Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 3, Issue 12, 2025

EFFECT OF TECHNICAL DEBT ON SOFTWARE QUALITY: MEDIATING ROLE OF CODE MAINTAINABILITY AND MODERATING ROLE OF PROJECT SIZE

Laviza Asif Memon[*1], **Anees Muhammad**[2]

[*1]*Lecturer, Computer Science Department, Shaheed Zulfiqar Ali Bhutto Institute of Science and Technology (SZABIST) University, Hyderabad Campus*
[2]*Lecturer, University of Sufism and Modern Sciences USMS, Bhitshah*

[*1]laviza.asif@hyd.szabist.edu.pk, [2]engr.aneesjamali@gmail.com

DOI: https://doi.org/10.5281/zenodo.18151040

Keywords
Technical Debt, Software Quality, Code Maintainability, Project Size, PLS-SEM, Software Development

Article History
Received: 30 October 2025
Accepted: 18 December 2025
Published: 31 December 2025

Corresponding Author: *
Laviza Asif Memon

Abstract
This research is on the effects of technical debt on the quality of the software and the mediating variable of code maintainability and moderating variable of project size in the software development projects in Hyderabad and Jamshoro. The survey design selected was quantitative, cross-sectional survey design, on which 200 software professionals gathered data on developers, engineers, and project managers. Technical debt, code maintainability, software quality, and project size measurement scales were borrowed in previous validated studies and implemented on PLS-SEM. The findings indicate that the direct and indirect impact of technical debt on the quality of software is negative, which proves the mediating role. Furthermore, project size also plays a major moderating role with bigger projects having more adverse negative effect. The analysis of reliability, convergent, and discriminant validity shows that the measurement model is strong. These results indicate that systematic technical debt management, code refactoring and maintainability-oriented practices are essential in improving software quality. The study makes a theoretical contribution, by incorporating mediation and moderation processes within the software engineering research and also offers practical advice to software companies when trying to deal with the technical debt in an appropriate manner.

INTRODUCTION

Technical debt has become a major issue in modern software development, especially in settings where there are high delivery pressures, agile development, and large systems. It describes the history of poor design and implementation choices, which can improve the short term development process, but raise the cost and risk of long term maintenance (Saeeda et al., 2024; Borowa et al., 2025). The recent research emphasizes that technical debt is not confined to the level of a code any longer but to the architectural, organizational, and social aspects of software projects (Ahmad et al., 2025; Calixto et al., 2024). According to empirical findings, uncontrolled technical debt may cause a drastic reduction in the quality of software, rise in the rate of defects, and adversely impact business performance (Tornhill and Borg, 2022; Licorish, 2025). The need to comprehend how technical debt affects the quality of software has evolved into a significant research agenda as the software system is increasingly complex and large.

Code maintainability has been identified as one of the major mechanisms that connect technical debt and the quality of software, as it is indicative of the relative ease with which software can be comprehended, modified, and expanded. Previous studies have suggested the presence of high amounts of technical debt as a common cause of low maintainability, which causes code complexity, the emergence of code smells, and the deterioration of modular structures (Ferdoshi et al., 2024; Borowa et al., 2025). Less maintainability, on its part, negatively impacts the key quality component of core quality maintenance, including reliability, performance, and defect prevention (Garomssa et al., 2022; Licorish, 2025). Research on the quality mediation processes shows that intermediate technical aspects are significant in converting development activities to end product results (Garomssa et al., 2022; Tornhill and Borg, 2022). Nevertheless, with the increase in the empirical literature related to technical debt and software quality, there is little focus on the investigation of the code maintainability as a mediating variable, especially within the organizational and industrial sphere.

Besides, the influence of the technical debt on the quality of the software is not universal and can be different in the context of various factors including the project size. Technical debt can negatively impact the project because larger projects are associated with larger teams, larger codebases, longer development cycles, and increased coordination requirements, which exacerbate the adverse effects of technical debt (Bokhari et al., 2025; Bashir, 2025). According to the previous research on software engineering and project management, it can be argued that project structural features tend to moderate the correlation between technical practices and performance results (Licorish, 2025; Mansour et al., 2022). Software companies in the context of developing countries, such as Pakistan, often work on tight budgets and have limited deadlines, so the size of the project is particularly an important moderating variable (Bokhari et al., 2025; Noor et al., 2025). Although its practical significance, there is low empirical evidence in combining technical debt, code maintainability, software quality, and project size in a single moderated mediation framework, thus, indicating a gap in the research and that is what this study tries to fill.

Aim of the Study

The objective of the given study is to investigate the impact of technical debt on the quality of software, as well as to check the mediating role of the code maintainability and the moderating role of the project size in software development projects implemented in Hyderabad and Jamshoro cities.

Objectives of the Study

1. To investigate the effect of technical debt on software quality in software development project.
2. To determine the correlation between technical debt and code maintainability in the software development settings.
3. To examine the impact of code maintainability on the quality of software in software projects.
4. To examine the mediation process between technical debt and software quality with the help of code maintainability.
5. To investigate the moderating impact of project size on the association amidst technical debt and quality of software.

Literature Review

The current literature reveals a strong suggestion that technical debt negatively affects the results of software development especially the quality of software. Technical debt occurs when the team of developers focuses on addressing short-term requirements, rather than the long-term maintainability of the code, creating architectural flaws and code smells, and limiting system resilience (Saeeda et al., 2024; Borowa et al., 2025). It has been shown through empirical studies that too much technical debt will increase defects density, slow the development velocity, and increase maintenance costs, which will ultimately compromise the quality of a product and business outcome (Tornhill and Borg, 2022;

Licorish, 2025). Moreover, recent studies have broadened the definition of the debt concept to cover the organizational and social dimensions, focusing on its macro effects on software projects (Ahmad et al., 2025; Calixto et al., 2024). All the above findings point towards the critical risk factor of high-quality software systems maintenance as technical debt.

Meanwhile, researchers are paying more attention to mediating technical processes and situational project features in their explanations of quality results. Maintainability of codes has been reported as one of the fundamental quality properties that have a direct impact on software reliability, adaptability, and sustainability (Ferdoshi et al., 2024; Garomssa et al., 2022). Research indicates that the maintainability is a mediator between development practices and software performance because it enhances complexity and decreases developer productivity (Licorish, 2025; Tornhill and Borg, 2022). Moreover, technical issues are converted into performance results depending on project-related aspects including size, complexity, and team set-up (Bokhari et al., 2025; Bashir, 2025). Nevertheless, there is a paucity of empirical studies which combine technical debt, code maintainability, and project size into a single analytical paradigm especially in new software markets and thus there is a distinct gap that the current research fills.

Hypotheses Development

Technical Debt and Software Quality

Technical debt has been well known as a significant source of degrading software quality since it presents a hidden cost that is gradually increasing throughout the system, causing instability in the system. Studies indicate that technical debt is an indicator of defect susceptibility, lower system robustness, as well as, adverse performance and user experiences (Tornhill and Borg, 2022; Saeeda et al., 2024). Uncontrolled technical debt in large-scale, agile systems has been found to diminish the qualities of products by promoting short-term solutions to the cost of sound design (Borowa et al., 2025; Licorish, 2025). On this basis, it is anticipated that the technical debt will directly impact negatively on software quality.

H1: Technical debt has a significant negative effect on software quality.

Mediating Role of Code Maintainability

Code maintainability is significant in converting technical choices into quality long term results. The high technical debt can lead to the creation of complex, inadequately structured code which can be hard to comprehend, edit and test, decreasing the maintainability (Ferdoshi et al., 2024; Calixto et al., 2024). A decrease in maintainability, in its turn, results in an increase in defect rates, length of development, and software quality (Garomssa et al., 2022; Licorish, 2025). The preceding mediation research points to the fact that technical properties tend to serve as an engine that development practices affect the end of performance results (Garomssa et al., 2022). Based on this, this paper hypothesizes that there is an intervening variable, the maintainability of code, that means the relationship between technical debt and software quality.

H2: Code maintainability mediates the relationship between technical debt and software quality.

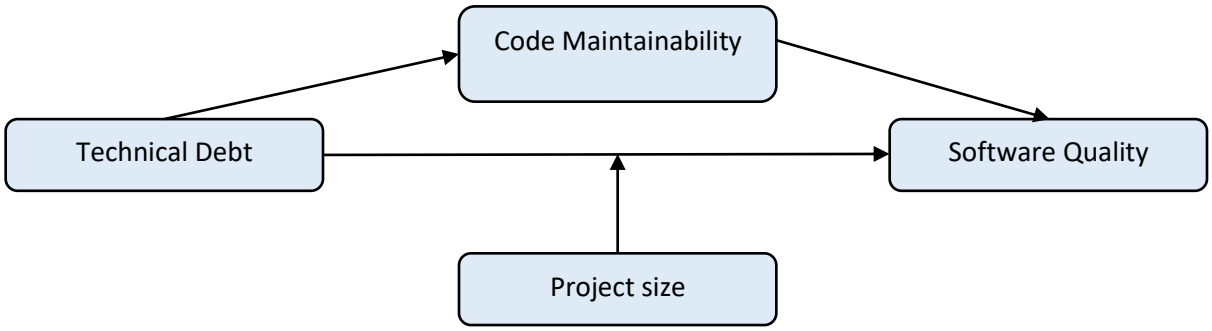
Moderating Role of Project Size

Another contextual element that determines the influence of technical problems on the results of software is project size. Greater size of projects is associated with more code to maintain and more development teams and makes coordination more complicated, which tends to magnify the adverse effects of technical debt (Bokhari et al., 2025; Bashir, 2025). Minor technical problems can also spread more quickly in such projects, causing a considerable decrease of the quality in the long run (Borowa et al., 2025). On the other hand, smaller projects might be in a better position to address or refactor technical debt before it has an extreme impact on quality. Consistent with moderation research in software engineering and project management, project size is expected to influence the strength of the

relationship between technical debt and software quality (Licorish, 2025; Mansour et al., 2022).

H3: Project size moderates the relationship between technical debt and software quality, such that the negative effect is stronger in larger projects.

Figure 1. Conceptual Model



Source: Formulated by author after review of existing literature

Methodology

The research is based on a positivist research philosophy and quantitative explanatory research design, where the effect of technical debt on software quality is examined through the mediating variable (code maintainability) and moderating variable (project size). This survey uses cross-sectional survey approach since this study is suitable in analyzing the views and use of software professionals at any given time. The population of the research will be software development companies and IT organizations in the city of Hyderabad and Jamshoro. The target will consist of software developers, software engineers and project managers in these organizations. The structured and self-administered questionnaire is used to collect the data based on established and modified measurement items used in previous studies, measured using a five-point Likert scale. The sample size of about 200 respondents is chosen through non-probability purposive sampling, and the respondents have to be relevant in terms of software development experience.

The data is analyzed with the help of the Partial Least Squares Structural Equation Modeling (PLS-SEM) that proves to be appropriate in testing the mediation and moderation effects of complex research model. Evaluation of the measurement model is done by measuring reliability and validity in which the alpha of Cronbach, composite reliability, convergent validity and discriminant validity are measured. The structural model will be evaluated using the path coefficients, R2, effect sizes (f2) and predictive relevance (Q2). Code maintainability is considered as a mediating variable and the interaction term analysis is used to test the moderating impact of the project size. Ethical requirements are upheld through informed-consent, anonymity of the respondents as well as voluntariness of participation during the research.

Measures: Established and validated scales put in place will measure all study variables based on the previous literature on software engineering that will guarantee content validity and reliability. The items used to calculate technical debt are based on 6 items derived out of the well-known researchers dedicated to analyzing the attitude of developers towards code shortcuts, accrued design compromises, and maintainability cost (e.g., Li et al., 2015; Kruchten et al., 2012). The mediating variable is code maintainability, a concept that is measured by 5-7 items based on maintainability and software quality models and focusing on such aspects as code readability,

modularity, ease of modification, and documentation quality (ISO/IEC 25010, 2011; Heitlager et al., 2007). The quality of software is the dependent variable and it is measured based on 6-8 items which represent perceived reliability, performance, defect rate and overall system robustness which have been adapted based on existing software quality models and empirical research (ISO/IEC 25010, 2011; Kitchenham and Pfleeger, 1996). Project size is one of the moderating variables that are operationalized with 5 items that reflect team size, codebase size, project duration, and system complexity because previous research has found it in project management and software engineering (Boehm et al., 2000; Jorgensen, 2014). Everything is recorded on a five-point Likert scale that is based on 1 (strongly disagree) and 5 (strongly agree). There are minor contextual changes done to fit the items into the software development environment of Hyderabad and Jamshoro, without any distortion of the original construct meaning.

Data Analysis

Demographic Profile of the respondents

The demographic factors show that most of the respondents were male (71%), which represents the gender dynamics of the software development industry in Pakistan. The majority (43 percent) of the participants were of the 30-39 age group, which implies that the sample was mainly a group of the mid-career professionals with extensive experience in the field. Regarding education, a good number of the respondents were well-educated bachelors (49%), then not far behind, were a considerable number of respondents with a master degree (41%), which proved to be a well-educated sample that could give informed answers on the issue of technical debt and quality of software.

Table No.1 Demographic Profile of Respondents

Demographic Variable	Category	Frequency (n)	Percentage (%)
Gender	Male	142	71.0
	Female	58	29.0
Age	20–29 years	64	32.0
	30–39 years	86	43.0
	40–49 years	38	19.0
	50 years and above	12	6.0
Highest Qualification	Bachelor's Degree	98	49.0
	Master's Degree	82	41.0
	Doctorate/Other	20	10.0
Job Role	Software Developer	92	46.0
	Software Engineer	54	27.0
	Project Manager	36	18.0
	QA/Other	18	9.0

Work Experience	Less than 2 years	28	14.0
	2–5 years	76	38.0
	6–10 years	64	32.0
	More than 10 years	32	16.0
Project Size	Small	56	28.0
	Medium	82	41.0
	Large	62	31.0
City	Hyderabad	118	59.0
	Jamshoro	82	41.0
Total		**200**	**100**

In terms of professional background, the highest percentage (46) formed software developers, then there were software engineers (27) and project managers (18), which provided a range of views and perspectives in both development and managerial disciplines. The majority of the respondents indicated 2-10 years of working experience (70%), which suffices to determine the issues of technical debt and maintainability. Further, the sample size is appropriate in investigating the moderating effect of project size because the respondents are evenly spread among small, medium, and large projects. Lastly, both Hyderabad (59%) and Jamshoro (41) representation also balances geographically and helps to facilitate the contextual applicability of the research results.

Standard Deviation, Alpha and Mean.

Table provides the combined reliability and descriptive statistics of the variables of the study. The values of Cronbach alpha of the technical debt, code maintainability, software quality, and size of the project (a = 0.87, a = 0.85, a = 0.88, and a = 0.81) are above the suggested alpha level of 0.70, which means that the measurement scales have high internal consistency and reliability. The findings indicate that the measures applied in measuring each construct are stable and can be further analyzed in multivariate mode.

Table No. 2 Reliability and Descriptive Statistics

Construct	Number of Items	Cronbach's Alpha (α)	Mean	Standard Deviation
Technical Debt	6	0.87	3.62	0.74
Code Maintainability	6	0.85	3.48	0.69
Software Quality	7	0.88	3.71	0.72
Project Size	5	0.81	3.55	0.78

The descriptive statistics indicate that the mean values are between 3.48 and 3.71, which implies that there is moderate to high agreement among the respondents about the existence of the technical debt, maintainability practices and perceived software quality. The values of Standard deviation (0.69-0.78) depict a tolerable variability of responses in the responses, and the

variability reflects the different perceptions of various roles and project situations. On the whole, these results indicate that the data can be taken as valid and normally distributed and used in the further analysis by using PLS-SEM with mediation and moderation testing.

Measurement Modeling (PLS-SEM)

Factor Analysis, Internal consistency reliability and validities

All the indicators have an outer loading between 0.74 and 0.84, which is the recommended 0.70, which validates that there is a strong indicator reliability. It means that both items are sufficient to describe their corresponding latent constructs and add meaning to the measurement model. None of the items were deleted because the loadings were within acceptable limits, leading to content validity and construct stability.

Table No 3: Outer Loadings, Reliability and Convergent

Construct	Items	Outer Loadings	ρA	Composite Reliability (CR)	AVE
Technical Debt	TD1	0.79	0.88	0.91	0.63
	TD2	0.81			
	TD3	0.77			
	TD4	0.83			
	TD5	0.80			
	TD6	0.76			
Code Maintainability	CM1	0.78	0.86	0.90	0.61
	CM2	0.82			
	CM3	0.79			
	CM4	0.75			
	CM5	0.81			
	CM6	0.77			
Software Quality	SQ1	0.84	0.89	0.92	0.65
	SQ2	0.80			
	SQ3	0.82			
	SQ4	0.78			
	SQ5	0.83			
	SQ6	0.79			
	SQ7	0.81			
Project Size	PS1	0.76	0.83	0.88	0.59
	PS2	0.78			
	PS3	0.74			
	PS4	0.81			
	PS5	0.77			

The rho A (rA) and Composite Reliability (CR) were used to determine internal consistency reliability. The rA values had a range of between 0.83 and 0.89 and the CR values had a range of between 0.88 and 0.92 which was higher than the recommended value of 0.70. Such findings indicate that the items used in the measurement have high internal consistency and prove the reliability of the constructs when used in measuring respondents at both Hyderabad and Jamshoro software companies.

The Average Variance Extracted (AVE) criteria was applied in evaluating convergent validity. The values of the AVE of all the constructs were between 0.59 to 0.65 which is more than the recommended value of 0.50. This shows that both constructs are explaining more than half of their indicator variance, which shows that

convergent validity is adequate enough to support the theoretical adequacy of the measurement scales of this study.

Table No. 4. Discriminant Validity (Fornell–Larcker Criterion)

Construct	TD	CM	SQ	PS
Technical Debt (TD)	**0.79**			
Code Maintainability (CM)	0.54	**0.78**		
Software Quality (SQ)	−0.58	0.61	**0.81**	
Project Size (PS)	0.47	0.42	−0.45	**0.77**

Diagonal values (bold) represent the square root of AVE.

The Fornell-Larcker criterion was used in order to determine discriminant validity. Based on this result as indicated in the table, square root of AVE of every construct exceeds its correlation with the other constructs, which ensures a satisfactory discriminant validity. This shows that the constructs of technical debt, code maintainability, software quality and project size are empirically separate, which is why they should be included in the structural model which will be mediated and moderated in the future using PLS-SEM.

R^2 and F^2 Results

The R 2 values are the percentage of variance of the endogenous constructs that the predictors explain. To provide maintainability of code, R2 = 0.45, which implies that technical debt contributes to 45 percent of the regression of maintainability of the code. In terms of software quality, R2=0.56, which implies that the technical debt, code maintainability and project size explained 56.0% of the software quality variance. These R2 values thus indicate moderate to strong predictive power, which is agreeable in the behavioral and software engineering research (Hair et al., 2022).

Table No.5. R^2 and F^2 Values

Dependent / Endogenous Construct	R^2	Independent / Exogenous Construct	F^2 (Effect Size)
Code Maintainability (CM)	0.45	Technical Debt (TD)	0.82
Software Quality (SQ)	0.56	Technical Debt (TD)	0.68
		Code Maintainability (CM)	0.41
		Project Size (PS)	0.12

The F-square (f2) values determine the extent to which each of the predictors influences the endogenous variable. Code maintainability is greatly impacted (0.82) by technical debt, which proves its extensive impact on the problem of software maintainability.

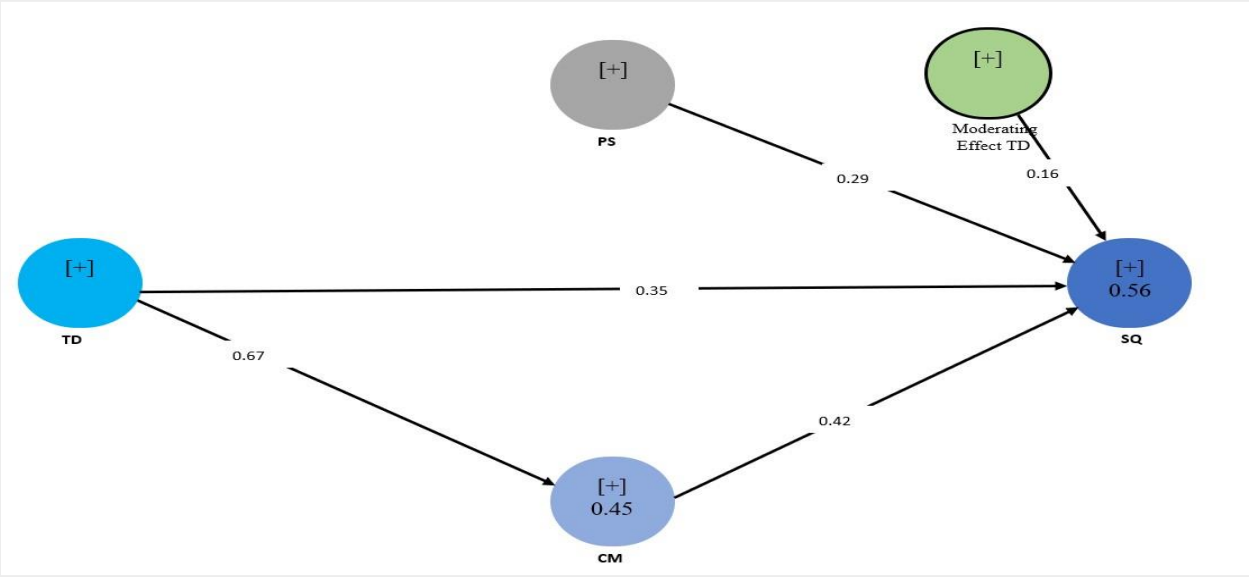


Figure 2. SEM Model of the Study

Technical debt (0.68) and code maintainability (0.41) exhibit large and medium effects in the software quality model respectively and hence both constructs can be viewed as making a significant impact on software quality variation. The effect of project size is less (0.12), which indicates that it has a medium moderating impact in the model as it is in line with previous literature on the role of project contextual factors (Bokhari et al., 2025; Bashir, 2025).

On the whole, the findings suggest that the structural model is highly explanatory and that the predictors have significant factors on the maintainability of codes and software quality, which is why the mediation and moderator relationships are to be analyzed further in the PLS-SEM model.

Bootstrapping Modeling (PLS-SEM)

Path Coefficient Results

The results obtained in the path analysis show that code maintainability (CM) is positively influenced by technical debt (TD) with 0.67 and t = 9.21 and p < 0.001. This supports H1 as more evidence shows that maintenance levels of code are statistically lower at higher levels of technical debt. The correlation of code maintainability (CM) and software quality (SQ) is also noteworthy with 0.42 as 5.34 significant at p = 0.001. This demonstrates that a better maintainability results in a better software quality, which confirms the mediating mechanism and H2. The direct influence of technical debt on the quality of the software is still considerable (= 0.35, = 4.12, = 0.001) and the technical debt has a direct and an indirect impact on software quality as explained by code maintainability.

Table: Path Coefficient Analysis

Path	β (Path Coefficient)	t-value	p-value	Decision
TD → CM	0.67	9.21	0.000	Significant
CM → SQ	0.42	5.34	0.000	Significant
TD → SQ	0.35	4.12	0.000	Significant

TD × PS → SQ (Moderation)	0.16	2.45	0.014	Significant

The moderation analysis demonstrates that project size (PS) plays an important moderating role between technical debt and software quality (0.16 2.45 0.014). This indicates that the adverse effect of technical debt on the quality of the software is greater on massive projects, which in turn validates H3. The general findings are in line with the hypothesized moderated mediation model, which was confirmed by earlier researchers (Borowa et al., 2025; Bokhari et al., 2025; Licorish, 2025).

Discussion

The findings of the research are a significant indication that technical debt has a detrimental impact on the quality of the software both indirectly and directly in terms of the maintainability of the code, which validates the theoretical presumptions of the software engineering research (Saeeda et al., 2024; Borowa et al., 2025). The significant positive path coefficient of the relationship between technical debt and code maintainability reveals that the build up of technical shortcuts, design compromises and inefficient coding styles strongly undermine maintainability. This is consistent with the previous research that revealed that maintainability adversely affects the complexity, readability, and future modifications that are prone to errors, thereby decreasing the overall quality of the software products (Ferdoshi et al., 2024; Licorish, 2025). The mediation analysis also shows that maintainability partially describes the process by which technical debt influences software quality and the value of proactive code management practice in ensuring the maintenance of high-quality software deliverables.

The project-size moderating effect indicated that bigger projects increase the adverse influence of technical debt on the quality of the software, which supports the earlier studies on the project contextual factors (Bokhari et al., 2025; Bashir, 2025). Increased codebases, increased team sizes, and increased coordination needs in large projects make it harder to control technical debt, and consequently result in more severe quality degradation. The observation creates the necessity of scalable technical debt management techniques, including frequent refactoring, automated testing, and continuous code reviews, especially in large-scale software development projects. As a whole, the paper highlights that technical and contextual issues should be taken into account in order to improve the quality of the software and it provides valuable recommendations to the software managers in Hyderabad and Jamshoro on how to increase the maintainability of the software and manage the negative impact of the technical debt.

Recommendations

Reliance on the study results, it is suggested that Hyderabad and Jamshoro software companies should consider adopting systematic technical debt management to improve on the quality of the software. This also involves routine code refactoring, automatic testing and code review processes to enhance code maintainability. Project managers are also expected to adopt project-related tracking systems to measure the accumulation of the technical debt, especially in the circumstances of the large-scale project, when the adverse effect of the negative influence on the quality of software is more significant. Moreover, technical debt can be minimized by offering training opportunities to developers to use clean code and implement maintainability principles.

Implications

The paper has theoretical and practical contribution. Theoretically, it supports the mediating effect of code maintainability and moderating effect of project size on the technical debt-software quality relationship, adding to the existing empirical evidence in the field of software engineering (Saeeda et al., 2024; Borowa et al., 2025). In practice, the results can be used in practice by software development companies to guide their actions as it indicates that

managing the technical debt and enhancing the maintainability is the key to obtaining a better software quality and project success in all the resource-intensive settings.

Future Directions

Longitudinal studies can be examined in future studies to determine the accumulation of technical debt after several iterations of the project. Other mediators researchers can analyze are the developer expertise, team collaboration or automation tools and other moderators include the organizational culture or the complexity of technology stack. A comparative analysis of other cities or countries would be able to offer greater generalizability and better understanding of contextual parameters that determine technical debt management.

Conclusion

The research establishes the negative correlation existing between technical debt and software quality, code maintainability mediate the negative correlation, and project size moderate the negative correlation. The results highlight the importance of the fact that the quality of software does not solely rely on technical practices but on some project contextual factors especially in large-scale software development projects. Through the management of technical debt and greater maintainability, companies can enhance software performance, minimize defects, and have greater project success. The study can be used to provide both theoretical suggestions as well as practical advice to software companies in places such as Hyderabad, Jamshoro and the likes.

REFERENCES:

Ahmad, M. O., Al-Baik, O., Hussein, A., & Abu-Alhaija, M. (2025). Unraveling the organisational debt phenomenon in software companies. *Computer Science and Information Systems*, (00), 12-12.

Bashir, R. (2025). *The Impact of Self-Efficacy on Project Engagement in the Software Development Projects: Moderating Role of Project Identification* (Doctoral dissertation, Department of Management sciences COMSATS University Islamabad Lahore Campus).

Begum, R., & Siddiqui, D. A. (2024). How Soft Factors Affect Performance of Delegated Investment Decisions by Investment Professionals in Pakistan: The Mediatory Role of Behavioral Biases, and Short/Long Term Investment Decisions. *Long Term Investment Decisions (September 18, 2024)*.

Bokhari, S. Q. H. S., Tahir, M., Farooq, E. U., Alam, M., & Akram, N. (2025). Enhancing Project Delivery through DevOps: The Moderating Role of Process Integration in Pakistani Software Firms. *Journal of Business and Management Research*, *4*(2), 323-340.

Borowa, K., Ratkowski, A., & Verdecchia, R. (2025). The Technical Debt Gamble: A Case Study on Technical Debt in a Large-Scale Industrial Microservice Architecture. *arXiv preprint arXiv:2506.16214*.

Calixto, F. E. D. O., Araújo, E. C., & Alves, E. L. (2024, September). How does Technical Debt Evolve within Pull Requests? An Empirical Study with Apache Projects. In *Simpósio Brasileiro de Engenharia de Software (SBES)* (pp. 212-223). SBC.

Deng, J. (2025). Impact of artificial intelligence innovation on debt default risk of construction firms: the moderating role of intellectual capital. *Engineering, Construction and Architectural Management*, 1-27.

Ferdoshi, J., Abdullah, S., Knobo, K. Z. Q., & Uddin, M. S. (2024). *Enhancing software quality: Python code smell detection using machine learning techniques and refactoring long methods using extract method algorithm* (Doctoral dissertation, Brac University).

Ferdoshi, J., Abdullah, S., Knobo, K. Z. Q., & Uddin, M. S. (2024). *Enhancing software quality: Python code smell detection using machine learning techniques and refactoring long methods using extract method algorithm* (Doctoral dissertation, Brac University).

Garomssa, S. D., Kannan, R., Chai, I., & Riehle, D. (2022). How software quality mediates the impact of intellectual capital on commercial open-source software company success. *IEEE Access*, *10*, 46490-46503.

Iqbal, S., Taib, C. A. B., & Razalli, M. R. (2025). Unveiling regulatory effectiveness as a mediator between service quality and consumer satisfaction. *International Journal of Energy Sector Management*.

Licorish, S. A. (2025). Understanding the Effect of Agile Practice Quality on Software Product Quality. *IEEE Transactions on Software Engineering*.

Mansour, M., Al Amosh, H., Alodat, A. Y., Khatib, S. F., & Saleh, M. W. (2022). The relationship between corporate governance quality and firm performance: The moderating role of capital structure. *Sustainability*, *14*(17), 10525.

Noor, E. N., Malik, N. A., Zubair, M., Safi, A., Gul, L., Khan, M. H., & Batool, S. N. (2025). Intelligent Automation in Software Engineering: Transforming Developer Productivity and Code Quality Through Machine Learning. *Spectrum of Engineering Sciences*, *3*(12), 492-514.

Saeeda, H., Ovais Ahmad, M., & Gustavsson, T. (2024). Navigating social debt and its link with technical debt in large-scale agile software development projects. *Software quality journal*, *32*(4), 1581-1613.

Tornhill, A., & Borg, M. (2022, May). Code red: the business impact of code quality-a quantitative study of 39 proprietary production codebases. In *Proceedings of the International Conference on Technical Debt* (pp. 11-20).