مجلة جامعة بني وليد للعلوم الإنسانية والتطبيقية
Bani Waleed University Journal of Humanities and Applied Sciences
تصدر عن– جامعة بني وليد – ليبيا
Website:https://jhas-bwu.com/index.php/bwjhas/index
المجلد التاسع ـ العدد الخامس ـ 2024 ـ الصفحات (405-415)
ISSN3005-3900

تحسين جودة البرمجيات المحلية من خلال تحسين كفاءة العمليات في البنوك الليبية دراسة بمدينة درنة

منصف محمد المنصوري[1*]، هشام صالح المسماري[2]، مصطفى محمود سلامة[3]،
ساره ساسي خليفة[4]، سلمى محمد البرغثي[5]
[123] قسم الحاسوب، كلية العلوم التقنية درنة، ليبيا
[4] قسم الحاسوب، المعهد العالي للعلوم والتقنية ترهونة، ليبيا
[5] قسم الحاسوب، كلية العلوم، جامعة درنة
mansaf.elmansori@ctsd.edu.ly

Improving Local Software Quality Through Process Efficiency Improving in Libyan Banks: Case Study of Derna City

Mansaf M. Elmansori[1*], Hesham Saleh Almssmari[2], Mostafa M. Salama[3]
Sarah Sasi Khaleefah[4], Salma Albargathi[5]
[1 2 3] Department of Computer, College of Technical Science-Derna, Libya.
[4] Department of Computer, Higher Institute of Science and Technology – Tarhuna, Libya.
[5] Department of Computer, Faculty of Science, University of Derna, Libya.

تاريخ الاستلام: 28-10-2024 تاريخ القبول: 22-11-2024 تاريخ النشر: 10-12-2024

الملخص:

في حين نمت مشروعات البرمجيات بشكل كبير، فقد لاحظنا أن عمليات البرمجيات القوية والفعّالة يمكن أن تدفع منتج البرمجيات إلى انتاج الجودة العالية. ترتبط البرمجيات عالية الجودة ارتباطًا وثيقًا بالعملية المستخدمة في تطوير البرمجيات. يعد تحسين عمليات البرمجيات نشاطًا مهمًا يبدأ عندما تنوي مؤسسة ما تعزيز أو تنقية قدرة عملياتها الجارية. في سياق نموذج نضج القدرة (CMM)، تعاملت مؤسسات متنوعة مع منتجات برمجيات مختلفة لتحسين عمليات البرمجيات بطرق متنوعة. وقد حققت كل منها مؤشرات أداء رئيسية ذات فائدة كبيرة يمكن استخدامها لتقييم كفاءة عمليات البرمجيات. ستؤدي نتائج هذا القياس الي تطوير المزيد من التعديلات والتحسينات في العمليات. لذا يهدف نموذج تحسين عمليات البرمجيات SPI إلى تزويد مؤسسات تطوير البرمجيات بآليات لتقييم عملياتها الحالية. حيث أن هذه الدراسة بادرت لمساعدة مؤسسات البرمجيات المحلية، المتمثلة في أقسام تقنية المعلومات متوسطة الحجم في البنوك الليبية، لتحقيق النضج والكفاءة في عمليات البرمجيات الخاصة بهم. كشفت هذه الدراسة من خلال الاستطلاع الذي تم في مؤسسات برمجيات محلية عن وجود مجموعة من المشاكل التي قد تؤدي إلى عدم الكفاءة في العمليات بأقسام تقنية المعلومات في البنوك الليبية. كما يهدف هذا إلى تحديد فعالية عمليات البرمجيات المحلية المستخدمة في المؤسسات المحلية وتحديد نقاط قوتها وضعفها في عمليات البرمجيات. ويمكن أن تكون نتائج هذه الدراسة

مفيدة في تحليل المشاكل التي تؤثر على تحسين جودة البرمجيات. حيث يساعد هذا في تحديد المشاكل الحرجة في مؤسسات البرمجيات لتحديد أولوياتها وتحسينها.

الكلمات المفتاحية: مشروعات البرمجيات، جودة البرمجيات، عمليات البرمجيات

Abstract

While the software project has grown huge, it's noticed that strong and effective software processes can drive the software product to high quality. High-quality software is inextricably linked to the process used in developing the software. Improving software processes is a crucial activity that starts when an organization seeks to refine or enhance the effectiveness of its existing processes. In the context of the Capability Maturity Model (CMM), organizations with diverse needs and distinct software products have adopted various approaches to software process improvement. Each has achieved significant benefits, highlighting key performance indicators that can be utilized to assess the efficiency of software processes. The measurement outcome would develop further process adjustments and improvements. Software Process Improvement SPI aims at equipping software development organizations with mechanisms for the evaluation of their existing processes. This study is initiated to help local software organizations, referred to as mid-size IT departments in Libyan banks, achieve maturity and efficiency in software processes within their respective organizations. The investigation of the survey study carried out at a local software organization has revealed that there is a set of problems that might lead to inefficiency in the IT departments in Libyan banks. This aims to evaluate the efficiency of the software process utilized by local organizations and to pinpoint its strengths and weaknesses. The results of this study could be useful in analyzing problems that affect the improvement in software quality. The identification of critical problems in software organizations helps them prioritize improvements.

Keywords: software project, software quality, software process

1 Introduction

The software process has been a cornerstone of software engineering, significantly shaping the way software is designed and developed. Over the past three decades, numerous software life-cycle models have emerged, providing valuable frameworks to guide software developers in managing their projects. While these models offer general principles and methodologies, they often fall short in addressing the intricate details and complexities inherent in large-scale software development projects. Despite these limitations, the evolution of these models has brought to light many previously overlooked aspects of the software life cycle. This deeper understanding has, in turn, led to the emergence of a specialized field known as software process engineering, which focuses on analyzing, designing, and improving software processes to enhance efficiency, quality, and adaptability in software development. This new discipline continues to refine our approach to managing the complexities of modern software systems. The organizations that are targeted have to improve their processes to become more effective in terms of higher quality, higher efficiency, and so forth. To be able to improve these processes, they need to have a process that has to follow in the first place. This study represents a significant contribution to the field of software engineering by addressing a persistent challenge: how to achieve an effective and efficient software process. By focusing on this critical issue, the research seeks to provide valuable insights and practical solutions for improving software development practices. Specifically, this survey study aims to identify the root causes of inefficiencies in the software processes commonly used by local organizations. Through an in-depth analysis, the study highlights factors that undermine process quality and suggests actionable strategies to overcome these obstacles. By tackling these issues, the research not only sheds light on the unique challenges faced by local software organizations but also offers recommendations to enhance process efficiency, improve product quality, and foster better software engineering practices. Ultimately, the findings of this study have the potential to guide organizations in refining their methodologies and achieving sustainable improvements in their software processes.

Improving local software quality while increasing process efficiency in banks necessitates a multifaceted approach that integrates software process improvement methodologies with quality management practices. Researches indicate that effective software quality management, which encompasses quality assurance, control, and testing, is crucial for enhancing project delivery efficiency and operational effectiveness as well as improving organizational processes is important to meet business goals effectively and delivering projects faster and with higher quality. ("Improvement in Software Quality", 2023).

Furthermore, improving local software quality through increased process efficiency can be achieved by integrating various methodologies and technologies. The Analytical Hierarchy Process (AHP) has been effectively utilized to measure software quality characteristics such as functional suitability, performance efficiency, and reliability, providing a structured approach to gather stakeholder feedback and automate quality assessments (Sarwosri et al., 2023).

Literature on the subject has dramatically evolved over the years to reflect the evolving nature of software development and the need for organizations to constantly be in a state of flux to cater to emerging demands. According to (Allison, 2005), agility and flexibility are the most important features in SPI activities. The author feels that organisations should, in response to innovation pressures and product upgrades, rapidly respond as would be expected within packaged software production.

Allison's proposal of a model-SPI as an emergent process rather than a deterministic one- emphasizes the fact that their particular context will inherently bind the design and conduction of changes. This perspective is vital in that respect; it moves the focus from strict methodologies to more flexible frameworks that can cope with the running nature of software development. The author underlines that if the commitment to high-added-value products is not present, there is a very real danger that commercial advantages could be lost, underlining once more the close link between process efficiency and product quality.

2 Related works

Dyba (2000) conducted an extensive literature review encompassing 55 software companies. The findings of this research revealed that a focus on measurement is one of six critical factors contributing to the success of process improvement initiatives that rely on software process improvement (SPI) metrics. Sharon Wheeler and Sheryl Duggins (1998) found that there is no universal formula for enhancing the quality of software processes. However, numerous guidelines and methodologies have been proposed by industry experts and quality professionals to assist organizations in achieving process improvements. A study conducted by Danish students in 2001 revealed that while nearly all organizations surveyed had a positive attitude toward software quality management (SQM), 40% (44 organizations) were unfamiliar with SQM standards or software process improvement (SPI) methodologies. The findings highlighted the need for increased research efforts in the fields of SQM and SPI. More importantly, the study emphasized the necessity of raising awareness and providing education about these practices. Many organizations without established SQM or SPI frameworks are unaware of the available standards and methodologies, making it challenging for them to initiate software quality management or process improvement efforts (Kautz, 2001).

The literature on enhancing the quality of local software by improving process efficiency has evolved quite significantly in the last few years, and diverse methodologies and frameworks have been suggested depending on the types of problems that software development teams face. A key contribution to the discussion was made through the extensive bibliometric analysis that was carried out by) Almeida, 2020(of agile software development and presents a comparative study of Scrum, Lean software development, and other agile methodologies. This analysis emphasizes the efficiency of agile practices within a global software development context and points out the relevance of embedding user-centred design principles within agile frameworks.

Research done by (Almeida, 2020) explored systematic literature on the cost of software quality and the use of rapid application development techniques. It critically reviews methodologies and insists that what is needed is a holistic approach: lean thinking combined with agile methods with the view to enhancing software quality, especially in local contexts. Hence, empirical evidence provided a basic reference on how agile methodologies could be adapted in the attempt to improve process efficiency and, consequently, the quality of software.

Significantly, the software quality discipline has gained a lot of steps, especially in recent years, in local software development by increasing the efficiency of processes. (Kokol, 2021) provided a basic understanding of the evolution through a historical and synthetic content analysis of the literature on software quality. This work is seminal in the sense that its holistic approach synthesizes a relatively small dataset, which indicates an exponential growth trend in research publications on software quality. The results, therefore, suggest that there is an increasingly concerned community with the subject at hand, hence one of growing interest in modern software engineering.

In the last decade, the literature on enhancing the quality of local software through bettering process efficiency received considerable input. Among all such works, one important recent study could be considered by (A Schtein 2018), which examines the adoption of Agile methodologies by distributed teams of software development. Among them, this research is specifically relevant because it explains how such increased rates of success in DSD projects translate into social effects, considering that the financial position and independence of software engineers in less developed countries would rise correspondingly.

A study by (Lalband Neelu, 2021) emphasized that improving local software quality involves enhancing quality attributes like performance, reliability, and usability through the Hybrid Agile Quality Parameter Analysis (HAQPA), which integrates strengths from Scrum, Extreme Programming, and Lean Software Development for better process efficiency.

The proposed Blockchain-Based Software Process Improvement (BBSPI) approach enhanced local software quality by reducing costs, time, and resource utilization while facilitating effective knowledge management, thereby encouraging software organizations to adopt process improvement for better quality and budget delivery (Farooq et al., 2021).

Additionally, the application of Business Process Reengineering (BPR) can streamline workflows by eliminating non-value-added activities, leading to operational excellence in manufacturing contexts. While it emphasizes manufacturing processes, the principles can be adapted to improve software quality by streamlining workflows and reducing waste (Wang et al., 2024).

3 Study Motivations

The study motivations are as follows:

1- To understand and determine the current software engineering process practices in local organizations.

2-To identify strengths, major weaknesses, and key areas for local software process improvement.

3-The problems that affect the effectiveness of the software process in local software organizations, and to develop proposals to solve these problems.

4 Study objectives

This study trying to get clear results in response to previous queries, can also determine the targets of the study in the three objectives:

1. To determine the local software process efficiency and range that local organizations use.
2. To identify the strengths and weaknesses of the local software process, the practice degree for process areas (sub-processes) must be determined.
3. Identify the problems that affect the effectiveness of the software process in local software organizations and develop suggestions to solve these problems.

5 The Methodology

This investigation will use the quantitative analysis methodology of data collected from questionnaires. Using the quantitative approach of data analysis, questionnaires were produced to determine the range of local software process efficiency and the strengths and weaknesses of the local software process used by local organizations. This analysis will use statistical charts and outcomes to measure the local software efficiency range.

6 The Software Process

The term "software process" intersects with various related concepts and terminologies, including business process, standard process, software process improvement, and many others. These terms reflect the multifaceted nature of software processes and their alignment with broader organizational workflows. For instance, business processes outline the overarching strategies and operations of a company, while standard processes provide structured guidelines for consistency and quality assurance. Similarly, software process improvement focuses on refining and optimizing the methodologies used to develop, deliver, and maintain software products. Together, these interconnected concepts demonstrate the comprehensive scope of software processes, emphasizing their critical role in achieving efficiency, adaptability, and success in software engineering and organizational management. In this study, the software process is described as a series of interconnected activities that commence with identifying a specific need and culminate in retiring a product that fulfils that need. More comprehensively, it is characterized as a systematic collection of activities, methodologies, practices, and transformations employed by individuals to create and sustain software along with its associated artefacts. These artifacts may include essential deliverables such as project plans, design documentation, source code, test cases, and user manuals. The software process encompasses the entire lifecycle of software development, ensuring that every stage from initial conceptualization to eventual decommissioning adheres to structured practices aimed at delivering high-quality, functional, and maintainable software products. This holistic approach highlights its role in coordinating technical tasks and managing resources to address user needs effectively. Software processes are inherently complex and heavily depend on human judgment and decision-making. Due to the necessity for creativity and subjective assessment, efforts to fully automate these processes have achieved limited success. Computer-aided software engineering (CASE) tools can assist with certain activities within the process. Still, broader automation where software systems take over the creative aspects of design from engineers remains unlikely in the near future. One factor limiting the effectiveness of CASE tools is the vast diversity of software processes. There is no single "ideal" process; many organizations have customized their development methods to leverage their team's unique skills and address the specific requirements of the systems they are building. Despite this diversity, all software processes share some core activities:

1. Software specification: Defining the functionality and operational constraints of the software.
2. Software design and implementation: Creating the software to fulfil the specified requirements.
3. Software validation: Ensuring the software meets the customer's needs and expectations.
4. Software evolution: Adapting the software to accommodate changing customer demands.

.

7 Data Analysis and Results

7.1 Survey study

The empirical part of the study was conducted in the information technology departments of Libyan banks. These departments are responsible for developing and maintaining the family of software products that are referred to as the product in this study.

7.2 Population

This survey study will cover the information technology departments at Libyan banks in Derna City, which are:

1- Aman Bank.
2- National commercial bank.
3- Bank of commerce & development.
4- Jumhouria bank.
5- Sahara Bank.
6- Al-wahda bank branch 1.
7- Al-wahda bank branch 2.
8- NAB bank.
9- Islamic Jumhouria Bank.

7.3 The Survey Study Method

The methods used in the survey study to accomplish the study's goals are predicated on the questionnaires:

The questionnaire document includes inquiries regarding the adoption of key software practices within their organization. These questions are categorized into specific process areas, such as software project planning, software configuration management, and other related domains. The Questionnaire contains 54 questions divided into 9 sets, which comprise all 9 Process Areas; the questions developed were based on the practices governed by the Software Engineering Institute (SEI). The questionnaires were distributed to the participants, and the distribution of forms and collection period was extended to four weeks to obtain a high response rate and to give participants the opportunity to make statements that can be relied upon. Reporting information about participants, there were a total of 74 participants in the survey, out of which 51 responded, the rest 14 did not respond, and nine missing papers. Details are shown in the table (1) below:

Table (1): information about participants in the questionnaire

	Respondents	Non-Respondents	Missed
Number	51	14	9
Percentage	69%	19%	12%

7.4 Study Scale Describing

Study Data were collected through the questionnaire, which consisted of 9 axes, to measure the variables of the study. The scale consists of (54) phrases measured on a scale of "Likert" of three degrees, and this means that each statement in the questionnaire measured three alternatives to answer, according to the following (yes, don't know, no), a measure has been formulated in a positive way, which is given to the participant (3) when the answer is that (yes), while giving the participant degree (1) when the answer is that he (no) do that, while giving the participant degree (2) when the answer is that (don't know). The researchers tested the questionnaire on the vocabulary of the sample to detect any ambiguity in the wording of phrases, modified mutates, amended, the vocabulary answers to the sample and clear, as the words are clear and do not have any ambiguity to the reader it.

7.5 Study scale stability

Conditions of safety measure stability and are known as "a pointer to the degree of precision or control in the measurement process" (Thorndike et al., 1986). To check the stability of the study's scale, Cronbach’s alpha has been applied to calculate the measure of the study's stability (Nunnally and Bernstein, 1994).

$$a = \left[\frac{N}{N-1}\right]\left[1 - \frac{\sum 6_q^2}{6t^2}\right]$$

As: a = stability coefficient   1 = fixed amount.
N = number of questions. ε = Total.

6_q^2 = deviation of each item of the test (degrees of individuals in this item).

$6t^2$ = deviation test as a whole.

Applying this equation using the sample, given the degree of stability, calls for Reliability, as shown in Table (2).

Table (2): stability coefficient of scale

stability coefficient	scale
0.83	effective software process

7.6 Study scale validity

The concept of validity is "the extent to which the examination for the job that was used to perform or to perform the questionnaire for the purpose for which it is" (Salkind, N. J., 2017).

$$v = \sqrt{a}$$

Where: V = validity coefficient.
A = stability coefficient of the scale.

In other words, the validity coefficient is the square root of the stability coefficient, and this equation can be calculated using the ratified standard used in the study. Table (3) shows the validity coefficient derived from the stability coefficient.

Table (3): validity coefficient of scale

validity coefficient	scale
0.91	effective software process

7.7 Measuring Local Software Process Efficiency

After conducting questionnaires on the software process efficiency, data analysis, calculating averages and standard deviations of the answers, and comparing the averages of the answers with an average standard, found that the arithmetic average was (1.93) and the standard deviation (0.27), the least of public scale average (2) but by practice degree of medium, which indicates that there is some software process efficiency in organizations under study. As shown in table (4):

Table (4): illustrates the scope of local software process efficiency

Aim	The Arithmetic Average	Standard Deviation
Software Process Efficiency	1.93	0.27

Looking at the presence of medium practice degree of software process efficiency, so conducted a deeper analysis of process areas, and this shows the rate of efficiency in process areas surrounding the local software. The following table (5) shows the rates:

Table(5): illustrates the practice degree of process areas

No	Process Area	Arithmetic Average	Standard Deviation
1	Requirements management	1.87	0.36
2	Software project planning	1.91	0.29
3	Project monitoring and control	1.88	0.36
4	Software quality assurance	1.92	0.38
5	Organization process focus	1.94	0.40

6	Organization process definition	1.91	0.46
7	Training program	1.92	0.48
8	Integrated software management	1.87	0.47
9	Software product engineering	1.91	0.41

7.8 Identify the Strengths and Weaknesses of the Local Software Process

Through the identification of the general average of the process areas, compared with the scale of the study, consisting of three degrees, we found that all process areas carry the practice degree of the medium, which proves that there is a kind of implementation of the concept of maturity software process in organizations under study. It also consists of formal characterization such as highly implemented, mediumly implemented, and lowly implemented. The following table (6) illustrates the range of implementation in these areas in the local software process:

Table(6): illustrates the range of implementation in these process areas in the local software process.

Process area	Practice degree
Requirements management	Mediumly implemented
Software project planning	Mediumly implemented
Project monitoring and control	Mediumly implemented
Software quality assurance	Mediumly implemented
Organization process focus	Mediumly implemented
Organization process definition	Mediumly implemented
Training program	Mediumly implemented
Integrated Software Management	Mediumly implemented
Software product engineering	Mediumly implemented

From the previous table, it transpired there, drawn in the degree of practice between processes areas items of strengths and weaknesses, and this is evident from the following chart:

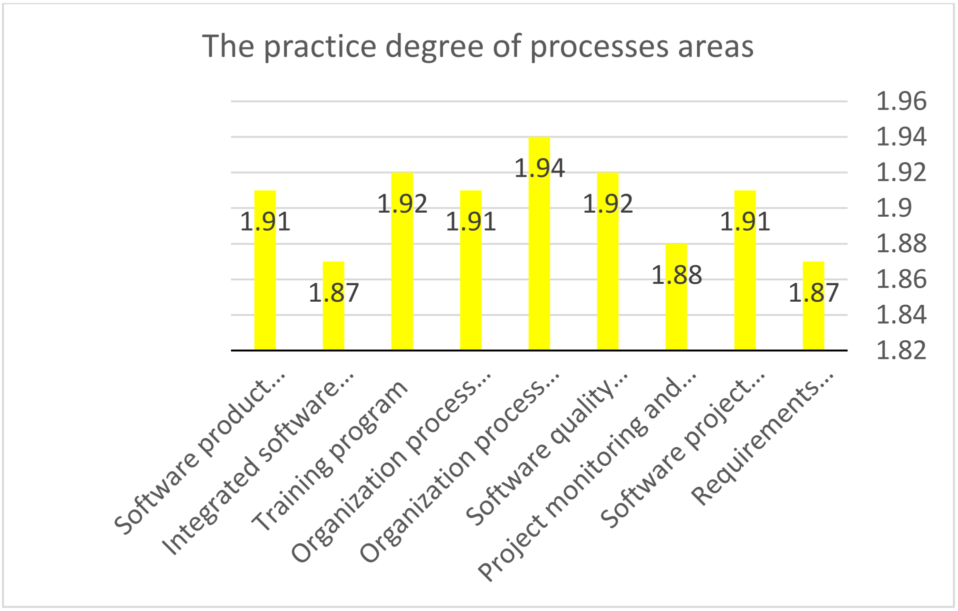

Figure (1): The practice degree of process areas

8 Conclusion

As shown previously, the objectives of this study are to determine the local software process efficiency, to identify the strengths and weaknesses of the local software process, and to identify the problems that affect the effectiveness of the software process in Libyan banks and Derna City as a case. After the discussion in the previous section, it can be concluded the number of problems that affect local software quality and suggestions to solve them as follows:

Problem 1: Weak in the software process of owning and monitoring.
Suggest resolving 1:

-A member must be responsible for monitoring the process and specifying the responsibilities of team members.

- Mechanisms and measures are put in place to identify and monitor the status of tracking the software process.

Problem 2: A process is not aligned with the project's objectives.
Suggest resolving 2:

It's noted that most organizations lack a software engineering process group (SEPG) responsible for software process management. This group functions to ensure that the software is compatible with the project objectives. Therefore, it's suggested that this group be established in all organizations.

Problem 3: The team is not motivated to follow the standard software process.
Suggest resolving 3:

-It's suggested that "Process institutionalization" be applied, which means that following a process is supported and encouraged by organizational policies and procedures and supported by management.

Reference

Allison, I. (2005). Towards an agile approach to software process improvement: addressing the changing needs of software products. *Communications of the IIMA*, *5*(1).

Almeida, F. (2020). Bibliometric analysis of agile software development. *arXiv preprint arXiv:2004.05876*.

Dyba, T. (2000). An instrument for measuring the key factors of success in software process improvement. *Empirical software engineering*, *5*, 357-390.

Farooq, U., Ahmed, M., Hussain, S., Hussain, F., Naseem, A., & Aslam, K. (2021). Blockchain-based software process improvement (BBSPI): An approach for SMEs to perform process improvement. *IEEE Access*, 9, 10426-10442.

Kautz, K., & Ramzan, F. (2001, January). Software quality management and software process improvement in Denmark. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences* (pp. 10-pp). IEEE.

Kokol, P. (2021). Software quality: A historical and synthetic content analysis. *arXiv preprint arXiv:2106.14598*.

Neelu, L., & Kavitha, D. (2021). Estimation of software quality parameters for hybrid agile process model. *SN Applied Sciences*, 3(3), 296.

Nunnally. J.C and Bernstein. I.H. (1994). *Psychometric theory*, McGraw-Hill. New York.

Salkind, N. J. (2017). *Tests & measurement for people who (think they) hate tests & measurement*. Sage Publications.

Sarwosri, S., Rochimah, S., Yuhana, U. L., & Hidayat, S. B. (2023). Software Quality Measurement for Functional Suitability, Performance Efficiency, and Reliability Characteristics Using Analytical Hierarchy Process. *JOIV: International Journal on Informatics Visualization*, 7(4), 2421-2426.

Schtein, I. A. (2018). *Management Strategies for Adopting Agile Methods of Software Development in Distributed Teams*. Walden University.

Shukla, S.S., Pandey, R.K., Gidwani, B., Kalyani, G. (2023). *Improvement in Software Quality. In: Pharmaceutical Calibration, Validation and Qualification: A Comprehensive Approach*. Springer, Singapore.

Thorndike, R. L., Hagen, E. P., & Sattler, J. M. (1986). *Stanford-Binet Intelligence Scale: Fourth Edition*. Itasca, IL: Riverside Publishing.

Wang, C. N., Vo, T. T. B. C., Hsu, H. P., Chung, Y. C., Nguyen, N. T., & Nhieu, N. L. (2024). Improving processing efficiency through workflow process reengineering, simulation and value stream mapping: a case study of business process reengineering. *Business Process Management Journal*. 30(7), 2482-2515.

Wheeler, S., & Duggins, S. (1998, April). Improving software quality. In *Proceedings of the 36th annual Southeast regional conference* (pp. 300-309).