Enhancing Software Quality Assurance in First Unicorn Coffee Shop in Indonesia: A Test Maturity Analysis

Margreth H Saragih * and Astari Retnowardhani

Department Information System Management Department, Bina Nusantara University, Jakarta, Indonesia
Email: margreth.saragih@binus.ac.id (M.H.S.); aretnowardhani@binus.edu (A.R.)
*Corresponding author

***Abstract*—Ensuring software quality involves a fundamental task known as software testing. In software development, there are many complex activities and the potential for errors. Quality Assurance (QA) stands as one of the most essential tasks. This paper presents qualitative research outcomes aimed at assessing the maturity level of the software testing process utilizing a designated framework. The significance of this study lies in its capacity to offer suggestions for enhancing the testing process through the development of a more comprehensive Standard Operating Procedure (SOP), thereby mitigating potential future fraud. The software testing process's maturity level is evaluated within the company under investigation in this study using the Test Maturity Model integration (TMMi) framework at level 2 (managed). This corresponds to the evaluation guidelines outlined in the TMMi Assessment Method Accreditation Requirements (TAMAR), which involve assessing process areas, sub-practices, specific practices, generic practices, and specific goals. After analyzing the evaluation of the TMMi software testing process's maturity against 5 process areas at level 2, it was determined that the company being examined is currently operating at maturity level 1 (initial stage).**

***Keywords*—software testing, quality assurance, maturity level, Test Maturity Model integration (TMMi), TMMi Assessment Method Accreditation Requirements (TAMAR)**

I. Introduction

Software testing is defined as a procedure for assessing the excellence of the software by scrutinizing its functionality, dependability, performance, and security aspects. The main goal of software testing is to ensure the seamless functioning of software and its alignment with user requirements [1]. This concept is strengthened by in Ref. [2], which portrays software testing as a systematic method for appraising the attributes and traits of software, all aimed at verifying the software's compliance with business requisites and user expectations. The ISEB Software Testing Foundation Syllabus Version 2018 also elucidates that software testing can be executed manually or automatically to enhance testing efficiency and efficacy. Furthermore, pivotal components of the software testing process encompass test-related documents, such as test blueprints and outcome summaries.

Quality Assurance (QA) is a crucial activity. To ensure the quality of software products, it is essential to continuously perform Verification and Validation (V&V) activities throughout the software development and maintenance process. The primary verification and validation activity in software development is testing, which aims to ensure the software's quality. Software testing is an activity that requires intensive knowledge [3].

This paper was created to find out the maturity level of the software testing process at one of the unicorn startup companies in Indonesia based on the Test Maturity Model integration (TMMi) and to provide recommendations for improvement based on the maturity level testing result. The case study was conducted at the First Unicorn Coffee Shop in Indonesia. It has more than 800 stores spread throughout Indonesia and has even extended its presence to other Southeast Asian nations. Start-up companies will be called unicorns when the valuation value of the company has reached US$ 1 billion or IDR 14.1 trillion. In 2021, this start-up received the first phase of series C funding worth US$96 million or around Rp1.3 trillion, officially one of Southeast Asia's first retail F&B Unicorns.

This company, particularly in the Technology Division's Software Quality Assurance section, is dedicated to delivering the best results by developing bug-free applications that function seamlessly for both internal and external use. To ensure the quality of our systems, we conduct daily testing and schedule weekly production releases. Software testing is a crucial step in guaranteeing software quality, as stated in Ref. [4]. According to Garousi *et al.* [1], software testing encompasses a comprehensive investigation of the software to validate its adherence to the determined criteria and to discern the repercussions of any detected flaws, even in cases where the precise locations of these defects remain unidentified. It is important to note that software testing can be costly, particularly as software

Manuscript received February 17, 2024; revised April 30, 2024; accepted June 17, 2024; published August 28, 2024.

development progresses and the number of test cases that need evaluation increases.

In this company, the software testing process during development is carried out by Software Quality Assurance (SQA) through several stages. These stages include System Integration Testing (SIT) in the Test environment, User Acceptance Testing (UAT), Regression Testing in the staging environment, and Sanity Testing in the production environment. Software development involves complex activities and potential errors, crucial for Quality Assurance (QA). Sustaining the quality of software products relies on the ongoing Verification and Validation (V&V) undertakings throughout the software development and maintenance phases. Testing is the core verification and validation endeavor within software development, aiming to secure the software's quality. It is worth noting that software testing requires substantial knowledge and expertise [3].

During the testing process at this Company, the SQA Team encountered several issues and shortcomings. One primary concern was the absence of a written Standard Operating Procedure (SOP) for the testing process, which proved critical when instances of fraud were discovered during sanity testing in the production environment, resulting in financial losses. Additionally, some bugs were overlooked in the production environment because the test cases used during SIT and UAT were not detailed enough, and the SQA Team might not have been thorough in their testing approach. Moreover, there were situations where testing processes did not align with the test cases, possibly due to the unavailability of fully prepared test cases caused by insufficient testing time. Lastly, some bugs were still found because the test cases did not cover all requirements. These shortcomings must be resolved to enhance the overall efficiency and precision of the testing procedure.

The research questions that guide the focus of this study are: What is the existing maturity stage of the software testing process in this company? What suggestions can be offered to improve the testing process for this company? This paper aims to evaluate the maturity level in the software testing process within this company, employing a particular framework for the assessment. The goal is to offer recommendations for enhancing the testing process by creating a more detailed Standard Operating Procedure (SOP) to prevent fraud in the future. The research will focus on the testing process of the central business system at this company, where the potential for fraud exists. The study will specifically concentrate on providing improvement recommendations for the testing process and will not cover the planning of follow-up recommendations.

II. Literature Review

A. Software Testing

As stated by Laksono *et al.* [5], software testing is an approach to assess software quality by scrutinizing its functionality, performance, and security attributes. The fundamental goal of software testing is to reduce the likelihood of software malfunctions, elevate software quality, and ensure its correct functioning while adhering to user specifications. This document introduces the Test Maturity Model (TMM) concept, which measures the maturity level of a company's software testing process. TMM encompasses five progressive stages: Initial, Repeatable, Defined, Managed, and Optimized. Each stage exhibits distinct attributes and outlines the organization's capability to conduct software testing activities effectively and efficiently.

Based on the objectives contained in Ref. [2, 5], it can be concluded that the purpose of software testing is to ensure by verifying and validating all software functions are by the expectations of user needs that a system with minimal defects is produced to reduce risk and increase user confidence in system quality.

As mentioned in Ref. [2], it outlines four testing tiers aligned with the four developmental phases. Each testing tier encompasses distinct characteristics such as well-defined objectives, testing reference points, tested elements, and the types of detected issues or imperfections.

Component unit or module testing examines separate parts of software development. The goal of component testing is to mitigate risks, verify the functional and non-functional requirements of the designed and specified components, build confidence in the quality of the components, identify defects in the components, and prevent defects from reaching higher levels.

Integration testing focuses on the interactions among components or systems involved in the tested software. Integration testing can be divided into two types: component integration testing, which demonstrates interactions among units or modules within the system, and system integration testing, which shows the integration among systems and services, whether internal to the organization or external. Integration testing aims to reduce risks, verify functional and non-functional requirements of the designed and specified interfaces, build confidence in the quality of interfaces, identify defects in the interfaces themselves or the components, and prevent defects from reaching higher levels.

System testing focuses on assessing the overall capabilities and features of the system, considering both functional and non-functional requirements throughout its operation. The objectives of system testing include reducing risks, verifying that the functional and non-functional features of the system align with its design and specifications, validating that the system is complete and operates as expected, building confidence in the overall quality of the system, identifying defects, and preventing defects from moving to higher levels or entering the production environment. Acceptance test, also known as User Acceptance Test (UAT), is like system testing; the difference is that in UAT, the one who tests is the user or direct user of the system. The purpose of the acceptance test is the same as that of system testing. The things that become the basis or reference for implementing this test are business processes, business or user needs, regulations, contracts, standards, use cases, installation

procedures, and risk analysis documents. The defects found in this test are system workflows that do not meet business needs, business rules are not implemented correctly, the system does not meet the needs of contracts or regulations, and non-functional failures such as information security and system performance efficiency.

B. Software Testing Process

As stated in Ref. [2], the fundamental procedures of software testing consist of testing planning, testing monitoring and control, testing analysis, testing design, testing execution, and testing completion. Testing planning is an activity that defines the research goals and the approach taken to achieve those goals, considering limitations related to the context. This includes specifying testing techniques and activities and formulating a testing schedule to align with the project timeline. Testing planning can be revisited based on feedback received from monitoring and control activities.

Testing monitoring involves comparing the current progress with the monitoring matrix established in the previous testing plan. Testing control encompasses necessary actions to achieve the goals outlined in the testing plan and is updated as the testing process unfolds. Testing monitoring and control activities support the evaluation of exit criteria, referring to the definition of completion at various stages of the Software Development Life Cycle (SDLC).

During test analysis, the basis of the test is analyzed to identify features and define the test environment. In other words, the test analysis determines "what will be tested?" and the scope according to the acceptance criteria. In the test design stage, the test conditions are elaborated into high-level test cases, a collection of high-level test cases, and other software needed for testing. The analysis stage discusses what will be tested, and the design stage will answer "how to carry out the test?". Moreover, testing tools are needed to execute and complete tests during the test implementation stage. This includes sorting the test cases to be tested into test procedures. So, this stage can answer the question, "Do we have everything we need to do testing?"

After the test implementation stage, test execution is carried out, and a series of tests is carried out according to the test execution schedule. The last stage is testing completion, where data collection from completed testing activities is carried out to consolidate experience, testing tools, and other relevant information. The test completion activity is a project reference when the software is released; testing is completed or canceled.

C. Software Testing Process Improvement

The improvement of the software testing process is a series of activities conducted to enhance software testing during its development within an organization [5]. According to Ref. [2], the software testing process improvement involves activities designed to enhance the performance and maturity of the testing processes within an organization and the testing outcomes, such as the generated system. Based on both sources, it can be concluded that the software improvement process is an initiative or activity aimed at improving the performance and maturity level of the testing processes to produce high-quality software.

A typical process is employed to improve the software testing process and assess the maturity level of the testing processes. The process begins with the analysis of requirements, as outlined in the problem formulation, indicating the need to assess the maturity level of the testing process as the foundation for enhancing the software testing process within the organization. The subsequent steps involve raising awareness among stakeholders and management. The team then defines several considerations, including targets, improvement areas, models, and approaches to be adopted. Following that, the process of assessing the maturity level of the testing process is carried out to identify and evaluate areas for improvement in the testing process. If necessary, the maturity level assessment process may be repeated to achieve continuous improvements in the testing process.

In the research conducted by Garousi *et al.* [6], 9 models for assessing the maturity level of testing processes are categorized into 3 groups. The first category is the generic category, which includes TMMi, TPI, and Test SPICE. The second category is for specific types of software development, encompassing models like the Agile Quality Assurance Model (AQAM), Agile Testing Maturity Model (ATMM), and TPI for Embedded Software and Industrial Characteristics (TPI-EI). The final category, based on specific testing activity targets, involves the Unit Test Maturity Model (UTMM), Automated Software Testing Maturity Model (ASTMM), and Personal Test Maturity Matrix (PTTM).

Based on the research conducted by Garousi, Felderer, & Hacaloğlu in "What We Know about Software Test Maturity and Test Process Improvement" in 2018, it can be concluded that there are three commonly used models to assess the level of software testing maturity, which can be applied to enhance testing processes within an organization. These models are TMMi, TPI Next, and Test SPICE. This finding is supported by a study conducted by Laksono, Budiardjo, & Ferdinansyah in 2019, identifying three types of testing maturity models: TMMi, TPI, and TMM. The research involves comparing the effectiveness of these three models in improving the quality of software testing processes. The results indicate that the TMMi model is the most effective in enhancing the maturity of software testing compared to the TPI and TMM models.

D. Test Maturity Model Integration

TMMi is commonly recognized as the successor or evolution of the Test Maturity Model (TMM), initially developed by a consortium of testing and quality professionals under the TMMi Foundation. The TMMi framework serves as a guiding principle and point of reference for enhancing the software testing process, employing the concept of maturity levels to assess and enhance the process. TMMi uses the same terminology as ISTQB [7]. This is also added by a survey conducted by Garousi *et al.* [8] on 74 respondents from several different industrial fields; 37 came from Asia and 25 from

Europe. For industry categories, 31% or 23/74 respondents came from the IT software industry, and 28% or 21/74 respondents came from the financial services industry. The survey results indicated that the primary motivations for adopting TMMi included enhancing product quality, minimizing product risks, boosting testing productivity, aligning with globally recognized models, and elevating testing team benchmarks. An impressive 87% of the participants affirmed that the enhancements brought about by the TMMi-based testing process not only met but even surpassed their expectations.

TMMi utilizes a maturity level format like the Capability Maturity Model Integration (CMMI), which is based on the testing procedures implemented within the organization. This progression begins with an ad-hoc or initial phase (level 1), advances to a managed stage (level 2), then a defined phase (level 3), followed by a measured phase (level 4), and culminates in an optimized phase (level 5) [7].

Laksono *et al.* [5] investigated the application of the Test Maturity Model (TMM) to assess the maturity stage of the software testing process and provide recommendations for its improvement. This study encompassed a comparative examination involving TMM and alternative software testing process models. Employing a descriptive analysis approach and case studies in two distinct organizations, the research assessed the efficiency and benefits of TMMi. The results showed that TMMi adds value in improving software testing quality and helps organizations understand the weaknesses and strengths of their testing process. However, the research also found some weaknesses in TMMi, such as the lack of support for adapting the test process model to different organizational contexts and the lack of focus on automated testing. The researchers propose that organizations adopt TMMi as an instrument to assess and enhance their software testing processes. However, it is advised that organizations consider both the merits and limitations of this model.

As per the findings of Garousi and Veenendaal [7], the global adoption of the Test Maturity Model Integration (TMMi) as an assessment framework for software testing maturity is widespread. This study draws upon data gathered from over 200 organizations across the globe that have employed TMMi to evaluate and enhance their software testing processes. The results reveal a significant increase in adopting TMMi in recent times, solidifying its position as the dominant benchmark for evaluating the maturity of software testing methodologies. Furthermore, this research also found that the banking and financial sector exhibits a higher testing maturity level than other sectors. In addition, it was observed that TMMi certification is increasingly recognized by organizations and governments worldwide. The researchers suggest that organizations contemplate the adoption of TMMi to assess and enhance their software testing processes. Additionally, they advocate pursuing TMMi certification to guarantee the execution of software testing at a high-quality standard. However, the authors also stress the importance of adapting this model to different organizational contexts, as each organization has different needs and challenges in software testing.

Unudulmaz and Kalıpsız [9] elucidate that TMMi functions as a framework enabling organizations to enhance their software testing maturity and assess the efficacy of their testing endeavors. The authors then discuss how TMMi can be integrated with Agile methodologies and software testing processes. The authors demonstrate that amalgamating TMMi with Agile methodologies can help organizations enhance software product quality, expedite development timelines, and boost customer contentment. Moreover, the authors delve into how TMMi contributes to refining software testing processes within organizations. They emphasize that TMMi serves to pinpoint aspects requiring enhancement in the software testing process, offering distinct directives on effecting those improvements. However, the authors also emphasize the importance of considering the challenges in integrating TMMi with Agile and software testing processes. They recommend that organizations consider the complexity and difficulty of changing organizational culture before integrating TMMi with Agile methodologies and software testing processes.

As per the findings of Veenendaal *et al*. [10], several pivotal drivers impel organizations towards embracing TMMi. These motivations encompass the imperative to enhance software product quality, fulfill contractual obligations, and enhance the efficiency and efficacy of software testing. The study further reveals that organizations adopting TMMi have reaped substantial advantages, including elevated software product quality, reduced developmental costs and timelines, and heightened customer contentment. Nevertheless, the research identifies certain impediments organizations face in adopting TMMi, such as expenses, intricacies, and challenges in transforming organizational culture. Hence, organizations seeking to enhance their software testing maturity should contemplate employing TMMi to assess and elevate their software testing processes.

E. Maturity Level Comparison of Testing Process

As stated by Laksono *et al.* [5], a comparative analysis of the three frameworks—TMMi, TPI Next, and Test SPICE—can be conducted using five distinct categories. The first one is the Framework Structure, which characterizes the model's organizational layout within the framework. The second is Core Emphasis, which identifies the primary focal point of enhancing the testing process based on the framework's owned process area. The third is the Comprehensive Approach, encompassing a portrayal of the methodology employed by the framework to ameliorate the testing process. Then the fourth is Testing Methodology, detailing the specific testing approach employed within the framework. Moreover the last one is software process improvement, which illustrates the interconnection between improving the testing process and enhancing the broader software development process.

Among the five groupings, there are extra divisions as per the classification by Garousi and Veenendaal [7].

These extra divisions include testing levels, terms, and certification for each framework. Based on comparing these three methodologies, the researcher concluded that TMMi is a suitable framework for use in the case study at Kenangan Brands Company. This is based on TMMi providing an assessment of the level of maturity at each level of testing, starting from unit testing, integration, system, and UAT; this is in line with the practice of the testing process running at Kenangan Brands Company today, which refers to the V-Model testing process. This differs from the TPI Next and Test SPICE frameworks, which focus only on the latest tests in the testing cycle, namely system testing and UAT. Moreover, TMMi establishes a connection with the CMMI model to establish a linkage between enhancing the testing process and improving the overall software development process. This unique feature is not present within TPI Next and Test SPICE. The last consideration is the terminology used in TMMi, which refers to ISTQB. The subsequent sub-section will provide a more detailed conversation regarding the TMMi framework employed for gauging the maturity stage of the software testing process.

F. TMMI Assessment

TMMi utilizes the TMMi Assessment Method Accreditation Requirements (TAMAR) to conduct its evaluation processes. TAMAR is based on the global standard ISO/IEC 33002 [ISO 33002], which outlines the requirements for TMMi Assessment Methods. Consequently, any TMMi Assessment Methods that meet the criteria specified in TAMAR will align with the interpretations of ISO/IEC 33003 (Goslin, Wells, Balla, & Veenendaal, 2023). This is necessary to ensure consistency of assessments referring to TAMAR. Based on Ref. [11], TAMAR has several activities: planning and preparation, data collection, data analysis and reporting, and assessment closure.

Four ratings are assigned to process components: Not Achieved (N), Partially Achieved (P), Largely Achieved (L), and Fully Achieved (F). These ratings apply to various elements, including specific and general practices, objectives, process areas, and maturity levels. However, the percentage-based assessment can only be applied to specific practices [12].

N, P, L, and F Scores can be applied across all four levels where evaluations are used, including Practices (Specific and Generic), Targets (Specific and Generic), Process Areas, and Maturity Levels. Achievement ratings based on percentages only relate to Practices (both Specific and Generic). The designations NA and NR are integrated following the recommendations outlined by Goslin *et al*. [13].

Attaining an N (Not Achieved) score or rating for a specific process component demands minimal evidence demonstrating compliance with TMMi standards. Substantial deficiencies exist in implementation, and viable alternatives are lacking. An “N” score within the Process area suggests the presence of at least one N-rated Goal, and within the Maturity Level, it indicates the existence of at least one N-rated Process area.

Earning a P (Partially Achieved) score or rating for a particular process component necessitates some evidence showcasing adherence to TMMi principles. However, the practices and processes remain unfinished, sporadically applied, or inconsistent, and the results might not be uniform. The Process segment includes at least one Goal with a P grade, while all remaining Goals are assigned F or L ratings. Simultaneously, the Maturity Level indicates the presence of at least one Process area with a P grade, while all other Process areas are classified with F or L ratings.

To secure an L (Largely Achieved) score or rating within a specific process component, substantial evidence validating adherence to TMMi requisites is necessary. Within the Process area, at least one Goal with an L rating is noted, while all other Goals bear F ratings. Simultaneously, the Maturity Level indicates the presence of at least one Process area with an L rating, while all other Process areas possess F ratings.

To achieve an F (Fully Achieved) score or rating for a given process component, compelling and consistent evidence showcasing adherence to TMMi principles is imperative. In Process areas, all Goals carry an F rating, and within Maturity Levels, all Process areas hold an F rating.

It will be utilized to attain an NA (Not Applicable) score or rating when a process area is irrelevant. The process area is considered irrelevant when there is no logical necessity for practicing it within the organization. A process area will not receive an NA rating if the stakeholder deliberately excludes it from the assessment scope.

An NR (Not Rated) score or rating will be assigned in cases where a process element cannot be appraised due to inadequate or inconsistent evidence or when it lies beyond the scope of the assessment. Process areas agreed upon by the sponsor or assessor to be excluded based on the various factors shall be marked with an NR rating.

The TMMi assessment using the TAMAR method will be conducted according to the evaluation process areas, with the implementation status for each practice categorized as fully implemented, partially implemented, primarily implemented, not implemented, or not applicable. To assess the Process Areas (PA), Specific Goals (SG) will be evaluated for each system, and Generic Goals (GG) will be evaluated at the organizational level.

The process area rating assessment is conducted based on the status of each practice within it. For example, if all practices in PA 2.1 receive a score above 85%, the Process Area 2.1 Test Policy and Strategy will be rated as Fully Achieved. After assessing the specific goals, the generic goals are evaluated next. Based on the assessment of both Specific Goals and Generic Goals, if the overall score for Process Area 2.1 is 85%, the Goal Rating will be Largely Achieved. Therefore, the Process Area 2.1 Test Policy and Strategy assessment result is Largely Achieved.

After completing the assessment for other process areas, it was found that 2.2 Test Planning received a

rating of Partially Achieved. In contrast, PA 2.3 Test Monitoring and Control, PA 2.4 Test Design and Execution, and PA 2.5 Test Environment received ratings of Largely Achieved. Based on these maturity level assessments, it is concluded that the lowest rating is Partially Achieved, making the overall maturity level rating Partially Achieved. From these results, it can be concluded that the organization has not yet reached TMMi maturity level 2 (managed) and remains at TMMi maturity level 1 (initial).

G. Deming Cycle

This section will elaborate on the Deming cycle, often called PDCA (Plan, Do, Check, Act). It is a subsequent step after assessing the software testing process's maturity level. Specifically, this cycle provides recommendations for enhancing the software testing process. The Deming cycle is a generic improvement used for continuous improvement involving setting improvement goals, taking action to achieve them, and once achieved, setting new improvement goals [2]. The Deming framework, as shown in Fig. 1, consists of the following steps:

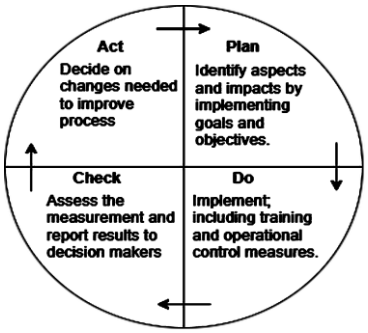

Fig. 1. Deming cycle (PDCA) (source: ISTQB, 2018).

The first two steps (Plan and Do) are essential as they contain the activities to be performed. In the last two steps (Check and Act), statistical methods and system analysis techniques are often used to show statistical significance dependencies and identify areas for improvement [2].

III. Methodology

Building upon the discourse presented in the preceding chapter, the enhancement of the software testing process through the TMMi framework will employ a qualitative data processing approach. As per Creswell [3], qualitative research involves data collection techniques encompassing interviews, observations, document analysis, and audio-visual materials.

A. Research Methodology

The assessment process is performed on the components within TMMi, including aspects such as sub-practices, distinct practices, general practices, particular objectives, general objectives, process areas, and the maturity level of the software testing process. The sequence of assessment steps is illustrated sequentially in Fig. 2.

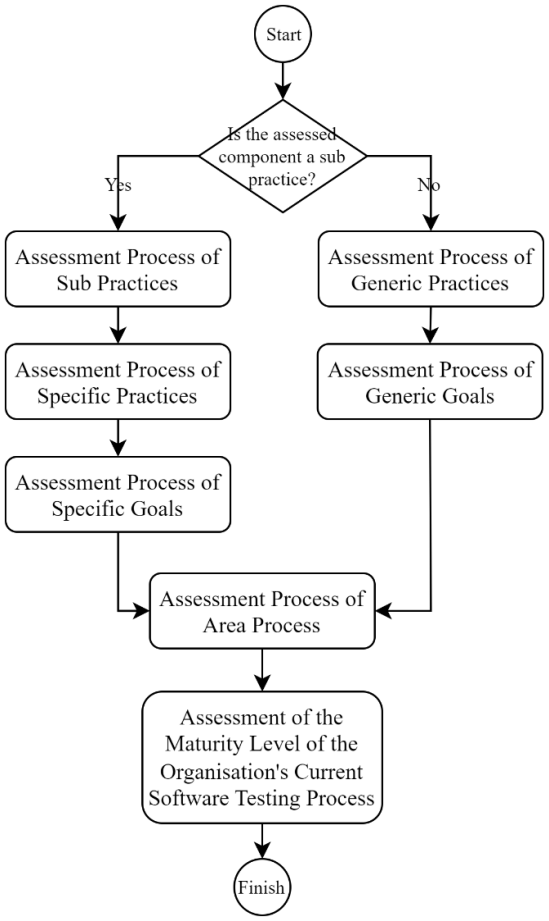

Fig. 2. Assessment of testing process maturity level using TMMi.

The overall procedure for evaluating the level of maturity in the testing process begins with an assessment of individual sub-practices that constitute specific practices. Likewise, the evaluation extends to scrutinizing each generic practice within each process area. The results of assessing specific practices serve as the foundation for evaluating specific objectives, while the findings from assessing generic practices contribute to evaluating generic objectives.

Furthermore, based on the evaluation of each process area's maturity level, the subsequent steps involve considering the following conditions: if the obtained value is "F" or "L," the assessment can progress towards a higher maturity level; however, if the outcome is "P" or "N," the maturity level value is positioned one level lower than the assessed level.

B. Research Instrument

This section will discuss the research data collection instruments consisting of a list of interview questions based on the process areas to be assessed, the selection of interviewees, and the process of processing and analyzing the data. In making a list of interview questions, it will refer to the area process components that will be assessed using the TMMi framework.

Through the survey results in [2] of more than 100 companies with 9 industrial sectors, two of which are mainly from the IT and financial services industries, it

was found that 63% of respondents reached level 1 to level 2. Meanwhile, 37% had progressed from level 2 to level 3. The second tier includes the process domains that cover test policy and strategy, test planning, test monitoring and control, test design and execution, and the test environment. Derived from the survey outcomes, the research on this case study will be assessed at level 2. However, it does not rule out the possibility that there will be process areas that suggestions or recommendations from sources will assess. After determining the level of maturity to be assessed, a list of questions will be compiled.

The process area components will be adjusted to the maturity level and assessed according to the organization's needs. This study will assess it at the managed level or level 2, so the process area to be assessed is PA 2. The process area will be assessed based on the practices that are evidence of the maturity level assessment and become a list of interview questions in this study.

TABLE I. LIST OF SPEAKERS

Source	Position	Reference
Source 1	SQA Manager	[13]
Source 2	SQA Engineer	[13]
Source 3	Product Manager	[13]
Source 4	Technical Project Manager	[13]

C. Data Analysis Method

In this research, data analysis begins with collecting and processing data obtained from interviews, FGDs, observations, and data analysis to collect assessment evidence. Furthermore, the management or analysis of research data is carried out. Data analysis follows the provisions of TAMAR.

Then, an assessment checklist is formed according to each practice's assessed process area and implementation status. Then, the status will be as fully implemented, partially implemented, primarily implemented, not implemented, and not applicable.

After the checklist for each practice in the process area is assessed, the next step is to assess the ranking of the process area itself. The process area rating assessment is carried out based on the status of each practice in it. For example, if all practices in PA 2.1 have been fully implemented, the PA 2.1 process area related to test policy and strategy will be given a Fully Achieved rating.

Moreover, after evaluating each process area at a specific maturity level, a subsequent maturity level assessment will be conducted. This assessment, termed the maturity level assessment, adheres to the guidelines established by TAMAR and employs the identical five ratings as the process area assessment, encompassing fully achieved, primarily achieved, partially achieved, not achieved, and not rated/not applicable.

IV. RESULT AND DISCUSSION

This segment will encompass the outcomes and discourse about the findings presented in this paper.

A. Maturity Level Assessment Planning

Within the assessment's defined scope, the testing process will encompass three systems: System 1, System 2, and System 3, representing the company's core business systems. The choice to assess these three systems was driven by establishing the minimum number of projects required for evaluation and analyzing maturity levels in similar industries. In alignment with these considerations, this research centers around evaluating the software testing process's second level of maturity, encompassing three selected systems. The evaluation targets the general testing process employed across all project samples rather than being limited to any test type.

The testing process's maturity level is evaluated through separate stages for each sample project. These stages entail interviews conducted with testers involved in the testing processes of the project samples, supplemented by internal documentation evidence. The interviews are structured around a predefined questionnaire checklist, addressing each sub-practice inherent to specific practices. This evaluation methodology remains consistent for generic practices as well. The scores derived from specific practices form the foundation for evaluating specific goals, while the scores from generic practices serve as the basis for assessing generic goals. The values assigned to specific and general goals subsequently form the basis for evaluating each process area at maturity level 2. Ultimately, these combined evaluations determine the maturity level of the software testing process within the organization.

B. Maturity Level Assessment Results

The outcomes of evaluating the maturity stage of the software testing process within the examined company are derived from appraising each element within the process domain aligned with maturity level 2, following the guidelines of TMMi. In this segment, we will elaborate on the evaluation of every process area, encompassing elements such as Specific Practices (SP), General Practices (GP), Distinct Goals (SG), General Goals (GG), and the assessments of process areas that serve as the foundation for determining maturity levels.

The evaluation of TMMi through the TAMAR approach involves scrutinizing each process area and assigning an implementation status to every practice. This status can range from fully implemented, partially implemented, primarily implemented, not implemented, to not applicable. The assessment of a Process Area (PA) entails appraising Specific Goals (SG) within individual systems and Generic Goals (GG) across the organization.

- **PA 2.1 Test Policy and Strategy**

The process area for test policy and strategy encompasses three SGs: establishing a test policy, formulating a test strategy, and defining test performance indicators. The assessment results for SP, presented as a percentage, are derived from evaluating sub-practices within each SP. Subsequently, the percentage values are categorized according to TAMAR guidelines, resulting in a rating. The cumulative value of each specific practice serves as the foundation for evaluating specific goals. In

the test policy and strategy process area, SG 1 rated "P", SG 2 rated "L", and SG 3 rated "F".

After assessing the accumulation of specific practices to obtain the specific goals score, the next step is to assess each general practice to obtain the generic goals score. The generic goals are assessed on all practices in the organization, not specific to a particular project. Based on the assessment of generic practices and generic goals in the Test Policy and Strategy Process Area, it is found that GG 2 institutionalizes a managed process, obtaining a score of 25%, which is categorized as partially achieved. The results of the Generic Goals (GG) and Specific Goals (SG) form the basis for the process area assessment, as seen in Table II.

TABLE II. GOAL RATINGS FOR PROCESS AREA 2.1

PA 2.1 Goal Ratings	Partially Achieved	P
Generic Goals	GG 2 – Partially Achieved	P
Specific Goals	SG 1 – Fully Achieved	P
	SG 2 – Fully Achieved	L
	SG 3 – Fully Achieved	F

Based on the GG and SG assessments obtained in Table II, the lowest value is partially achieved, so PA 2.1 gets a partially achieved value. This is by the provisions of TAMAR, namely the process area, which is assessed based on the lowest value obtained from comparing GG and SG.

- **PA 2.2 Test Planning**

The test planning (PA 2.2) includes five SGs: performing a product risk assessment, establishing a test approach, defining test estimates, developing a test plan, and obtaining a commitment to the test plan. SG 1 consists of three SPs, SG 2 consists of five SPs, SG 3 consists of three SPs, SG 4 consists of five SPs, and SG 5 consists of three SPs. The evaluation results for SP, presented as percentages, are acquired from assessing sub-practices within each SP. Additionally, the percentage value is classified based on the TAMAR provisions, resulting in a rating of "N" for SG 1 Perform a Product Risk Assessment, an "L" for SG 2 Establish a Test Approach, SG 4 Develop a Test Plan, and SG 5 Obtain Commitment to the Test Plan. Meanwhile, SG 3 Establish Test Estimates is assigned a rating of "F".

After assessing the accumulation of specific practices so that the value of specific goals is obtained, each general practice is assessed to get the value of generic goals. The generic goals are assessed for the whole organization, not specific to a particular project. The assessment results of generic practices and generic goals in the Test Planning Process Area obtained a score of 50%, which was categorized as partially achieved. The assessment of GG and SG is the basis for assessing the process areas, as seen in Table III.

Based on the GG and SG assessments obtained in Table III, the lowest value is not achieved, so the test planning (PA 2.2) gets a not achieved value. This is by the provisions of TAMAR, namely the process area, which is assessed based on the lowest value obtained from comparing GG and SG.

TABLE III. TEST PLANNING PROCESS AREA RATING

PA 2.2 Goal Ratings	Not Achieved	N
Generic Goals	GG 2 – Partially Achieved	P
Specific Goals	SG 1 – Partially Achieved	N
	SG 2 – Largely Achieved	L
	SG 3 – Largely Achieved	F
	SG 4 – Largely Achieved	L
	SG 5 – Largely Achieved	L

- **PA 2.3 Test Monitoring and Control**

The test monitoring and control (PA 2.3) includes three Specific Goals (SG): monitoring test progress against the plan, monitoring product quality against the plan and expectations, and managing corrective actions to closure. SG 1 consists of seven SPs, SG 2 consists of seven SPs and SG 3 consists of three SPs. The assessment results for SP, presented as percentages, are derived from evaluating sub-practices within each SP.

After an accumulated assessment of specific practices to obtain a value for specific goals, each generic practice is assessed to obtain a value for generic goals. The generic goals are assessed for the whole organization, not specific to a particular project. The assessment results of generic practices and generic goals in the Test Monitoring and Control Process Area are primarily achieved. The assessment of Generic Goals (GG) and Specific Goals (SG) is the basis for assessing the process area, as seen in Table IV.

TABLE IV. TEST MONITORING AND CONTROL PROCESS AREA RATING

PA 2.3 Goal Ratings	Largely Achieved	L
Generic Goals	GG 2 – Largely Achieved	L
Specific Goals	SG 1 – Largely Achieved	L
	SG 2 – Largely Achieved	L
	SG 3 – Fully Achieved	F

Based on Table IV's GG and SG assessment, the lowest value is achieved mainly, so the test monitoring and control (PA 2.3) gets a largely achieved value. This is by the provisions of TAMAR, namely the process area, which is assessed based on the lowest value obtained from comparing GG and SG.

- **PA 2.4 Test Design and Execution**

The test design and execution (PA 2.4) comprises four Specific Goals (SG): performing test analysis and design using test design techniques, executing test implementation, conducting test execution, and managing test incidents to closure. SG 1 consists of four SPs, SG 2 consists of four SPs, SG 3 consists of four SPs and SG 4 consists of three SPs. The evaluation results for SP, presented as percentages, are obtained from assessing the sub-practices within each SP.

Furthermore, an assessment of each general practice is carried out to get the value of GG. The assessment of GG is done for the whole organization, not specifically for a particular project. The results of the assessment of GP and GG in the PA 2.4 are categorized as primarily achieved. The assessment of GG and SG is the basis for assessing the process area, as seen in Table V.

TABLE V. TEST DESIGN AND EXECUTION PROCESS AREA RATING

PA 2.4 Goal Ratings	Largely Achieved	L
Generic Goals	GG 2 – Largely Achieved	L
Specific Goals	SG 1 – Largely Achieved	L
	SG 2 – Fully Achieved	F
	SG 3 – Fully Achieved	F
	SG 4 – Fully Achieved	F

Based on the GG and SG assessments obtained in Table V, the lowest value is primarily achieved, so the PA 2.4 is largely achieved. This is by the provisions of TAMAR, namely the process area, which is assessed based on the lowest value obtained from comparing GG and SG.

- **PA 2.5 Test Environment**

The test environment (PA 2.5) encompasses three SG: developing the test environment, performing test environment implementation, and managing and controlling the test environment. SG 1 consists of three SPs, SG 2 consists of four SPs and SG 3 consists of four SPs. The assessment results for SP, presented as percentages, are obtained from evaluating sub-practices within each SP. The assessment results for specific practices are derived from the overall assessment of sub-practices and then converted into a percentage.

Additionally, each broad practice is evaluated to determine the value of GG. The assessment of GG pertains to the entire organization and is not specific to any project. The outcomes of assessing GP and GG in PA 2.5 reveal that GG 2 scored 70%, falling under the category of primarily achieved. Table VI illustrates that the comprehensive evaluation of Generic Goals (GG) and Specific Goals (SG) is the foundation for assessing the process area.

TABLE VI. TEST ENVIRONMENT PROCESS AREA RATING

PA 2.5 Goal Ratings	Largely Achieved	L
Generic Goals	GG 2 – Largely Achieved	L
Specific Goals	SG 1 – Fully Achieved	F
	SG 2 – Largely Achieved	L
	SG 3 – Largely Achieved	L

Based on the GG and SG assessment obtained in Table VI, the lowest value is achieved mainly so that the test environment (PA 2.5) gets a largely achieved value. This is by the provisions of TAMAR, namely the process area, which is assessed based on the lowest value obtained from comparing GG and SG.

C. Maturity Level of Testing Process

The assessment of each process area at maturity level 2 is grounded in the thorough evaluation of SP, SG, GP, and GG. The process areas evaluated at TMMi maturity level 2 include PA 2.1, PA 2.2, PA 2.3, PA 2.4, and PA 2.5. An overview of the scores for all process areas is presented in Table VII.

Table VII shows that the lowest value is not achieved, specifically in the test planning process area (PA 2.2). In other words, the maturity level of the software testing process at this company for maturity level 2 has not yet been achieved. PA 2.3, 2.4, and 2.5 can largely be achieved, while PA 2.1 is still partially achieved. This is due to specific practices in PA 2.2 not being consistently fulfilled and reviewed, particularly the creation of test plan documents.

TABLE VII. MATURITY LEVEL OF SOFTWARE TESTING PROCESS

Maturity Level 2 Rating	Not Achieved	N
Process Area Rating	PA 2.1 – Partially Achieved	P
	PA 2.2 – Not Achieved	N
	PA 2.3 – Largely Achieved	L
	PA 2.4 – Largely Achieved	L
	PA 2.5 – Largely Achieved	L

Furthermore, in PA 2.3, 2.4, and 2.5, most specific practices can still be fulfilled despite some practices in the test plan section not being carried out. Based on these results, it can be concluded that the maturity level of the company's software testing process, according to TMMi, is currently at maturity level 1 (initial). Several process improvements are necessary to reach maturity level 2, particularly in the test planning process area, which received the lowest score of not achieved. The recommendations for enhancing the software testing process will further discuss these improvements.

D. Recommended Improvement for Testing Process

From the results of the Focus Group Discussion (FGD) and the Plan-Do-Check-Act (PDCA) process that have been conducted, it can be concluded that the proposals obtained from the FGD can be applied as recommendations for improving the testing process. However, it is essential to prioritize these proposals to select the most critical solution for implementation, considering the cost and resources available.

Currently (as of 2023), the testing process lacks standardization, as there is no Standard Operating Procedure (SOP) related to the testing process. Therefore, the first step that should be taken is to create an SOP for the testing process so that all testing processes have standardization followed by everyone involved in the testing process. Afterward, risk identification is conducted by adding product or system risks to the risk register. This recommendation is also supported by the PDCA process in SP 1.1 (Define product risk categories and parameters), SP 1.2 (Identify product risks), and SP 1.3 (Analyze product risks).

The next step is to create user access management that will regulate the access rights of all users accessing the system and implement a double-check system with approval from immediate superiors for all activities performed. This is one way to identify and analyze potential risks that may arise during testing, supported by PDCA SP 1.2 (Identify product risks), SP 1.3 (Analyze product risks), and SP 4.4 (Identify test project risks). Following this, creating and reviewing test cases is necessary before testing to ensure that testing is thorough and minimizes the possibility of overlooking test steps. This aligns with PDCA SP 2.1 (Identify items and features to be tested), SP 5.1 (Review test plan), and SP 5.3 (Obtain test plan commitments).

The following process to be facilitated is testing with a specific User Group (CUG) number to ensure that everyone involved in testing does not use personal numbers, reducing the risk of fraud. Additionally, all numbers used for testing should be allowed to limit access only for transactions during testing. This is also related to the PDCA process in SP 1.2 (Identify product risks) and SP 4.4 (Identify test project risks).

In addition to the software improvement recommendations that can be implemented in the future, companies or organizations can also consider several inputs, such as ensuring comprehensive testing plan documentation, including detailed procedures, achievements, and criteria for testing activities. Implement standard templates for testing plan documentation and establish review processes to validate the completeness and accuracy of the testing plan. They strengthen quality assurance oversight by assigning specific roles and responsibilities to monitor and control the testing process. Empowering the quality assurance team to conduct regular audits, identify areas for improvement, and implement corrective actions to enhance the overall effectiveness of testing activities. Moreover, they could even adopt or implement advanced automated testing tools and techniques if feasible.

In addition to the software improvement recommendations that can be implemented in the future, companies or organizations can also consider several inputs. Firstly, they should develop and implement standard testing planning practices to adequately document all aspects of the testing process. This includes ensuring comprehensive testing plan documentation, including detailed procedures, achievements, and criteria for testing activities, implementing standard templates for testing plan documentation, and establishing review processes to validate the completeness and accuracy of the testing plan.

Secondly, strengthening quality assurance oversight is crucial. This can be achieved by assigning specific roles and responsibilities to monitor and control the testing process, empowering the quality assurance team to conduct regular audits, identify areas for improvement, and implement corrective actions to enhance the overall effectiveness of testing activities.

Thirdly, developing clear guidelines and standards for testing in all areas is essential to promote consistency and uniformity. Additionally, providing training and resources to ensure team members understand and consistently adhere to best practices is essential. Fourthly, robust monitoring mechanisms are recommended to track progress and performance in all process areas. Utilizing feedback from monitoring activities to identify areas for improvement and effectively implement corrective actions is also crucial. Furthermore, fostering collaboration among stakeholders, project teams, and quality assurance professionals to streamline the testing process and ensure a holistic approach is beneficial. Encouraging open communication channels to address issues, share best practices, and collectively drive continuous improvement efforts should also be prioritized.

Lastly, conducting periodic assessments using the TMMi framework to track progress, identify trends, and compare them with industry standards, then using the assessment results to drive targeted improvements in specific areas and consistently improve the overall maturity level of the software testing process is essential. By implementing these additional inputs, companies or organizations can further enhance their software testing processes and achieve higher levels of maturity and effectiveness.

V. Conclusion

Should the lowest maturity level assessment rating within a specific process area be classified as partially achieved, the resulting overall maturity level rating would also be categorized as partially achieved. Based on these findings, the inference can be drawn that the organization has yet to attain maturity level 2 (managed) but instead remains at the initial level 1 of TMMi. Consequently, offering recommendations to the organization to enhance the testing process within the assessed process area becomes imperative.

Developing recommendations for improving the software testing process using the Deming cycle or PDCA (Plan, Do, Check, Act) method aimed at the test plan process area (PA 2.2 in TMMi), which obtained a score of not achieved. Process improvements are proposed on specific practices in the process area so they can be largely or entirely achieved. The proposed improvement recommendations include 8 specific practices with 7 Plan, 27 Do 7, Check 7, and 7 Act.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

The first author is responsible for being the principal author of this paper, and the second author is responsible for being the first author's supervisor for writing this paper. All authors had approved the final version.

References

[1] V. Garousi, A. Rainer, P. Lauvås, and A. Arcuri, "Software-testing education: A systematic literature mapping," *Journal of Systems and Software*, 110570, 2020.

[2] ISTQB, *ISEB Software Testing Foundation Syllabus Version 2018*, 2018.

[3] J. W. Creswell and J. D. Creswell, *Research Design Qualitative, Quantitative, and Mixed Methods Approaches*, 5th Ed. Los Angeles: SAGE Publication, 2018.

[4] Experimentus, *Test Maturity Model Integrated (TMMi) Survey Results*, London: Experimentus, 2018.

[5] M. A. Laksono, E. K. Budiardjo, and A. Ferdinansyah, "Assessment of test maturity model: A comparative study for process improvement," in *Proc. 2nd International Conference on Software Engineering and Information Management*, 2019, pp. 110–118.

[6] V. Garousi, M. Felderer, and T. Hacaloğlu, "What we know about software test maturity and test process improvement," *IEEE Software*, vol. 35, pp. 84–92, 2018.

[7] V. Garousi and E van Veenendaal, "Test Maturity Model integration (TMMi): Trends of worldwide test maturity and certifications," *IEEE Software*, 2021.
[8] V. Garousi, E. V. Veenendaal, and M. Felderer, *Motivations for and benefits of adopting the Test Maturity Model*, United Kingdom: TMMi Foundation, 2021.
[9] A. Unudulmaz and O. Kalıpsız, "TMMI integration with agile and test process," in *Proc. ACM EASE Conference (EASE'20)*, 2020.
[10] E. V. Veenendaal, V. Garousi, and M. Felderer, "Motivations for and benefits of adopting the Test Maturity Model integration (TMMi)," *Software Quality*, 2022.
[11] TMMi Foundation, *Test Maturity Model integration: Guidelines for Test Process Improvement Release 1.2*, Ireland.: TMMi Foundation, 2018.
[12] E. V. Veenendaal, J. J. Cannegieter, and K. Balla, "The new MDD and Tamar a study on how Mdd, the new CMMi assessment method can affect TMMi assessments," TMMi Foundation, 2023.
[13] A. Goslin, B. Wells, K. Balla, and E. V. Veenendaal, *TMMi Assessment Method Application Requirements (TAMAR) Release 1.1*, United Kingdom: TMMi Foundation, 2023.