WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Development and Delivery of Defectless Banking Payment Softwares: A Development of Conceptual Framework

Magesh G

Department of Business Administration, Aligarh Muslim University.

ABSTRACT:

Payments landscape has seen heaps of changes in the last 2 decades, The Paper processing to electronic and the digital transformation has led to huge dimensional change in the software development and the face of Banking Industry. With over 22 years of experience in the Payments World involving in various roles at SWIFT Operations, Product Development and Implementation and being part of the Technology, I have observed, the importance of defectless software in the banking industry, emphasizing the critical role it plays in maintaining customer trust and protecting sensitive financial data. It then highlights the complexity of banking payment software, considering factors such as intricate business rules, regulatory compliance requirements, and integration with various systems and channels. Coordinating and integrating the diverse systems and technologies within this ecosystem poses significant challenges, including ensuring interoperability, data integrity, and adherence to industry standards and regulations. Further it delves into the specific issues and challenges encountered during the development and delivery phases. These include requirements management, security considerations, robust testing methodologies, compatibility with diverse platforms and devices, and adherence to evolving industry standards. Additionally, the paper explores the impact of time-to-market pressures, cost constraints, and the need for agile development practices on ensuring defectless software. It directly impacts customer satisfaction, regulatory compliance, and the overall reputation of the financial institution. It highlights the importance of proactive management approaches to mitigate potential issues and challenges throughout the software development lifecycle.

The Primary objective of this study is to investigate and analyze the key areas where the more effective the error detection the greater the savings in the development and maintenance cost over the life cycle of the product. The study also finds the consensus on the causes of Business requirements, System Design considerations for the software development and System administration required for the development using the technology.

The outcome of the research will help the product companies, Bankers and practitioners to explore the possible challenges in Payments banking softwares. Also the study will help the product companies to concentrate on the challenging areas of the Payments softwares to better implement and upgrade the product time to time to embrace the new gen technology in future. Also methods to improve the quality of payments software product during implementation.

Keywords – Payment software, Business Requirement, Design Management, System Administrators, Quality Certification, Business Acceptance, Budget, Support, Scalability, Competition.

In summary, this paper provides valuable insights and guidance from a management standpoint for the development and delivery of defectless banking payment software. It highlights the challenges faced in managing complex software projects and offers strategies to address these challenges effectively. By adopting proactive management approaches and prioritizing quality, financial institutions can successfully navigate the challenges and ensure the development and delivery of defectless banking payment software.

Introduction

The convergence of product and service-oriented firms within the banking domain has led to more hybrid, complex organizational models. Today's banking enterprises are increasingly focused on delivering customer-centric value propositions, balancing cost efficiency with innovation. There's a clear shift toward outcome-driven strategies, powered by data-driven decision-making becoming core to daily operations. As the sector grows more competitive, the challenge lies not only in acquiring new customers but also in fostering long-term retention through consistent, high-quality service delivery. This demands smarter resource allocation, optimized workforce utilization within budgetary constraints, and an unyielding drive toward operational excellence. At the heart of all this is the customer's expectation for defect-free, reliable products—solutions that function seamlessly from the start and earn lasting trust through performance and precision

Topic & relevance

This study aims to establish a theoretical framework that empowers organizations to deliver defect-free software by design.

The findings are intended to equip both management and product leadership with actionable insights to proactively reduce defect occurrences, optimize resource allocation, and lower long-term maintenance costs.

The framework will serve as a strategic tool, linking quality assurance practices with business objectives such as customer satisfaction, regulatory compliance, and cost efficiency.

It will also help drive consistency across development processes, enhance stakeholder confidence, and support continuous improvement initiatives across the software lifecycle.

Research Gap

The specific objectives of the research are:

Develop a framework for the management to identify the effectiveness of the independent variables such as Business Requirements, Design Management, System Administrators and their impact on Dependent Variables like Quality Certification (defectless) , Budget based Support and Business Acceptance , Scalability of the Product and Competition.

Thematic Analysis of Literature

Business Requirements:

Khan, I. A., and Kumari, D. (2021) critically assess software development practices in small-scale business applications, highlighting the overlooked importance of the analysis phase as foundational to software quality. They argue that reliance on informal methods, lack of tailored approaches, and incomplete documentation—especially in low-resource environments—undermines project outcomes. Key insights include: errors in analysis ripple through later stages; active client participation is crucial; missing BRD, SRS, and RTM documents cause rework; and security risks are often ignored. The authors advocate structured analysis and documentation to ensure maintainability, accuracy, and defect-free development.Azmat Ullah and Richard Lai (2013) argue that requirements engineering (RE) is essential for aligning IT systems with business goals. They highlight a two-way relationship where organizational context shapes RE activities, and RE, in turn, informs business strategy. Rather than being a purely technical task, RE is presented as a governance tool that ensures IT solutions precisely meet business needs. This is critical in high-risk sectors like payment software, where poor requirements can lead to defects and compliance issues. The authors advocate for a structured, context-driven RE approach to achieve alignment and quality outcomes.Nikolaos A. Panayiotou, Sotiris P. Gayialis, Nikolaos P. Evangelopoulos, and Petros K. Katimertzoglou (2014) highlight the strategic value of using a structured requirements engineering (RE) framework in ERP system development. Their study combines technology-focused and process-oriented methods to improve requirements analysis and implementation. By using business process modeling techniques, the framework helps define clear functional specifications and strengthens overall requirements management. This enables project managers to better align ERP systems with business needs, ensuring more accurate delivery and minimizing defects throughout the development lifecycle.Keith Ellis and Daniel M. Berry (2013) emphasize that mature Requirements Definition and Management (RDM) practices are essential for the success of large commercial application (CA) projects. Based on insights from global enterprises, they find a strong link between RDM sophistication and project success. Rather than a procedural task, RDM acts as a strategic quality control tool—especially in sensitive areas like payment software. High RDM maturity helps reduce ambiguity, control scope, and prevent errors, ultimately leading to compliant, defect-free systems aligned with business goals.Ahmad Shaharudin Abdul Latiff, HaryaniHaron, and Muthukkaruppan Annamalai (2017) describe domain ontologies as structured models that capture key concepts and relationships within specific knowledge areas. In managing payment software, they emphasize that these ontologies help unify understanding among stakeholders, reducing ambiguity in requirements and improving system interoperability. By formalizing knowledge, ontologies enhance communication between business and technical teams and support early defect prevention across the development lifecycle—from requirements through to deployment—ultimately ensuring higher quality outcomes.

Design Management

Walter Brenner, Falk Uebernickel, and Thomas Abrell (2016) present Design Thinking as a holistic approach made up of mindset, process, and toolbox—each essential for fostering innovation and reducing software delivery risks. The mindset promotes understanding user needs and iteratively prototyping to catch defects early. The process includes both a detailed innovation cycle (Define, Needfinding, Ideate, Prototype, Test) and broader governance for milestone-based validation. As a toolbox, it draws from multiple disciplines to support creative problem-solving. Together, these elements help organizations deliver user-focused, defect-free software with improved alignment and reduced uncertainty.Justus D. Naumann and Shailendra Palvia (1982) emphasize that choosing the right systems development methodology (SDM) is a complex decision with significant business impact. This choice is especially crucial in areas like digital payments, where software must be defect-free. They propose a quantitative framework that starts by identifying key SDM functions, then uses the Delphi technique to prioritize them based on expert consensus. This method offers managers a structured, evidence-based way to select SDMs that align with business objectives, improve traceability, reduce ambiguity, and lower defect risks throughout development.Ulla Johansson-Sköldberg, Jill Woodilla, and MehvesÇetinkaya (2013) critically assess Design Thinking, emphasizing that its meaning varies widely depending on context. While often celebrated in management as a driver of innovation, they caution against overly simplistic portrayals. In settings like payment software, where reliability and compliance are vital, the authors argue for a nuanced application. They advocate adapting Design Thinking with

domain-specific awareness, balancing its creative tools and mindset with rigorous engineering, regulatory standards, and measurable quality—ensuring that innovation does not compromise precision or accountability. Jeanne Liedtka (2014) presents Design Thinking as a powerful innovation strategy grounded in understanding user needs. She explains that this approach, especially valuable for delivering defect-free products, encourages ongoing refinement through empathy and iteration. From a management standpoint, Design Thinking isn't just about creativity—it's a strategic tool for staying competitive in fast-changing markets. By embedding it into development processes, teams can better identify user pain points, reduce ambiguity in requirements, and design solutions that proactively prevent defects while maximizing customer satisfaction and long-term value.

System Administrator

Suma. V. and T. R. Gopalakrishnan Nair (2012) stress that achieving near-zero-defect software is vital for stable operations and user confidence. Their proposed Four-Step Approach Model of Inspection (FAMI), built into the V-model of software development, enhances quality assurance through structured inspections and metrics like Depth of Inspection (DI) and Inspection Performance Metric (IPM). By applying Bayesian analysis, FAMI enables data-driven defect prevention. For managers, this model offers a proactive way to reduce post-release issues, ensure service-level compliance, and strengthen trust by aligning software quality with business resilience.Tim Speed and Juanita Ellis (2003) highlight the importance of enterprise-level security policies in maintaining system integrity and reliability. They stress that for defect-free product delivery—especially in critical areas like financial software—organizations need robust frameworks covering network protection, secure design, audits, and incident response. The authors also point out that strong security governance supports legal compliance through structured reviews. For management, these measures are not optional—they're vital for reducing vulnerabilities, preventing post-deployment defects, and sustaining customer trust in high-risk digital environments.Souppaya, Scarfone, and Dodson (2022) highlight critical gaps in secure software development despite existing literature. They note that major SDLC models (e.g., Agile, Waterfall, DevOps) lack built-in security practices, leading to vulnerability gaps. The absence of a unified language further complicates implementation across industries. Many models focus on inputs and processes rather than measurable security outcomes. Their proposed framework—Prepare, Protect, Secure, Respond—is risk-based, tool-agnostic, and emphasizes early integration ("shift left") of security into development. They also stress the importance of automation and traceability to ensure scalable, consistent, and auditable secure development practices.

Quality Certification

Jingjun (David) Xu, IzakBenbasat, and Ronald T. Cenfetelli (2013) extend Wixom and Todd's (2005) work by introducing the 3Q Model—System Quality (SysQ), Information Quality (IQ), and Service Quality (SQ)—to explain how these elements shape user adoption in e-services. They show that SysQ impacts IQ and SQ, and IQ further influences SQ. For digital product teams, especially in defect-sensitive environments, managing all three qualities is critical. This model guides organizations in delivering reliable, user-focused platforms by aligning technical performance, content accuracy, and service responsiveness to meet user expectations and reduce failure rates.Andriy Zapechelnyuk (2020) explains that since consumers often struggle to assess a product's true quality, they rely on external cues—such as third-party certification—for reassurance. This is especially important in high-assurance sectors like payment software, where trust hinges on proven reliability and security. Certification provides independent confirmation that a product meets established quality and regulatory standards. For management, this acts as both a credibility booster and a visible signal of compliance and performance, helping to reduce ambiguity, inspire stakeholder trust, and support the delivery of defect-free, dependable systems.Marina Godinho Antunes, Joaquín TexeiraQuirós, and Maria Rosário Justino (2018) emphasize Total Quality Management (TQM) as a key driver of organizational excellence, especially for delivering defect-free software. TQM enhances efficiency, agility, and competitiveness by aligning internal processes with customer needs. It promotes a culture of continuous improvement, reducing defects across the product lifecycle. For management, TQM offers a structured approach to embed quality into every phase of delivery—improving performance and building long-term differentiation in fast-changing, customer-focused markets. It positions quality not as a task, but as a core strategic function.Correia, J. P., and Visser, J. (2008) propose a certification method based on a layered model of software quality aligned with ISO/IEC 9126. This model evaluates key attributes like functionality, reliability, usability, and maintainability—essential for delivering defect-free software. They also reference ISO/IEC 14598, which offers a structured process for assessing quality across the software lifecycle, from development to maintenance. Together, these standards help organizations build strong quality governance, enforce measurable evaluation criteria, and proactively manage risks before they impact end users—especially in high-stakes software environmentsKishan Patel (2022) critically reviews existing literature on Quality Assurance (QA) in Software Development Life Cycle (SDLC) models and identifies key gaps, including poor integration of QA, limited focus on human and organizational factors, and neglect of maintenance-phase QA. He emphasizes that QA must span all SDLC phases to ensure defect prevention and compliance. Agile methods enable adaptive QA, while traditional models offer structure. Patel outlines best practices such as early stakeholder involvement, automated testing, and ongoing maintenance QA. He concludes that software quality is multidimensional—combining functionality, usability, reliability, and organizational culture for sustained performance.Galin (2004) outlines nine key root causes of software defects, spanning human, process, and technical issues. These include incomplete requirements, poor communication, and deviations from specifications. He highlights flaws in logic and code, inconsistent standards, weak testing, procedural errors, and outdated documentation as critical quality threats. To minimize defects and rework, Galin emphasizes proactive leadership in enforcing standards, improving communication, investing in testing, and ensuring rigorous validation throughout the development lifecycle. His framework positions defect prevention as a shared organizational duty, anchored in structured practices and continuous quality oversight.

Competent Resource

Mr. S. Thowseaf and Dr. M. AyishaMillath (2016) emphasize that skilled human resources are vital for delivering defect-free software. They argue that in today's digital landscape, technical competence, proactive planning, and quality awareness drive business continuity and competitiveness. Rather than relying solely on post-deployment fixes, the authors stress prevention through early quality intervention. Introducing the concept of "Zero Defect Therapy," they frame it as a strategic philosophy to raise quality standards. Their findings highlight that success depends on early-phase planning and skilled teams, making talent investment a core driver of reliable, cost-effective software outcomes.Suma. V. and T. R. Gopalakrishnan Nair (2010) highlight defect prevention as a vital yet often overlooked element of software quality assurance. Their study of five diverse projects shows a strong link between software quality and team capability, with top-performing teams eliminating up to 99.75% of defects through early testing and structured inspections. They argue that defect prevention is a strategic leadership choice, driven by investment in skilled, quality-focused teams. For management, this means prioritizing human capital and disciplined practices to boost product integrity, delivery speed, and long-term cost efficiency.Otero et al. (2009) critically examine the literature on personnel assignment in software development, highlighting key gaps. They argue that current methods lack systematic, skill-based frameworks and often rely on intuition or narrow metrics. Most practices prioritize a single dominant skill, overlooking engineers' broader competencies, which can lead to mismatches and longer training periods. Existing models fall short by ignoring holistic skill evaluation and task complexity. The authors advocate for relationship-based models that consider how proficiency in one skill can ease learning in others—offering a more balanced and effective approach to resource allocation.

Technology Usage

Wynne W. Chin, Norman Johnson, and Andrew Schwarz (2008) critique existing uses of the Technology Acceptance Model (TAM) for relying on traditional measurement methods that weaken the validation of its constructs—known as nomological validity. This shortfall can lead to poor user adoption strategies and potential defects, especially in usability. To address this, the authors introduce the "fast form," a semantic differential scale designed to more accurately assess TAM's core elements. For user-focused domains like financial software, this improved evaluation supports better alignment with user expectations, reducing resistance and promoting defect-free product outcomes.Viswanath Venkatesh, James Y. L. Thong, and Xin Xu (2012) expand the Unified Theory of Acceptance and Use of Technology (UTAUT) to study how consumers engage with tech solutions. Their extended model, tailored to consumer contexts, helps organizations predict user behavior, design intuitive features, and minimize friction in adoption. Especially in sectors like digital payments, this framework supports the delivery of reliable, user-focused software by aligning functionality with expectations—ultimately reducing defects and enhancing user trust from the outset.Dr. Winston W. Royce (1987) criticized traditional linear software development models—where analysis is followed by coding—as overly simplistic for complex systems. Drawing from experience, he emphasized that real-world development involves multiple interdependent phases like requirements, design, testing, and operations, which are often undervalued. Royce was skeptical of idealized models that ignore iterative processes and feedback loops. His critique challenged prevailing practices of his time and called for more structured, realistic approaches. Ironically, his warning evolved into what is now known as the Waterfall Model—though he originally presented it as a caution against rigid, oversimplified methods.

Regulator

Mustapha, O. T. Arogundade, Sanjay Misra, RobertasDamasevicius, and RytisMaskeliunas (2018) highlight compliance management as a major challenge in sectors like financial software, where failure to meet diverse regulations can lead to delays, risks, and penalties. While many organizations use compliance techniques, the authors identify a lack of comprehensive reviews on their contexts, strengths, and impact on product quality. They call for an integrated framework to help teams align compliance strategies with defect prevention goals, enabling better decision-making and risk management in regulated software environments.Obeng et al. (2024) critically assess FinTech compliance literature, noting key issues such as fragmented perspectives, regulatory lag, lack of global harmonization, and insufficient practical frameworks. They emphasize that while technologies like AI, blockchain, and big data improve compliance through automation and transparency—especially in AML and KYC processes—significant challenges remain. These include outdated systems, cybersecurity risks, and uncertain regulations. The authors stress the importance of collaborative models like regulatory sandboxes and public–private partnerships to bridge the gap between innovation and effective oversight, advocating for more structured, outcome-based compliance frameworks.

Risk

Krishan Kumar and Mohit Mittal (2013) explore how digital transformation in banking—fueled by IT—has boosted customer convenience and efficiency but also intensified risks like cyber threats and data breaches. As banks rely more on digital platforms, ensuring software quality includes strong defenses against fraud and disruptions. The authors caution that while tools like cloud computing offer benefits, they also increase vulnerability. Thus, they advocate for proactive risk management and built-in cybersecurity from the outset—key to delivering secure, defect-free services and maintaining trust in an increasingly digital financial ecosystem.Mehmood Ali, Muhammad Ali Khan, and Muhammad Ahmed Kalwar (2021) explore how digital transformation reshapes key organizational areas like decision-making, governance, and marketing—especially in the financial sector, where software quality is vital. While technology boosts efficiency, it also heightens risks tied to cybersecurity, compliance, and data privacy. The authors argue that these challenges are not just technical but strategic, with potential impacts on trust and performance. They call for risk management frameworks integrated into software development to build resilient systems that ensure defect-free, secure, and compliant financial services from the ground up.Hu et al.(2012)

critically review systemic risk literature, highlighting limitations like the focus on networks over individual banks, lack of dual-risk modeling, and underuse of network science and business intelligence (BI) tools. To address this, they propose a novel framework combining financial theory, BI algorithms, and simulation for stress-testing and capital allocation. Their model identifies vulnerable banks and suggests where capital injections are most effective. A key insight is that during severe market shocks, interbank payment links become more influential than portfolio similarities, underscoring the value of integrated, data-driven tools for targeted regulatory responses.

Budget based Support and Business Acceptance

AnolBhattacherjee and Clive Sanford (2006) examine how external influences—like communication quality and source credibility—affect user acceptance of information technology. Using the Elaboration Likelihood Model, they compare two routes: the central (argument quality) and peripheral (source credibility), both shaping perceived usefulness and user attitudes. These effects vary based on users' IT expertise and job relevance. The study emphasizes that aligning communication, context, and product design is vital for adoption success. For managers, this means going beyond features to ensure user expectations match the product—essential for defect-free, long-term usage.AnolBhattacherjee (2001) makes a key distinction between initial acceptance and continued use of information systems, which is critical for sustaining defect-free digital products. He introduces one of the first IS continuance models, integrating user satisfaction and expectation confirmation. His work shows that product quality must extend beyond launch, focusing on long-term engagement. The study provides a framework to understand why users stop using systems—often due to usability issues or unmet expectations—and emphasizes the need for feedback-driven improvements to enhance both technical quality and lasting satisfaction.Hua (Jonathan) Ye and AtreyiKankanhalli (2018) emphasize the strategic value of involving users in digital innovation, especially to lower development costs and address market uncertainty. They highlight that platforms like iOS and Android, with built-in toolkits and policies, allow early user feedback that helps reduce defects and rework. The study also notes the crucial role of lead users—those with high innovation potential—but observes a lack of research on how user traits and platform features interact. For leaders, the findings stress designing systems that foster early co-creation to improve quality and reliability before launch.Pankaj M. Madhani (2020) highlights how intense competition, high consumer expectations, and uncertainty demand constant transformation, especially in financial services. To succeed, firms must achieve both innovation and flawless execution—balancing cost, quality, and customer satisfaction. Madhani stresses that consistent delivery is only possible through strategic refinement of systems and processes. For those aiming to deliver defect-free products, agile, reliable operations and a culture of continuous improvement are essential. In today's high-stakes environment, operational excellence isn't optional—it's foundational to long-term resilience, trust, and competitiveness in the financial sector. ZhiyiZhuo (2019) highlights that in banking, customer satisfaction and loyalty distinguish top performers from weaker institutions. To support defect-free digital services, banks must align support systems with rising customer expectations by ensuring responsive, transparent, and empathetic service across all channels. Zhuo emphasizes that performance reliability, fairness, and excellence are key to retention. Effective support models should identify hidden pain points, predict failures, and deliver prompt resolutions. For leaders, this approach isn't just operational—it's a strategic asset that builds trust, enhances competitiveness, and drives long-term service quality.

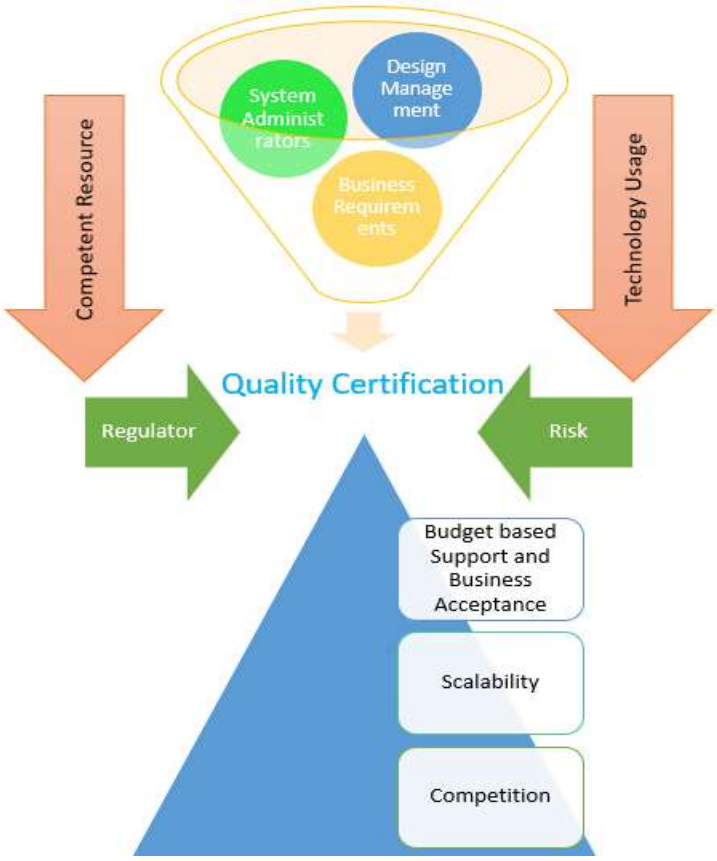
Scalability

Nazar Abbas Saqib and Shahad Talla AL-Talla (2023) tackle blockchain's scalability issue by addressing slow transaction verification speeds. They propose a solution using parallel mining and a Particle Swarm Optimization–Proof of Work (PSO-POW) model, which intelligently selects an optimal mining manager to reduce idle time and boost responsiveness. This method mitigates defects like latency and uneven workloads, improving trust and system integrity. Their evaluation shows marked gains in throughput and efficiency. For leaders in high-transaction sectors like finance, the study highlights proactive design innovation as key to resilient, defect-free blockchain systems.PriyadharshiniMuthukannan, Barney Tan, Felix Ter Chian Tan, and Carmen Leong (2021) explore how the Fintech revolution—especially in Indonesia—is transforming traditional banking into more adaptive models using platformization, decentralization, localization, and democratization. These approaches not only drive innovation but also enable defect-free, scalable financial services by aligning platforms with local needs and minimizing onboarding barriers. The authors present Indonesia's Fintech ecosystem as a replicable model for inclusive, high-quality digital finance. They emphasize the importance of building technically robust and locally attuned systems to maintain trust, resilience, and consistency in underserved, fast-growing markets.Nitin Malhotra (2005) emphasizes the need for structured, process-oriented methods in software development to meet modern demands for scalability and delivery assurance. Drawing on manufacturing practices, he advocates for integrating statistical process control (SPC) to standardize cost, timelines, and quality. Once stability is achieved, organizations can pursue continuous improvement without compromising performance. Malhotra argues that scalability depends not just on technology, but on disciplined engineering and governance. His work supports using rigorous, repeatable processes to ensure defect-free, resilient software—especially in fast-paced environments requiring speed, innovation, and consistent quality.Jayathilake et al. (2011) address gaps in product-level quality control, noting that current practices (e.g., CMMI, Agile) focus too much on process quality. They highlight challenges such as limited adaptability for outsourcing vendors and fragmented toolchains lacking real-time, rule-based evaluations. To tackle this, the authors propose a modular framework with three key components—Commit Manager, Offline Quality Analyzer, and Product Quality Analyzer—for automated quality checks. It supports customization, integrates with standard tools (e.g., Sonar, FxCop), and proved effective in improving awareness and early issue detection during a commercial pilot, though it also revealed training and communication gaps.

Competition

Joseph S. Valacich, Xuequn Wang, and Leonard M. Jessup (2012) highlight how evaluation context affects user perceptions of product features. Their "evaluability hypothesis" shows that users weigh features differently when comparing two products versus reviewing one in isolation—focusing on comparable traits in the former and simpler, obvious features in the latter. This can lead to preference reversals, where user choices shift unexpectedly. For managers, this insight calls for evaluation methods that mirror real usage contexts, helping align design, testing, and marketing with how users actually judge product quality and avoid misinterpreted flaws.Xavier Vives (2020) explains how digital disruption—accelerated after the 2007–2009 financial crisis—has transformed the competitive landscape of banking. Traditional banks now face shrinking margins, strict regulations, and rising competition from FinTech and BigTech. This shift to platform-based, customer-focused models has heightened complexity and risk, leading to potential service defects. With higher expectations for responsiveness and personalization, banks must embed agility, quality assurance, and innovation into their operations. Vives emphasizes that in this volatile environment, defect-free execution is essential for sustaining trust, ensuring service reliability, and remaining competitive in a digital-first market.Dan Ma and Robert J. Kauffman (2012) offer a critical, multidimensional review of SaaS competition literature, highlighting fragmented approaches to pricing, service quality, and vendor differentiation. They note that past research often overlooks key factors such as switching costs, client uncertainty, and horizontal differentiation—like feature fit—especially relevant under SaaS's multitenancy model. To address these gaps, the authors propose a game-theoretic duopoly model that integrates vertical and horizontal differentiation, sampling behavior, and cost-efficiency of quality improvements. Their framework enables more realistic analysis of vendor strategies in increasingly complex SaaS markets.

Conceptual Framework Constructs



Conclusion

As Business Requirements, Design Management, System Administrators, Competent Resource, Technology Usage, are key drivers from the literature study, which formulates for a construct and need to identify the impacts or effects of these constructs for the outcome of Quality Certification which inturn helps the user acceptance with budget , scalability of the product and the competition strength in the market. The prospective and potential to develop and deliver a defect less payment banking software product requires cohesive unit to achieve the outcome. Software development process is not something new and the product management team is aware of the process and steps involved to create a desired outcome of defect free product. The quantum of time spent on the aspects of Business requirements, Design Management , System Administration layer , Competent Resource and Technology Usage will drive efforts and time to prevent for future rectification and maintenance overhead.

Quality certification is a critical component towards a defect free product labelling. These results highlight that for a quality product to be developed and delivered and accepted by Business with the availability of budget and support, Provides a scalable defect less software and emboss a positive competition in the market.

Managerial Implications:

The primary study has resulted in to a comprehensive literature review on Business Requirements, Design Management, System Administration, Quality certification in the area development and delivery of a defectless payment banking software product. The literature study helped to identify factors that act as a determinants for developing a defectless product.

The present study includes development of a comprehensive model which is a major contribution of this research which could be tested for other financial product software in different geographies.

References

Ullah, A., & Lai, R. (2013). A systematic review of business and information technology alignment. *ACM Transactions on Management Information Systems (TMIS)*, *4*(1), 1-30.

Panayiotou, N. A., Gayialis, S. P., Evangelopoulos, N. P., &Katimertzoglou, P. K. (2015). A business process modeling-enabled requirements engineering framework for ERP implementation. *Business Process Management Journal*, *21*(3), 628-664.

Ellis, K., & Berry, D. M. (2013). Quantifying the impact of requirements definition and management process maturity on project outcome in large business application development. *Requirements Engineering*, *18*, 223-249.

Latiff, A. S. A., Haron, H., & Annamalai, M. (2017). Software engineering approach for domain ontology development: a case study of Islamic banking product. *Journal. Info. Retrieval Knowledge Management (JIRKM)*, *3*(2017), 36-53.

Brenner, W., Uebernickel, F., &Abrell, T. (2016). Design thinking as mindset, process, and toolbox: Experiences from research and teaching at the University of St. Gallen. In *Design thinking for innovation: Research and practice* (pp. 3-21). Cham: Springer International Publishing.

Naumann, J. D., &Palvia, S. (1982). A selection model for systems development tools. *MIS Quarterly*, 39-48.

Johansson-Sköldberg, U., Woodilla, J., &Çetinkaya, M. (2013). Design thinking: Past, present and possible futures. *Creativity and innovation management*, *22*(2), 121-146.

Liedtka, J. (2015). Perspective: Linking design thinking with innovation outcomes through cognitive bias reduction. *Journal of product innovation management*, *32*(6), 925-938.

Nair, T. G., & Suma, V. (2012). Quality enhancement by implementation of depth of inspection metric and inspection performance metric in software industry. *International Journal of Productivity and Quality Management*, *9*(2), 137-157.

Speed, T., & Ellis, J. (2003). *Internet security: a jumpstart for systems administrators and IT managers*. Elsevier.

Antunes, M. G., Quirós, J. T., & Justino, M. R. (2018). Total quality management and quality certification: effects in organisational performance. *International Journal of Services and Operations Management*, *29*(4), 439-461.

Nair, T. G., & Suma, V. (2010). The pattern of software defects spanning across size complexity. *International Journal of Software Engineering*, *3*(2), 53-70.

Chin, W. W., Johnson, N., & Schwarz, A. (2008). A fast form approach to measuring technology acceptance and other constructs. *MIS quarterly*, 687-703.

Venkatesh, V., Thong, J. Y., & Xu, X. (2012). Consumer acceptance and use of information technology: extending the unified theory of acceptance and use of technology. *MIS quarterly*, 157-178.

Xu, J., Benbasat, I., &Cenfetelli, R. T. (2014). The nature and consequences of trade-off transparency in the context of recommendation agents. *MIS quarterly*, *38*(2), 379-406.

Zapechelnyuk, A. (2020). Optimal quality certification. *American Economic Review: Insights*, *2*(2), 161-176.

Wixom, Barbara H., and Peter A. Todd. "A theoretical integration of user satisfaction and technology acceptance." *Information systems research* 16, no. 1 (2005): 85-102.

Maskeliunas, R. AM Mustapha, OT Arogundade, Sanjay Misra, RobertasDamasevicius&.

KUMAR, K., & MITTAL, M. E-Banking Security and Challenges in India: A Survey.

Bhattacherjee, A., & Sanford, C. (2006). Influence processes for information technology acceptance: An elaboration likelihood model. *MIS quarterly*, 805-825.

Bhattacherjee, A. (2001). Understanding information systems continuance: An expectation-confirmation model. *MIS quarterly*, 351-370.

Ye, H. J., &Kankanhalli, A. (2020). Value cocreation for service innovation: Examining the relationships between service innovativeness, customer participation, and mobile app performance. *Journal of the Association for Information Systems*, *21*(2), 8.

Madhani, P. M. (2020). Lean six sigma deployment in retail industry: Enhancing competitive advantages. *IUP Journal of Business Strategy*, *17*(3).

Zhuo, Z. (2019). Research on using Six Sigma management to improve bank customer satisfaction. *International Journal of Quality Innovation*, *5*(1), 3.

Saqib, N. A., & AL-Talla, S. T. (2023). Scaling up security and efficiency in financial transactions and blockchain systems. *Journal of Sensor and Actuator Networks*, *12*(2), 31.

Muthukannan, P., Tan, B., Tan, F. T. C., & Leong, C. (2021). Novel mechanisms of scalability of financial services in an emerging market context: Insights from Indonesian Fintech Ecosystem. *International Journal of Information Management*, *61*, 102403.

Malhotra, N. (2005). An Integrated Six-Sigma and CMMI framework for software process improvement.

Valacich, J. S., Wang, X., & Jessup, L. M. (2018). Did I Buy the Wrong Gadget? How the evaluability of technology features influences technology feature preferences and subsequent product choice. *MIS Quarterly*, *42*(2), 633-644.

Vives, X. (2020). Digital disruption in banking and its impact on competition. *Organisation for Economic Co-operation & Development (OECD)*.

Khan, I. A., & Kumari, D. (2021). The role of analysis phase of SDLC for small scale business application-a review. *International Journal of Humanities, Engineering, Science and Management*, *2*(01), 63-75

Souppaya, M., Scarfone, K., & Dodson, D. (2022). Secure software development framework (ssdf) version 1.1. *NIST Special Publication*, *800*(218), 800-218.

Patel, K. (2022). An Analysis of Quality Assurance Practices Based on Software Development Life Cycle (SDLC) Methodologies. *J. Emerg. Technol. Innov. Res*, *9*(12), g587-g592.

Otero, L. D., Centeno, G., Ruiz-Torres, A. J., & Otero, C. E. (2009). A systematic approach for resource allocation in software projects. *Computers & Industrial Engineering*, *56*(4), 1333-1339.

Hu, D., Zhao, J. L., Hua, Z., & Wong, M. C. (2012). Network-based modeling and analysis of systemic risk in banking systems. *MIS quarterly*, 1269-1291.

Royce, W. W. (1987, March). Managing the development of large software systems: concepts and techniques. In *Proceedings of the 9th international conference on Software Engineering* (pp. 328-338).

Ma, D., & Kauffman, R. J. (2014). Competition between software-as-a-service vendors. *IEEE Transactions on Engineering Management*, *61*(4), 717-729.

Jayathilake, D., Yaggahavita, H., Senanayake, U., Elvitigala, C., &Sriyananda, D. (2011, December). A scalable product quality verifier framework for a outsourcing supplier. In *2011 IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE)* (pp. 390-395). IEEE.

Obeng, S., Iyelolu, T. V., Akinsulire, A. A., &Idemudia, C. (2024). The transformative impact of financial technology (FinTech) on regulatory compliance in the banking sector. *World Journal of Advanced Research and Reviews*, *23*(1), 2008-2018.