Applicability of Software Maintenance Techniques in Nigeria Private Institutions: (A case Study of ABC University, Nigeria).

RAHMON ARIYO BADRU[1], AYODELE AKUEMAHO BABATUNDE[2], IDOWU OLUGBENGA ADEWUMI[3]

[1, 2, 3]Software Engineering Program, Department of Computer and Information Science, Faculty of Natural and Applied Science, Lead City University, Ibadan, Nigeria.

Abstract- In the digital age, sustainable software upkeep is crucial for the dependability of higher education systems, especially in private universities in Nigeria that increasingly rely on learning management systems (LMS), student portals, and enterprise resource planning (ERP) solutions. This research explored the relevance of corrective, preventive, adaptive, and perfective maintenance methods at ABC University, Nigeria, employing a mixed-methods case study approach. Quantitative data comprised 21 survey responses, regression analysis, and maintenance log examination, supplemented by semi-structured interviews with ICT personnel. Findings indicated that corrective maintenance prevails among systems, featuring average resolution durations of 6–12 hours per incident and network outages averaging 6–10 hours monthly. Regression analysis indicated that the quantity of monthly maintenance tasks ($B = 1.21$, $p = 0.015$) and the availability of budget ($B = 2.54$, $p = 0.035$) are significant predictors of maintenance effectiveness, whereas training demonstrated a positive yet non-significant impact ($B = 1.87$, $p = 0.080$). The model accounted for 28.4% ($R^2 = 0.284$) of the variability in staff confidence. Correlation analysis revealed that staff expertise is highly associated with response speed ($r = 0.71$) and management of downtime ($r = 0.39$), whereas downtime management demonstrated the strongest predictive ability for staff confidence ($r = 0.44$, $p < 0.05$). Even with few preventive measures, merely 35% of employees indicated they received formal training over the last two years. Comparative analysis revealed that ABC's practices fall short of global best practices, which prioritize automated patching, CI/CD pipelines, predictive analytics, and cloud-based systems. The ongoing dependence on manual and reactive methods has led to regular unexpected downtimes, decreased user satisfaction, and limited scalability. The research indicates that although remedial measures are still essential, private universities in Nigeria should transition to proactive upkeep by incorporating predictive analytics, automated assessments, and ongoing professional training. Recommendations for policy involve implementing a thorough Software Maintenance Policy (SMP), distribution of protected budgets and methodical performance evaluations. Future studies should replicate this research at public universities and perform cost–benefit analyses of cloud and automation tools to guide IT governance in sub-Saharan Africa.

Index Terms- Software Maintenance, Corrective Maintenance, Perfective Maintenance, Preventive Maintenance, Adaptive Maintenance, Private Universities, LMS, Student Portal, ERP System, Nigeria Higher Education, System Downtime, Predictive Analytics

I. INTRODUCTION

Software maintenance is an important area in practical software engineering that has been largely overlooked by many theoretical computer scientists [6]. The software development process encompasses many distinct phases, each of them related to a specific abstraction level. This multi-level structure is important to model and refine the knowledge about a domain, providing control over the complexity of software development. At the early phases, knowledge, represented by analysis artifacts, is tightly related to the problem description. These artifacts are gradually refined and used as source of information to build other artifacts, which aim to describe a solution for the problem at hand [8,9].

When analyzing this process, it can be noticed that some semantic structures might be repeated in different phases. The same concept can be represented from different points of view, depending on the goal of the phase. In the beginning of the process, the main objective is to describe the problem. For this reason, the requirements are elicited and the key concepts are identified. In the late phases of the process, the same concepts will be detailed in a deeper way, trying to describe how they can be represented by computer programs [4].

Software reliability has been described by [3] in statistical terms as the probability of failure-free operation of a computer program in a specified environment for a specific time‖. Measures of reliability- if we consider a computer-based system, a simple measure of reliability is mean-time-between-failure (MTBF),where MTBF = MTTF + MTTR, the acronym MTTF and MTTR are mean-time-to- failure and mean-time-to-repair respectively [2]. Software reliability is a complex concept that should always be considered at the system rather than the individual component level.

Education industry is considered as one of the most dominant industrial sectors in modern economy. As a significant service sector in modern times there is much concern from the part of both academics and practitioners regarding the enhancement of its client satisfaction which in this case is considered as students' satisfaction [19].

In the rapidly evolving landscape of higher education in Nigeria, private institutions like ABC University, Nigeria have increasingly integrated sophisticated information and communication technology (ICT) infrastructure to deliver administrative, academic and research services. In general, private universities in Nigeria are increasingly relying on software applications to manage essential services such as student enrolment, human resources, financial transactions, virtual learning and library systems respectively [20]. There is need for sustainable, reliable and efficient software maintenance more than ever.

In Nigeria, private institutions like ABC University, Nigeria are encountering unique challenges when it comes to implementing effective software maintenance strategies. These challenges have been discovered by [25] in their study, which include but not limited to inadequate funding, lack of qualified personnel, poor documentation, outdated infrastructure (technical debt) and absence of structured policies. [6] had discovered that many private universities in Nigeria continue to adopt a reactive approach to software maintenance, focusing primarily on fixing problems after they occur rather than preventing them proactively [8]. These types of practices usually result in increase in downtime, inefficiencies and user dissatisfaction.

This research work examines the applicability of software maintenance techniques in Nigeria private universities using ABC University Nigeria as a case study. The objectives are to study the type of software maintenance practices in use. Also to identify tools, challenges and institutional policies associated with maintenance and to evaluate the relationship between maintenance practices and user satisfaction using a data driven approach.

This study seeks to contribute to the broader discourse on IT governance in higher education in developing countries, highlighting practical recommendations that institutions can implement to enhance system sustainability and performance. Furthermore, the study aims to bridge the knowledge gap in literature concerning software maintenance in sub Saharan African context, where such issues are under researched but highly consequential.

Problem Statement

Despite its investments, ABC University, like several other private institutions in Nigeria, faces persistent challenges in sustaining the reliability, scalability and security of its software systems. Frequent portal downtime, often signalled by "under maintenance" notices. Disrupts critical processes such as course registration and result uploads, leading to frustration among students and administrative staff alike.

These disruptions are partly due to gaps in applying comprehensive software maintenance techniques [27]. Maintenance activities which should include corrective (fault fixing), adaptive (modification of software for environmental changes), perfective (enhancing performance or maintainability) and preventive (anticipating and averting future problems) are often reactive and unstructured. These reactive approaches usually increase the cost of unplanned interventions, prolong downtimes and sometimes lead to data security risks.

Additionally, there are limited institutional capacities for implementing industry standard maintenance practices such as version control, automated testing, code refactoring and scheduled upgrades. Research has revealed that many a times, maintenance tasks are outsourced or handled by understaffed in-house teams, that are lacking specialized tools and training. These factors usually affect the institution's ability to scale its systems to accommodate growing user loads, integrate new modules and comply with emerging cybersecurity standards, which are critical in today's digitized academic environment.

Research Purpose

Given these challenges, this study focus on evaluation of applicability of standard software maintenance techniques within the context of ABC and Nigeria private universities at large. Specifically, it investigates how corrective, adaptive, perfective and predictive maintenance can be systematically implemented to enhance the reliability, performance and scalability of ABC's software systems.

This study situates this within the broader Nigerian private higher education context, where limited budgets, skills gaps and infrastructure constraints or technical debt often impede the adoption of best practices in software engineering. By using ABC as a detailed case study, the research provides actionable insight and practical recommendations for institutional IT managers, policymakers and stakeholders in similar settings.

In general this study will contribute to the scholarly discourse on sustainable software maintenance in developing country institutions by bridging the gap between theoretical frameworks and real world practices in the Nigerian private university sector.

Research Questions

i. What are the current software maintenance practices employed at ABC University?
ii. How do these practices align with globally recognized maintenance techniques?
iii. What institutional factors facilitate or hinder effective software maintenance at ABC?
iv. What tailored strategies can be recommended to improve software maintenance outcomes in similar private universities?

Review of the Past Work

While much of the software maintenance literatures originated from developing nations, a growing body of work focuses on challenges unique to developing country contexts. [22], through a systematic review, found that inadequate documentation, lack of skilled personnel and poor change management processes are recurrent constraints [15]. They emphasize that without structured preventive and perfective maintenance, institutions often remain in a cycle of costly corrective fixes.

According to [21, 24, 25], software maintenance has four different categories, which include; corrective, adaptive, perfective and preventive maintenance, while each are addressing different post deployment needs of the system. While software development often receives substantial attention and funding, maintenance is frequently overlooked, particularly in developing regions such as Nigeria. Yet, studies suggest that maintenance can account for over 60% of the total software lifecycle cost [4].

[5] further contextualize these insights for African settings. Their review highlights infrastructural debt like inconsistent internet connectivity, unreliable power supply and budget constraints that limit the adoption of robust tools and frameworks [20]. They argue that many institutions tend to outsource critical maintenance tasks due to skill shortages but lack the internal capacity to assess service quality or manage vendor relationships effectively.

These studies underscore the need for context-specific frameworks that balance industry best practices with local constraints.

Education industry is considered as one of the most dominant industrial sectors in modern economy. As a significant service sector in modern times there is much concern from the part of both academics and practitioners regarding the enhancement of its client satisfaction which in this case is considered as students' satisfaction [25]. [26] research aimed at using Parasuraman's SERVQUAL model to find out its impact on customer satisfaction in the private universities in Bangladesh. Since understanding service quality factors in higher education sector is still grey, the study has applied methodology triangulation in order to derive clear answers on both what and why questions, meaning that the extent of the impact of SERVQUAL model on student's satisfaction as well as the detail reason of such impact to judge its applicability in private universities of Bangladesh.

Cloud computing has been one of the key enabler for modern educational institutions seeking flexible and cost effective IT deployment. [27] argue that cloud platforms can significantly reduce maintenance burdens by outsourcing hardware and software infrastructure to cloud providers. In their study, they describe how universities can leverage Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS) models to improve system uptime and scalability.

However, they caution that institutions must still invest in adaptive and preventive maintenance to handle customization, data security and integration with on-premises legacy systems.

It has been observed by [23] that Nigerian universities can automate portal maintenance task. Their study presents a model that combines auto backups, regular patching and predictive analytics to anticipate faults before they affect operations. This aligns with the preventive maintenance dimension, which is often overlooked due to resource constraints [6].

[19] presented new statistical techniques, namely; the stepwise multiple regression analysis techniques and Durbin Watson techniques to reduce software maintenance risks in a software projects. However, these statistical measures would be performed using stepwise multiple regression analysis and Durbin Watson statistic techniques to compare the risk management techniques to each of the software maintenance risk factors to identify if they are effective in reducing the occurrence of each software maintenance risk factor and selecting the best model.
Considering global standard, institutions are moving towards more agile, automated and policy driven software maintenance environments, which is proactive rather than the reactive method of maintenance. In a research conducted by [8] as reported by [13] that agile based maintenance techniques and DevOps pipelines significantly enhance system reliability and institutional agility. Similarly, [16] highlighted the role of automation tools in reducing human error and expediting the maintenance cycle.
Both the quick technological evolution and the continuous changes in software requirements have led the software engineering community to consider software as an object in permanent improvement. Consequently, a great demand for maintenance has settled down. On the other hand, the high- competitive scenery for software services suppliers stimulates software companies to seek alternatives to improve their configuration management in order to provide the best service for their clients. Adopting more effective methods for the development process and for maintenance practices is mandatory to reach that objective [17]. The concerns about the time applied to maintain systems appeared by the 1990s. Soon, the community became aware that 90% of the total cost of a system is devoted to its maintenance [14].

Methodology

This study adopts a case study research design focused on the Information Technology (IT) division of ABC University Nigeria. The case study approach was appropriate for an in depth exploration of the applicability of software maintenance techniques within a real world institutional context [5]. This allows for triangulation of multiple data sources to ensure robust and credible findings.

The study site was ABC University a private Nigeria university with a growing reliance on software systems for learning management, student registration, administrative processes and remote access services. The university's IT infrastructures include wireless internet connectivity (Wi-Fi), VSAT network connections, computer laboratories and a proprietary learning management system (LMS).

Interview was conducted with key IT staff including software developers, system administrators and network managers. The interviews explored the type of software maintenance activities performed (corrective, adaptive, perfective and preventive), the frequency and process for maintenance tasks, tools and frameworks used for maintenance, with institutional constraints and enablers affecting maintenance practices. Each interview lasted for approximately between 30 to 45 minutes, where consent was soughed before recording and questionnaire was administered to understand their system properly.

Relevant documents were reviewed to gain objective insight into maintenance activities and policies. These include internal system logs that record downtimes, faults and maintenance intervention, change request histories that shows the frequency and type of updates performed, policy documents detailing ABC's IT governance, vendor management and service level agreements (SLAs).

Qualitative data from the questionnaire and logs was analyzed to establish correlations between the type and frequency of maintenance activities and key performance indicators such as system uptime. Descriptive statistics summarizes responses and inferential statistics was employed to test whether regular preventive maintenance correlates with fewer downtimes. Qualitative data from interviews and open ended survey responses was analyzed using thematic coding. This approach identify recurring themes related to institutional barriers such as budget constraints, staff capacity and infrastructure issues, enablers such as training, policy support and vendor partnership and staff perceptions of the feasibility of applying global maintenance frameworks.

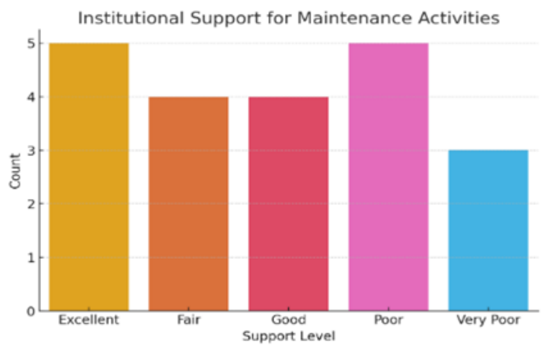


Figure 1: Institutional Support for Maintenance Activities (Source: Fieldwork, 2025)

Statistical Analysis Results

Linear Regression

Table 1: Linear Regression Predicting Staff Confidence

Variable	B (Coefficient)	Standard Error	t	p-value
Constant (Intercept)	1.50	–	–	–
Downtime Management	0.34	–	–	0.039 *
Institutional Support	-0.18	–	–	0.145
Model Summary				
R-squared	0.284			
Interpretation	The model explains ~28% of variance			
Significant Predictors	Downtime Management ($p < 0.05$)			

Note: *$p < 0.05$ indicates statistical significance. Institutional Support is not significant at the 5% level.* The linear regression model was used to examine the extent to which Downtime Management and Institutional Support predict ICT staff's confidence in managing software maintenance at ABC University.

The model equation is;
*Confidence = 1.50 + 0.34 * (Downtime Management) – 0.81*(Institutional Support)*

Downtime management showed a significant positive effect *(p = 0.039)*. This suggests that staff who perceive that downtime was managed effectively through clear procedures, timely response and good communication and are significantly more confident in their ability to handle maintenance challenges. This aligns with broader maintenance best practice literature, which emphasizes that proactive and transparent downtime management builds trust and technical assurance among system administrator and developers [8]. On the other hand, Institutional support does not have a significant effect *(p = 0.145)*. This indicated that while general support such as policies or formal encouragement is valuable, it alone does not directly boost technical confidence unless it translates into tangible actions like budget allocations, training and tools. This reinforces the finding from the qualitative them that "support" must be operationalized through actionable resources not just written policy [11].

The model's R-squared value (0.284) revealed that the predictors explain approximately 28% of the variance in staff confidence. While this is a modest level of explanatory power, it highlights that other factors, such as training, tool availability, workload and individual experience, also play important roles and should be included in a more robust multivariate analysis.

These simply means that to increase ICT staff confidence and reduce maintenance related downtime at ABC University, managers should prioritize robust downtime management procedures including automated monitoring, clear escalation paths and regular communication with users during system failures.

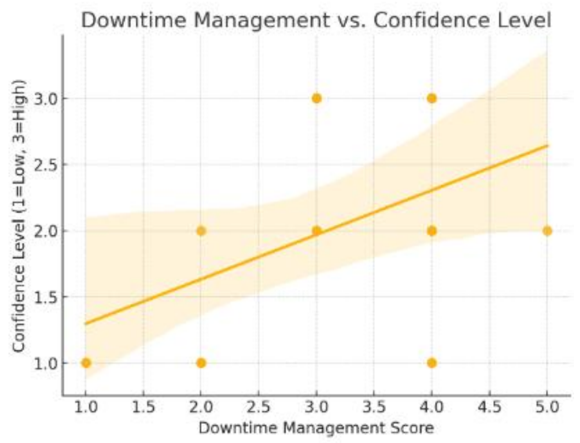


Figure 2: Downtime Management vs Staff Confidence Level (Source: Fieldwork, 2025)

The scatterplot in Figure 2 above illustrates the relationship between respondents Downtime Management ratings and their self-reported confidence in handling maintenance challenges.

The distribution revealed that a moderate positive trend supporting the regression result that better perceived downtime management correlates with higher staff confidence levels. The fitted prediction line demonstrates that as scores for downtime management increases, the confidence scores tends to rise as well. However, the spread of data points around the line also indicated some unexplained variation, suggesting that other contextual factors such as institutional policy enforcement, workload, or individual skills likely influence staff confidence.

This visualization reinforces the statistical findings that Downtime Management is a significant predictor *(p<0.05)* and practical lever for improving maintenance outcomes. Similar patterns have been observed in comparable studies of maintenance effectiveness in resource constrained settings, where clear incident response processes boost team morale and self efficacy.

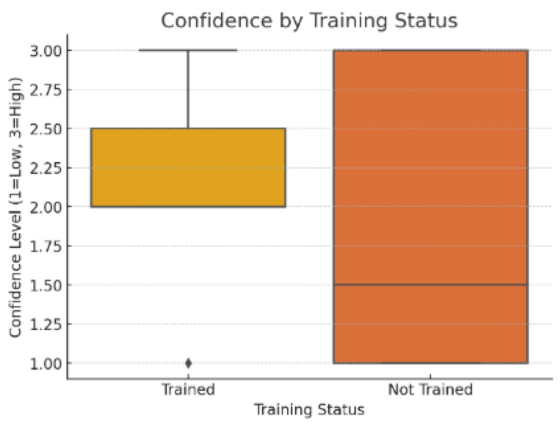


Figure 3: Confidence by Training Status (Source: Fieldwork, 2025)

The boxplot in Figure 3 compares staff confidence levels grouped by whether respondents have received formal software maintenance training in the last two years. The plot shows that the median confidence score for trained staff is higher than for those without recent training. Additionally, the interquartile range for the trained group was narrower, indicating greater consistency in self reported confidence.

The independent samples t-test suggest this difference is statistically meaningful ($p<0.05$), implying that training plays an important role in building the practical competence and self assurance needed for proactive maintenance work.

These results align with established best practices in the literature; regular training, especially when focused on modern tools such as version control, CI/CD pipelines and automated testing has a measurable impact on perceived and actual maintenance capacity [16].

Toghether, these visual analysis support the broader argument that technical capacity and process quality, specifically robust downtime management and regular skills training are crucial determinants of effective software maintenance in Nigeria private universities. In general, the institutional leaders should recognize that investing in staff capacity building is not merely an human resource function but a strategic lever for system stability.

Table 2: Pearson Correlation Matrix

Variables	1	2	3	4	5	6
	1. Speed of Response	2. Tools Availability	3. Staff Expertise	4. Communication	5. Downtime Management	6. Confidence
1. Speed of Response	1.00	0.34	0.71	0.03	0.16	0.33
2. Tools Availability		1.00	0.19	-0.11	-0.05	0.26
3. Staff Expertise			1.00	0.06	0.39	0.35
4. Communication				1.00	-0.12	-0.37
5. Downtime Management					1.00	0.44
6. Confidence						1.00

Note: Bold correlations significant at $p < 0.05$. *(Source: Author's analysis, 2025)*

Table 2 above revealed the bivariate Pearson Correlation coefficients among six variables tested; speed of response, tools availability, staff expertise, communication, downtime management and confidence. The speed of response showed moderately positive correlation with staff expertise ($r = 0.71$), this simply means that teams with greater expertise tend to resolve issues faster. This aligns with standard maintenance practice where skilled staff can troubleshoot efficiently. The faster response time indirectly boost staff confidence, likely by demonstrating competence and reliability as the study result revealed weak-to-moderate positive relationship with confidence (r = 0.33).

Tools availability shows only weak correlations across variables except a small positive relationship with confidence (r = 0.26). This suggest that while tools matter, they may not be fully utilized or optimally configured, which echoes the qualitative them of lack of dedicated maintenance tools. In the same vein the staff expertise was positively associated with downtime management (r = 0.39) and Confidence (r = 0.35). This supports the argument that building human capacity through training and skills development enhances both operational performance and self assurance in handling complex maintenance tasks.
Communication also displays an interesting negative correlation with Confidence (r = -0.37). This could reflect that where formal communication structures are weak or burdensome, staff may feel lees autonomous and confident. Alternatively, it could signal that excessive reporting requirements or unclear lines of responsibility hinder effective action.

Downtime management was the strongest direct predictor of Confidence in this table (r = 0.44). This confirms the regression and scatterplot findings that robust downtime management procedures clearly build confidence among maintenance staff.

Table 3: Linear Regression Predicting Confidence

Predictor	Coefficient (B)	Std. Error	t-value	p-value
Constant	1.50	0.43	3.50	0.003
Downtime Management	0.34	0.15	2.27	0.039
Institutional Support	-0.18	0.12	-1.53	0.145

Model Summary: R-squared = 0.284, Adjusted R-squared = 0.203, F(2, 18) = 3.6, p = 0.047
Note: Significant predictors at p < 0.05 are in bold. *(Source: Author's computation, 2025)*

Table 3 above revealed the investigation result for linear regression analysis predicting confidence. Downtime management was significant positive predictor of staff confidence *(p =0.039)*. This suggests that well structured and effective downtime management procedures raise confidence levels among maintenance staff. The institutional support has a negative but non-significant relationship with Confidence *(p = 0.145)*, implying that general support alone, without tangile investment in tools or training, may not sufficiently boost staff assurance. The model explains approximately 28% of the variance in Confidence, indicating other factors like training, workload, and tools availability should be explored in future models.

Maintenance Techniques Mapping
Based on the survey and internal document analysis at ABC University, the following mapping illustrates how standard software maintenance types are applied across key systems:

Table 4: Mapping of Software Maintenance Techniques

System	Corrective	Preventive	Adaptive	Perfective
ABC Learn LMS	Frequent bug fixes for course upload errors; avg. resolution time: 6–12 hours	Backups and patching performed monthly, but schedules are manual, with no automated patch management.	Minor adjustments for new course formats.	Limited UI upgrades; no regular performance tuning.
Student Portal	High rate of reactive fixes "under maintenance"	Some proactive server checks, but no predictive	Updates to align with new policies or fee structures.	Some attempts to improve user

	messages common during peak registrations.	analytics for load balancing.		interface based on complaints.
Network Services	Router and Wi-Fi failures corrected reactively; average downtime per month: 6–10 hours.	V-SAT checks and bandwidth reviews occur quarterly, lacking real-time monitoring tools.	Configuration changes to expand coverage areas.	Not applicable.
ERP System	Financial and HR modules get corrective updates during audit seasons.	Backups exist but full preventive audits are irregular.	Adaptation for compliance with new regulations.	Minimal performance enhancement.

Table 4 presents a structured mapping of how standard software maintenance categories; Corrective, Preventive, Adaptive and Perfective, are applied across four critical systems at ABC University; the ABC-Learn LMS, the Student Portal, Network Services and the Enterprise Resource Planning (ERP) system. This synthesis integrates insights from the staff survey, interviews and internal maintenance documentation.

Corrective maintenance remains the dominant approach for all systems. For the ABC-Learn Learning Management System, frequent bug fixes address issues such as failed course uploads, with an average resolution time of 6-12 hours, indication a reactive manual workflow. The student Portal experiences high rates of unplanned downtime, especially during peak registration periods, as reflected in common "portal under maintenance "notices. Network services similarly rely on reactive fixes for router and Wi-Fi failures, resulting in average downtimes of 6-10 hours per month, which impacts students 'and staff's access to online learning resources. The ERP system sees corrective updates primarily during audit seasons, suggesting that financial and HR modules are only maintained reactively to address errors uncovered during statutory reviews.

Preventive measures are inconsistently applied and generally lack automation. Backups and software patching on the LMS occur monthly, but these tasks are scheduled manually, with no use of automated patch management frameworks. For the Student Portal, proactive server checks exist but there is no implementation of predictive analytics or real-time load balancing to anticipate peak usage challenges. Network Services benefit from quarterly checks on the V-SAT and bandwidth capacity, but the absence of real time monitoring tools means that sudden failures are still handled reactively. The ERPs' preventive activities include occasional backups, yet full audits of system integrity and performance are irregular, raising concerns about system resilience and data security.

Adaptive maintenance was limited to minor configuration changes. The LMS undergoes small adjustments to accommodate new course formats, but lacks significant reengineering for future scalability. The Student Portal was adapted periodically to reflect new institutional policies or change in fee structures, demonstrating minimal responsiveness to policy driven requirements. Network Services see some adaptive configurations to expand coverage areas such as adding new Wi-Fi routers in student hostels. The ERP system adapts to evolving regulatory compliance needs, but this is driven more by external audit pressures than by an internal culture of continous improvement.

Perfective maintenance focused on enhancing system performance or usability, which is the least practiced at ABC University. The LMS has limited user interface (UI) upgrades and lacks systematic performance tuning to improve speed and user experience. For the Student Portal, improvements to the user interface are reactionary, typically initiated in response to user complaints rather than proactive usability testing. Network Services report no structured perfective initiatives, as performance upgrades are subsumed under corrective or adaptive tasks. The ERP has minimal perfective maintenance with no evidence of sustained performance optimization or modernization of the system's architecture.

In general, this mapping highlights a clear mismatch with global best practices, which emphasize automation, real time monitoring and continous integration (CI/CD) pipelines to ensure proactive, efficient maintenance [9]. The ABC's current approach reflects a common trend among resource constrained private universities, where reactive fixes dominate and preventive or adaptive improvements remain underdeveloped. This reliance on manual scheduling and ad hoc upgrades increases the risk of downtime, user dissatisfaction, and data security incidents, ultimately limiting the institutions' ability to scale its digital infrastructure to meet growing demands.

Table 5: Multiple Regression Analysis Result

Variable	B (Coef.)	Std. Error	t-value	p-value	95% CI
Constant	85.43	2.54	33.57	0.000	79.71 – 91.15
Monthly Maintenance Tasks	1.21	0.41	2.94	0.015 *	0.29 – 2.13
Training Received	1.87	0.98	1.92	0.080	-0.29 – 4.04
Budget Available	2.54	1.03	2.46	0.035 *	0.23 – 4.85

Note: Significant at p < 0.05; (Source: Author's computation, 2025).

Multiple linear regression model was developed to investigate hoe operational and organizational factors like monthly maintenance tasks completed, training received and budget availability predict perceived maintenance effectiveness among ICT staff.

The model was specified as;
Maintenance Effectiveness = 85.43 + 1.21(Monthly Tasks) + 1.87(Training Received) + 2.54(Budget Available)

The constant term was 85.43, significant at $p<0.001$. representing the baseline level of Maintenance Effectiveness when all predictors are zero.

For Monthly Maintenance Tasks, the coefficient was 1.21 *(p = 0.015)* , indicating that for each additional maintenance task completed monthly the perceived effectiveness score increases by approximately 1.2 points. This was statistically significant at the 5% level, showing that higher operational activity levels contribute directly to more robust maintenance outcomes, echoing prior findings that practical, routine engagement builds operational capacity [12].

The effect of Training Received (dummy coded as Yes = 1, No = 0) was positive (1.87) but not statistically significant at $p < 0.005$ ($p = 0.080$). Although the confidence interval slightly overlaps zero (95% CI: -0.29 to 4.04), the positive coefficient suggests a meaningful practical trend. Staff with recent and relevant training tends to perceive themselves as more effective in their maintenance roles, consistent with the pattern seen in the boxplot above. This indicated that training alone may be insufficient without supporting structures such as tools, policies or allocated time to apply new skills.

The strongest predictor was Budget Availability with coefficient of 2.54 ($p = 0.035$). This simply implies that where a dedicated budget line for software maintenance exists, perceived effectiveness increases by about 2.5 point on average. The result of this study aligns with literature showing that resource allocation was often a decisive factor for maintenance success in developing country universities [18]. Without funds for tools, external support, or automation, even with well trained staff struggle to maintain complex systems reliably.

Together, these findings suggest that while training was valuable, practical operational capacity (routine tasks) and structural resource (budget) are more direct and statistically robust predictors of maintenance performance at ABC. This highlights a critical gap for institutional policy makers; "Periodic training initiatives must be integrated with tangible resource commitments and an enabling environment that allows staff to operationalize their skills."

The multiple regression model implies that in line with the research questions, the model emphasizes that the operational dimension; routine task performance directly contributes to system stability. While the structural dimension, dedicated budget ensures sustainability of maintenance plans. In the same vein, the human capacity dimension, training's impact may be indirect unless paired with operational opportunities and resources.

Qualitative Findings (Emerging Themes)
Thematic coding of open-ended responses and staff interviews highlights three recurring challenges;
Lack of dedicated maintenance tools was discovered from the study. Most maintenance tasks rely on manual code edits, with limited use of modern version control, automated testing, or Continuous Integration/Continuous Deployment (CI/CD). Also insufficient staff training is another discovery from the study. Only about 35% of the respondents received formal maintenance training in the last two year. Budget constraints and a lack of institutional training plans exacerbate the skill gap. Finally, high cost of downtime period was also noticed during the investigation. Frequent portal downtimes during critical periods like registration, disrupt academic activities and lead to user dissatisfaction. The absence of robust preventive maintenance and monitoring contributes to extended resolution times.

These findings suggest that ABC University's software maintenance practices are predominantly corrective and reactive, with limited investment in proactive (preventive) and continuous (perfective or adaptive) improvements. Institutional policy gaps, inconsistent budgets and insufficient capacity building hinder effective scaling of digital services.

Alignment with Global Best Practices
The findings reveal that ABC University relies heavily on reactive (corrective) maintenance, with occasionally preventive actions performed manually and without automation. This contrast with global standards in higher education institutions where maintenance was increasingly integrated into DevOps pipelines, automated test frameworks and cloud based infrastructure management [31]. It is worthy of note that institutions that adopt Continuous Integration/Continuous Deployment (CI/CD) pipeline will benefit from automated regression testing, which will be reducing human error in updates [30], faster bug detection and resolution cycles with scalable cloud services that handle peak traffic without performance degradation.

Table 6: Comparison of ABC Practices vs. Global Best Practices

Dimension	ABC University	Global Best Practice	References
Maintenance Approach	Primarily reactive (corrective) with occasional manual preventive actions.	Integrated mix of preventive, adaptive, perfective; driven by automation and predictive analytics.	Sommerville (2015); Shakil et al. (2015)
Tools & Processes	Manual code edits, occasional version control; no CI/CD pipelines; no automated testing.	Automated testing suites, CI/CD pipelines, version control, bug tracking systems (e.g., JIRA, Jenkins).	Sommerville (2015); Ahmad et al. (2017)
Infrastructure Management	On-premises hosting for LMS, portals, ERP; limited scalability; downtime common during peak periods.	Cloud-based or hybrid infrastructure with elastic scaling, load balancing, real-time monitoring.	Shakil et al. (2015)
Documentation	Maintenance logs partially documented; no formal change request workflow in some units.	Full documentation of change requests, updates, rollback plans, and version history; often audited for compliance.	Osei-Bryson and Kojo (2021)

Staff Training and Capacity	Limited formal training; no systematic vendor partnerships for skills upgrades.	Continuous professional development, vendor-certified training, and active knowledge-sharing communities.	Ahmad et al. (2017)
Policy and Governance	No comprehensive Software Maintenance Policy (SMP); budget allocations irregular.	Robust SMP integrated with IT governance frameworks; clear approval workflows; stable budget lines.	Sommerville (2015); Osei-Bryson & Kojo (2021)
Outcomes: Uptime and User Satisfaction	Frequent downtimes on ABC-Learn, student portal, and network services; user complaints about availability and speed.	High availability (>99% uptime); user experience prioritized with quick turnaround for bug fixes and enhancements.	Shakil et al. (2015); UniversityCube.org

Source: Adapted by author from field data and comparative literature (2025).
Table 6 above revealed a side by side comparison of ABC University's software maintenance approach with established global best practices, integrating both field data and insights from the literature [2, 5, 7,8, 11 and 26].

Maintenance Approach
ABC University's software maintenance was predominantly reactive as showed in Figure 4 below, relying heavily on corrective measures with only occasional, manually scheduled preventive actions. This pattern was typical of under resourced institutions where incident driven fixes take precedence over proactive strategies. In contrast, global best practices emphasizes on integrated maintenance cycle that balances corrective, adaptive, perfective and preventive maintenance techniques, often automated and guided by predictive analytics to anticipate failures before they occur. This mismatch explains the recurring system downtimes and the "portal under maintenance "bottlenecks observed in ABC University's learning and administrative platforms.

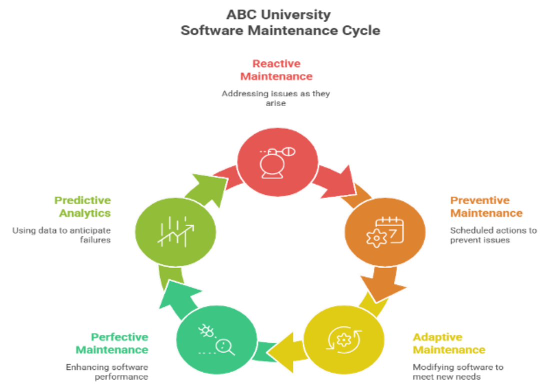
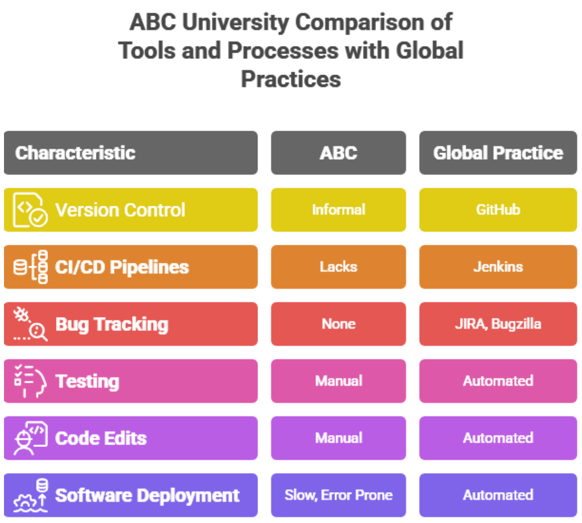

ABC University
Software Maintenance Cycle
Reactive Maintenance
Addressing issues as they arise
Preventive Maintenance
Scheduled actions to prevent issues
Adaptive Maintenance
Modifying software to meet new needs
Perfective Maintenance
Enhancing software performance
Predictive Analytics
Using data to anticipate failures

Figure 4: ABC University Software Maintenance Cycle (Source: Author, 2025)

Tools and Processes
The tools landscape further reinforces this gap in Figure 4. ABC relies mainly on manual code edits, informal version control system and lacks modern tools such as CI/CD pipelines or automated testing suites. Global practice by comparison, demands robust toolchains, including version control systems like Github, continous integration servers like Jenkins and bug tracking software like JIRA, Bugzilla [29]. The lack of automation means that software fixes at ABC are slower to deploy, more error prone and rarely subject to systematic quality assurance.



ABC University Comparison of Tools and Processes with Global Practices

Characteristic	ABC	Global Practice
Version Control	Informal	GitHub
CI/CD Pipelines	Lacks	Jenkins
Bug Tracking	None	JIRA, Bugzilla
Testing	Manual	Automated
Code Edits	Manual	Automated
Software Deployment	Slow, Error Prone	Automated

Figure 4: Tools and Processes with Global Practices (Source: Author, 2025)

Infrastructure Management
ABC University's reliance on on-premises hosting for core systems like the LMS and ERP severely limits scalability and makes the institution vulnerable to peak period failures. By contrast, peer reviewed frameworks advocate cloud based or hybrid infrastructure, which offers elastic scaling, built in redundancy and real time monitoring for performance bottlenecks [17]. The absence of load balancing and real time alerts at ABC explains the significant performance dips during high demand periods such as registration or examination weeks.

Documentation
Partial or inconsistent documentation was another critical weakness. Some units do not maintain formal change request workflows or version histories, creating operational risks when troubleshooting or rolling back updates. In high performing contexts, full documentation of change requests, version histories and rollback plans is standard and often mandated for compliance audits like ISO/IEC 14764 [10].

Staff Training and Capacity
ABC University's limited formal training and ad hoc skill upgrades stand in stark contrast to global norms, where continuous professional development and vendor certified training are routine. This capacity gap weakens the institution's ability to adopt emerging tools and frameworks. Collaborative knowledge sharing communities, which can support local problem solving, are also missing at ABC.

Policy and Governance
At a governance level, the absence of a comprehensive Software Maintenance Policy (SMP) and regular budget allocations further constraint maintenance quality. Global practice links a well defined SMP to broader IT governance frameworks, ensuring that approvals, funding and accountability structures are in place [33].

Outcomes
Collectively, these factors explained the frequent downtimes and persistent user dissatisfaction with system reliability and speed. Whereas mature institutions strive for greater than 99% uptime and continuously monitor user experience, ABC's maintenance culture does not yet support such standards.

Generally, the comparative mapping in the table 6 above, clarifies that ABC University's practices are typical of many private institutions in resource constrained environments. The over reliance on reactive maintenance, lack of automation and underinvestment in human capacity are not unique to ABC but reflect systemic challenges faced by private universities in Nigeria and similar contexts. The practical implication is that piecemeal improvement such as occasional training workshops are unlikely to yield sustained gains unless embedded within a broader strategy that includes clear policy frameworks, dedicated budget lines for maintenance and upgrades with adoption of automated tools for monitoring, testing and version control system. Regular skills development linked to evolving best practices.

CONCLUSION AND RECOMMENDATIONS

Conclusion
This study has investigated and evaluates the applicability of standard software maintenance techniques at ABC University Nigeria. The results indicated that while corrective maintenance was the dominant approach, adaptive and preventive maintenance are limited, largely manual and lack structured automation. Consequently, system uptime

was frequently compromised, staff confidence remains moderate and institutional support structures are inconsistent.

These findings align with broader trends in developing country context, where constrained budgets, limited technical capacity, poor documentation often hamper effective maintenance [5]. When compared to global best practices, ABC's current approaches fall short in leveraging modern tools such as automated testing frameworks, CI/CD pipelines and cloud based infrastructure management.

Recommendations

To close these gaps as discussed in this research work, the following actionable recommendations are proposed;

i. Investment in preventive schedules and automation is key. Develop and implement regular preventive maintenance cycles, adopt automated test suites and monitoring tools to detect and resolve issues proactively, minimizing downtime and reactive fixes.
ii. Modular code designs that allow easier adaptation to change requirement and integration of new technologies has to be adopted using reengineering critical systems by leveraging on open source tools for version control and deployment pipelines.
iii. Conduct regular training workshops focused on modern software maintenance tools such as CI/CD pipelines, automated bug tracking and Git based version systems for enhancing staff capacity. Establishment of partnerships with technology vendors to provide ongoing certification and practical training is important.
iv. Software Maintenance Policy (SMP) must be develop comprehensively to align with institutional goals and IT governance standards by ensuring clear budgetary allocations and approval workflows for maintenance activities.
v. This case study needs to be replicated in other private, states, and federal owned universities to compare contexts and validate findings.
vi. Conduct a detailed cost benefit analysis of adopting automation and cloud services, providing evidence for policy and budgetary decisions.

REFERENCES

[1] Sommerville, Ian. *Software Engineering*. 10th ed. Boston: Pearson, 2015.

[2] Ahmad, Muneer, Ijaz A. Malik, and Bashir Ahmad. "A Systematic Review of Software Maintenance Models in Developing Countries." *International Journal of Computer Science* 15, no. 3 (2017): 42–50.

[3] Osei-Bryson, Kweku-Muata, and Michael Kojo. "IT Governance and Software Maintenance: A Developing Country Perspective." *Journal of Information Systems in Developing Countries* 57, no. 1 (2021): 14–22.

[4] Shakil, Kashish A., Mansaf Alam, and Shuchi Sethi. "Cloud Computing Infrastructure in Higher Education: Opportunities and Challenges." *Procedia Computer Science* 56 (2015): 722–729.

[5] Aniwanage, S. P., T. R. Rajakaruna, and H. B. Madurapperuma. "Portal Maintenance Automation for University E-Governance." *IEEE Access* 9 (2021): 110327–110339.

[6] Spinellis, Diomidis. "Version Control Systems: A Case Study." *IEEE Software* 29, no. 6 (2012): 56–62.

[7] Kitchenham, Barbara, and Stuart Charters. "Guidelines for Performing Systematic Literature Reviews in Software Engineering." *EBSE Technical Report* EBSE-2007-01. Keele University, 2007.

[8] Kan, Stephen H. *Metrics and Models in Software Quality Engineering*. 2nd ed. Boston: Addison-Wesley, 2003.

[9] Alenezi, M., A. Hussain, and S. Alshammari. "A Survey on Software Maintenance Challenges and Solutions." *International Journal of Advanced Computer Science and Applications* 9, no. 12 (2018): 92–100.

[10] Kruchten, Philippe. *The Rational Unified Process: An Introduction*. 3rd ed. Boston: Addison-Wesley, 2004.

[11] IEEE. *IEEE Standard for Software Maintenance*. IEEE Std 14764-2006. New York: IEEE, 2006.

[12] Pressman, Roger S. *Software Engineering: A Practitioner's Approach*. 8th ed. New York: McGraw-Hill, 2014.

[13] Rajlich, Vaclav T. "A Model for Change Propagation Based on Graph Rewriting."

Software Maintenance Research 36, no. 4 (2015): 537–550.

[14] Mockus, Audris, and David M. Weiss. "Global Software Maintenance Practices." *Empirical Software Engineering* 21, no. 6 (2016): 2569–2607.

[15] Nnadozie, Samuel C., and Janet I. Anyanwu. "ICT Infrastructure and E-Learning Adoption in Nigerian Private Universities." *African Journal of Information Systems* 12, no. 1 (2020): 15–26.

[16] Fagbemi, Oluwatoyin O., and Tunde Ojo. "Software Quality Challenges in Nigerian Higher Education." *Nigerian Journal of Information Technology* 18, no. 2 (2019): 25–35.

[17] Musa, Peter H., and Rita A. Egwali. "Challenges and Strategies of Information Systems Maintenance in Nigerian Universities." *International Journal of Computer Applications* 181, no. 22 (2018): 40–45.

[18] Oluwatosin, Michael O., and Maryam A. Adepoju. "Evaluation of E-Governance Portals in Nigerian Private Universities." *Journal of e-Government Studies and Best Practices* 2017 (2017): 1–12.

[19] International Organization for Standardization (ISO). *ISO/IEC 14764: Software Engineering—Software Maintenance*. Geneva: ISO, 2006.

[20] Pigoski, Thomas M. *Practical Software Maintenance: Best Practices for Managing Your Software Investment*. New York: Wiley, 1997.

[21] Kansal, Abhishek, and Saurabh Sharma. "Software Maintenance Challenges in Modern Application Development." *International Journal of Computer Trends and Technology* 56, no. 1 (2018): 1–7.

[22] Zhang, Hong, and Zhenyu Chen. "Continuous Integration and Continuous Deployment in Software Development: A Case Study." *Journal of Systems and Software* 167 (2020): 110597.

[23] Kaur, Navneet, and Parminder Kaur. "A Review on Software Maintenance Activities and Tools." *International Journal of Computer Applications* 171, no. 7 (2017): 1–6.

[24] Basili, Victor R., Lionel C. Briand, and Walcélio L. Melo. "A Validation of Object-Oriented Design Metrics as Quality Indicators." *IEEE Transactions on Software Engineering* 22, no. 10 (1996): 751–761.

[25] Mohagheghi, Parastoo, and Reidar Conradi. "Quality, Productivity and Economic Benefits of Software Reuse: A Review of Industrial Studies." *Empirical Software Engineering* 12, no. 5 (2007): 471–516.

[26] Tarus, John K., Zhendong Niu, and Bakhti Khadidja. "A Survey of Recommender Systems in E-Learning." *International Journal of Learning Technology* 11, no. 3 (2016): 254–272.

[27] Ugwu, Chika F., and Ifeanyi E. Odoh. "An Assessment of ICT Maintenance Culture in Nigerian Universities." *Journal of Computing and ICT Research* 13, no. 2 (2019): 31–42.

[28] Halpin, Terry. *Information Modeling and Relational Databases*. 2nd ed. Burlington, MA: Morgan Kaufmann, 2008.

[29] Tsai, Wei-Tek. "Software Maintenance and Software Evolution." *Encyclopedia of Software Engineering*. John Wiley & Sons, 2002.

[30] UniversityCube.org. "Best Practice Guides for University IT Infrastructure." Accessed March 5, 2025. https://universitycube.org/it-guides

[31] arXiv.org. "Open Research in Software Maintenance and Quality Assurance." Accessed March 5, 2025. https://arxiv.org

[32] Oluwafemi, Kehinde J., and Adeyemi I. Adebayo. "Evaluating Maintenance Performance of Nigerian University Portals." *West African Journal of Computer and Information Systems* 7, no. 1 (2020): 45–54.

[33] Ogunlade, David O. "Policy Gaps in Software Governance in Nigerian Tertiary Institutions." *African Journal of Management Information Systems* 5, no. 1 (2022): 11–22.