakeholders in both academic and industrial sectors.

REFERENCES

1. Mohan M, Greer D. A survey of search-based refactoring for software maintenance. J Softw Eng Res Dev [Internet]. 2018;6(1). Available from: http://dx.doi.org/10.1186/s40411-018-0046-4
2. Nguyen-Duc A. The impact of software complexity on cost and quality-A comparative analysis between Open source and proprietary software. 2017.
3. Ali SS, Zafar MS, Saeed MT. Effort estimation problems in software maintenance-a survey. In: 2020 3rd international conference on computing, mathematics and engineering technologies (iCoMET). IEEE; 2020. p. 1–9.
4. Rush C, Roy R. Analysis of cost estimating processes used within a concurrent engineering environment throughout a product life cycle. In: Advances in Concurrent Engineering. CRC Press; 2023. p. 58–67.
5. Lenarduzzi V, Sillitti A, Taibi D. Analyzing forty years of software maintenance models. In: 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C). IEEE; 2017.
6. Jimenez JJM, Schwartz S, Vingerhoeds R, Grabot B, Salaün M. Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics. Journal of manufacturing systems. 2020;56:539–57.
7. Van Dinter R, Tekinerdogan B, Catal C. Predictive maintenance using digital twins: A systematic literature review. Information and Software Technology. 2022.
8. Almogahed A, Mahdin H, Omar M, Zakaria NH, Mostafa SA, AlQahtani SA, et al. A Refactoring Classification Framework for Efficient Software Maintenance. IEEE Access [Internet]. 2023;11:78904–17. Available from: http://dx.doi.org/10.1109/access.2023.3298678
9. Almashhadani M. Challenges in Agile Software Maintenance for Local and Global Development: An Empirical Assessment. An Empirical Assessment Information. 2023;14(5).
10. Heričko T, Šumak B. Commit classification into software maintenance activities: A systematic literature review. In: 2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC). IEEE; 2023.
11. Al-Saqqa S, Sawalha S, Abdelnabi H. Agile software development: Methodologies and trends. International Journal of Interactive Mobile Technologies. 2020;(11).
12. Ahmed M, Ibrahim NB, Nisar W, Ahmed A, Junaid M, Soriano Flores E, et al. A hybrid model for improving software cost estimation in global software development. Computers, Materials & Continua. 2024;78(1):1399–422.