Journal of Embedded System Security and Intelligent Systems
E-ISSN: 2722-273X; P-ISSN: 2745-925X
https://journal.unm.ac.id/index.php/JESSI

Sensitivity Analysis of Weight Normalization Schemes in McCall-Based Software Quality Measurement

Ahmad Farisi✉
Universitas Multi Data Palembang, Palembang, Indonesia
Dafid
Universitas Multi Data Palembang, Palembang, Indonesia

ARTICLE INFO

Keywords:

McCall model
Sensitivity analysis
Software quality measurement
Weight normalization
Weighted sum model

Article History

Received: May 13, 2026
Revised : July 26, 2026
Accepted : August 16, 2026

ABSTRACT

Purpose – This study examines the sensitivity of McCall-based software quality measurement to alternative weight normalization intervals and rounding configurations. It specifically evaluates whether different weighting schemes produce consistent Total Quality scores and quality classifications when the evaluated system, criteria values, and aggregation procedures remain unchanged.

Design/methods/approach – A quantitative experimental design was applied to the Product Operation dimension of the McCall model using Simponi, a web-based learning management system at Universitas Multi Data Palembang, as the case study. Criteria values were obtained from 166 student respondents, while indicator importance ratings were provided by two software engineering experts. Three normalization intervals, namely 0.1–0.5, 0.1–0.4, and 0.4–0.8, were evaluated using floor and ceiling rounding configurations. The resulting scores were calculated through hierarchical weighted additive aggregation.

Findings – Weight normalization substantially influenced the resulting software quality assessment. The moderate and expanded weighting schemes produced comparable Total Quality outcomes and consistently classified the evaluated system at the highest quality level, whereas the compressed weighting scheme generated a lower overall assessment and shifted the system into a lower quality category. In contrast, differences between floor and ceiling rounding configurations were negligible and did not alter the resulting quality classification.

Research implications/limitations – Weight normalization should be explicitly documented and standardized to improve comparability across McCall-based evaluations. The findings are limited to the Product Operation dimension, three predefined intervals, and one academic web application.

Originality/value – This study provides empirical sensitivity evidence showing that weight interval selection is a substantive methodological decision that can alter software quality interpretations, whereas rounding configurations have negligible practical effects.

To cite this article : Farisi, A., Dafid. (2026). Sensitivity Analysis of Weight Normalization Schemes in McCall-Based Software Quality Measurement. *Journal of Embedded System Security and Intelligent Systems, 7*(3), 47- 63. 10.59562/jessi.v7i3.2604

Correspondence Author: ✉ahmadfarisi@mdp.ac.id

INTRODUCTION

One way to maintain software quality is by conducting software quality measurement that is crucial for several reasons, as it aids in process and product control by providing detailed information about the software. This includes facilitating process improvement, adhering to standards, enhancing code quality, and managing and assuring overall quality [1]. Though numerous software quality measurement approaches exist, the software industry still faces challenges in their practical implementation and institutionalization [2]. Several methods can be used to measure software quality, including Boehm, ServQual, FURPS, WebQual, ISO 9126, Delone & McLean, ISO 25010, SQL-OSS, ISO 25022, and McCall [3]. One method widely accepted

and considered a best practice for measuring software quality from a product perspective is the McCall method [4], [5], which continues to be applied in recent software quality evaluations [6], [7], [8]. McCall is a software quality model that defines quality factors and criteria and organizes them into Product Operation, Product Revision, and Product Transition. The McCall method provides a structured, product oriented evaluation across multiple quality attributes. It conducts comprehensive and in depth assessments. The McCall method measures system quality across three aspects: product revision, product transition, and product operation [9].

Some previous research has measured software quality, and many of these studies have considered the McCall model as a prominent and widely adopted framework for software quality assessment, due to its comprehensive and systematic approach to evaluating various quality attributes [4], [5]. This includes quality measurements on applications such as the online sales application for light steel materials [10], the assessment of healthcare software systems [11], the software quality measurement of the integrated Hajj information system [12], the news portal website analysis [13], the measurement of district government websites quality [14], the analysis of the Edu-Smart Learning Management System software [15], and the software quality analysis of e-voting software [16], the quality measurement on Academic Information System [17], among others that have used McCall to measure software quality [18]. Most of the studies utilizing McCall to assess software quality focus on the product operation aspect. Despite the widespread use of the McCall method, researchers have highlighted the challenge of accurately determining the appropriate weighting of the quality factors within the model [19], [20], [21].

The calculation of software quality using the McCall method begins with determining the criteria values (c), which are obtained through variables and indicators in the Product Operation aspect, including correctness, reliability, efficiency, integrity, and usability [22]. Each variable consists of specific indicators used to assess software quality. Correctness includes completeness, consistency, and traceability; reliability consists of error tolerance, accuracy, and simplicity; efficiency is measured through execution efficiency and conciseness; integrity encompasses access control, security, and access audit; while usability is assessed based on operability, training, and communicativeness. Once the c values are determined, the next step is to calculate the weight (w) for each indicator within a range of 0 to 1, which is used to measure the significance level of each indicator in evaluating software quality [23].

Previous studies have adopted various weighting schemes. Some studies, such as [13], [15], [16], [20], [24] applied weight scales ranging from 0.1 (least important) to 0.5 (most important). Meanwhile, studies [14], [25], [26], [27], [28] used a weighting scale ranging from 0.1 (least important) to 0.4 (most important). Additionally, studies [15], [21], [29] implemented a different range, from 0.4 (least important) to 0.8 (most important). These variations in weighting schemes indicate differences in how the significance of indicators is determined within the McCall method. Therefore, further investigation is necessary to examine how alternative weighting approaches influence software quality measurement outcomes within the McCall model. This investigation is expected to contribute empirical evidence that supports the standardization of weighting interval selection in McCall-based software quality evaluations.

Almost all studies that measure software quality using the McCall method determine the weights through interviews and questionnaires with experts who are familiar with the software being evaluated [10], [25], [27]. The experts complete a questionnaire containing McCall variables and indicators to assess the importance level of each indicator, using a scale ranging from 1 (very unimportant) to 5 (very important). The questionnaire results are then normalized to ensure that the weights remain within the 0 to 1 range [16]. In many McCall-based studies, the normalization process is commonly performed by transforming expert ratings into predefined weight intervals, such as 0.1-0.5, although alternative intervals have also been reported in the literature [13], [20]. In the weighting process for McCall method indicators, it is highly likely that multiple experts will be involved, leading to variations in the assigned weights. As a result, the average values may not always match the predefined range of 0.1 to 0.5, but instead fall between these values, such as 0.15, 0.25, 0.35, or 0.45, depending on the number of

experts involved. Instead, the average values may vary, making it necessary to consider both ceiling and floor values to determine their impact on the final weight distribution. Given this uncertainty, it is essential to investigate alternative weighting schemes, particularly when ceiling and floor adjustments are applied, to determine their impact on software quality measurement outcomes. By analyzing these variations, this research seeks to examine how different weighting schemes influence software quality measurement outcomes and to provide empirical evidence supporting methodological standardization in McCall-based evaluations. This study contributes empirically by demonstrating how alternative weighting schemes influence hierarchical aggregation outcomes within the McCall model. Rather than proposing a new quality model, this research provides sensitivity-based evidence to support methodological standardization in practical software quality evaluation. Based on these considerations, this study seeks to answer the following research question: To what extent do alternative weight normalization intervals influence Total Quality scores and quality classifications in McCall-based software quality evaluations?

METHOD

This study employs a quantitative experimental research design to investigate the impact of alternative weighting schemes within the McCall software quality measurement method. The methodological framework is structured to ensure alignment between the research objective, namely examining how different weight normalization intervals and rounding configurations influence software quality scores, and the computational procedures used to derive those scores. The research process begins with instrument development based on the Product Operation aspect of the McCall model, followed by expert judgment for weight determination and student-based evaluation to obtain criteria values (c). The collected data are normalized and processed using the established McCall equations to compute Fa indicator, Fa variable, and Total Quality scores. By comparing alternative weighting schemes under controlled computational conditions, this method provides an analytical basis for evaluating the sensitivity and consistency of software quality results when different weight normalization strategies are applied.

McCall Software Quality Model

The McCall software quality model, first introduced by Jim McCall and colleagues in the late 1970s, remains a foundational framework for assessing software quality through structured and hierarchical attributes. The model classifies software quality into three main dimensions: Product Operation, Product Revision, and Product Transition, each comprising specific quality factors and criteria [18].

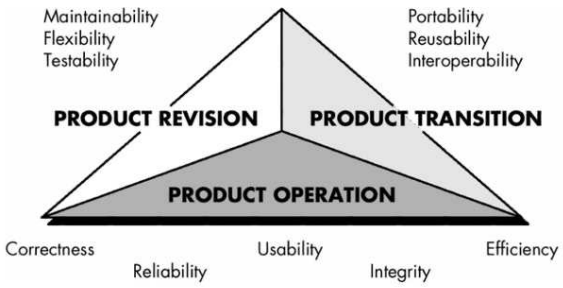

Figure 1. McCall Quality Factors

This study focuses on the Product Operation aspect, which remains widely applied in software quality evaluations due to its practical and functional orientation [3], [11].

The Measurement Process of McCall Method

Based on some previous research like [3], [9], [13], [19], the McCall method in the Product Operation aspect begins with determining the criteria values (c), which are obtained through predefined variables and indicators, namely correctness, reliability, efficiency, integrity, and usability. Each variable consists of specific indicators used to measure software quality. Correctness includes completeness, consistency, and traceability; reliability has indicators such as error tolerance, accuracy, and simplicity; efficiency is measured through execution efficiency

and conciseness; integrity includes access control, security, and access audit; while usability is assessed based on operability, training, and communicativeness. These criteria values are obtained through questionnaire data analysis, where each indicator is measured using a Likert scale of 1 to 5, with 1 representing "strongly disagree" and 5 meaning "strongly agree". The original Likert scores were used directly to compute the average criteria values without any scale modification across experimental scenarios.

The next step is to determine the weights for each indicator (w indicator), with a value range between 0 and 1. The weighting process is carried out through consultations with experts familiar with the software being evaluated. Experts also complete a questionnaire with the same variables and indicators, but using a different Likert scale, which assesses the importance level of each indicator, ranging from 1 (very unimportant) to 5 (very important). The questionnaire results are then normalized to ensure that the weights remain within the 0 to 1 range. Once the w indicators are obtained, the variable weights (w variable) are calculated based on the average w indicator values for each variable. Before proceeding to the next stage, it must be ensured that the total w indicator and total w variable each sum to 1. If the total w exceeds 1, normalization is required. The next step is to calculate the quality factor value (Fa indicator) using the following equation (1).

$$Fa\_indicator = (w_1 \times c_1) + (w_2 \times c_2) + \ \dots + (w_n \times c_n) \quad (1)$$

Once the Fa indicator values are determined, the next step is to calculate the total factor value per variable (Fa variable) using the equation (2).

$$Fa\_variable = \frac{Fa\_indicator_1 + Fa\_indicator_2 + \dots + Fa\_indicator_n}{n} \quad (2)$$

where n represents the number of indicators within a variable. The Fa variable is then converted into a percentage using the equation (3).

$$Percentage = \frac{Obtainerd\ Value}{Maximum\ Value} \times 100\% \quad (3)$$

This percentage represents the quality level of each variable. After that, the overall software quality value (Total Quality) is calculated using the equation (4).

$$Total\ Quality = \frac{(Fa\_variable_1 \times w\_variable_1) + (Fa\_variable_2 \times w\_variable_2) + \dots + (Fa\_variable_n \times w\_variable_n)}{Maximum\ Value} \quad (4)$$

The result of this calculation provides an overview of the overall software quality. The final step is to interpret the Total Quality value by comparing it against predefined quality ranges. These ranges are generally adjusted according to the Likert scale used. For example, if a Likert scale of 1 to 5 is applied, the software quality interpretation can be categorized into five levels: 81-100% (highly qualified), 61-80% (qualified), 41-60% (moderately qualified), 21-40% (low quality), and 1-20% (very low quality). With this interpretation, the Total Quality value can be used to assess the extent to which the software meets the quality standards measured using the McCall method in the Product Operation aspect.

The aggregation structure applied in the McCall computation corresponds to a weighted linear aggregation model, commonly referred to as the weighted sum model (WSM). In this approach, each criterion contributes proportionally to the final score according to its assigned weight, and the overall evaluation is derived through additive combination. Weighted aggregation models are widely used in multi-criteria evaluation frameworks due to their interpretability and computational stability [4], [30], [31]. In additive aggregation models, variations in weight dispersion directly influence the magnitude and distribution of the final composite score, making weight normalization a critical determinant of model sensitivity.

Research Design

The research is classified as experimental quantitative, as it involves empirical testing by manipulating independent variables (i.e., weighting schemes) in the form of alternative weight normalization ranges and observing their effect on the dependent variable (i.e., software quality score). According to [32], experimental research is used to determine whether a specific treatment influences an outcome, which aligns with the purpose of this study in comparing the effects of alternative weight normalization intervals and rounding configurations on software quality measurements. The research methodology is illustrated in Figure 2.

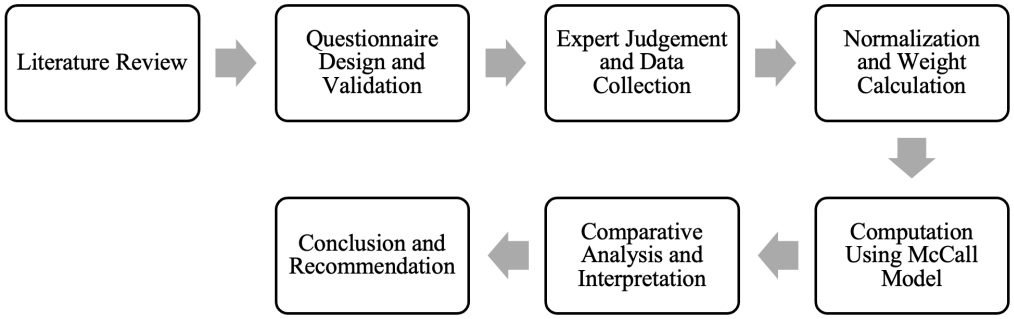

Figure 2. Research Methodology

Figure 2 shows the research methodology which was conducted through several sequential stages to ensure a structured and traceable research process, namely:

1. Literature review, which involved examining previous studies to identify how the McCall software quality model has been applied and to explore the different weighting strategies used in similar research. This step also served to identify research gaps and formulate the rationale for comparing ceiling and floor weighting schemes.
2. Questionnaire design and validation, where the research instrument was developed based on indicators under the Product Operation aspect of the McCall model. The draft questionnaire was then validated by experts to ensure the clarity, relevance, and consistency of each item.
3. Expert judgement and data collection, where the validated questionnaire was distributed to selected experts in software engineering. Each expert was asked to assess the importance of each indicator using a Likert scale ranging from 1 (very unimportant) to 5 (very important).
4. Normalization and weight calculation, in which expert importance ratings (1-5) were linearly normalized into predefined weight intervals according to each weighting scheme (0.1-0.5, 0.1-0.4, and 0.4-0.8). For multiple experts, the average weight of each indicator was calculated prior to applying rounding adjustments. Two rounding schemes were then evaluated: ceiling and floor.
5. Computation using the McCall model, where for each scheme, software quality scores were calculated by multiplying each indicator's normalized weight with its corresponding criteria value (c), calculating the average score for each variable, and then deriving the overall software quality score in percentage form.
6. Comparative analysis and interpretation, which involved comparing the results obtained from all weighting schemes and rounding configurations to identify their influence on Total Quality outcomes and classification consistency.
7. Conclusion and recommendation, where conclusions were drawn based on the analysis, and suggestions were provided for future studies and practitioners using the McCall model in software quality evaluation.

Weighting Schemes

This study employs three alternative weighting schemes as experimental treatments to evaluate the sensitivity and computational stability of the McCall software quality measurement model. The selection of these schemes is grounded in variations observed in prior empirical studies, where different normalization ranges have been adopted, namely 0.1-0.5, 0.1-0.4, and 0.4-0.8. These inconsistencies indicate the absence of a standardized weighting interval in

McCall-based quality evaluations. In addition to range transformation, this study also applies ceiling and floor adjustments to the averaged expert weights in order to simulate practical rounding decisions commonly encountered in empirical McCall implementations. Although previous studies have applied different weighting intervals, little empirical evidence has been reported regarding how these alternative normalization ranges influence the resulting Total Quality score and quality classification within the McCall model.

In this research framework, the weighting scheme functions as the independent variable, while the resulting Total Quality score represents the dependent variable. The criteria values (c), derived from student respondents, the hierarchical McCall structure (Fa indicator, Fa variable, and Total Quality), and the evaluated software system remain constant across all scenarios. This controlled configuration ensures that any variation in the final quality score is attributable exclusively to differences in the applied weight ranges.

It is important to emphasize that the criteria values (c), derived from student responses using a 5-point Likert scale (1-5), remain identical across all three scenarios. No transformation, scale compression, or response adjustment was applied to the student evaluation data. Therefore, any variation observed in the resulting Fa indicator, Fa variable, or Total Quality score is exclusively attributable to differences in the applied weighting normalization scheme.

The three weighting schemes are defined as follows:

1. Scenario 1: Moderate Range (0.1-0.5)
   In this scheme, expert importance ratings using a Likert scale of 1 to 5 are normalized by dividing the scores by 10, resulting in weights ranging from 0.1 to 0.5. This range represents the most commonly adopted normalization approach in previous McCall-based studies. It provides proportional differentiation between indicators while maintaining moderate dispersion within the weighting distribution.
2. Scenario 2: Compressed Range (0.1-0.4)
   This scheme applies a narrower upper bound, limiting the maximum normalized weight to 0.4. Conceptually, this compression reduces the differentiation gap between low-importance and high-importance indicators. While this approach may promote conservative weighting, it may also introduce aggregation dampening effects within the hierarchical structure of the McCall model, potentially influencing the magnitude of the Total Quality score. In Scenario 2, the expert importance ratings (1-5) were normalized into a compressed interval between 0.1 and 0.4 using linear scaling. The transformation follows a linear min-max normalization principle, which preserves ordinal relationships while adjusting the dispersion of values within a specified interval [33]. The normalization process applied in Scheme 2 is expressed as follows:

$$w' = a + \frac{x - x_{min}}{x_{max} - x_{min}}(b - a) \qquad (5)$$

   where w' represents the transformed weight, x is the original expert rating, x_{min} and x_{max} denote the minimum and maximum values of the original Likert scale (1 and 5), while a and b represent the lower and upper bounds of the target normalization interval. For Scheme 2, the normalization parameters were defined as a=0.1 and b=0.4. Linear normalization is widely used in multivariate data analysis and measurement scaling to ensure comparability across alternative metric ranges [33].
3. Scenario 3: Expanded Upper Range (0.4-0.8)
   In this scheme, the weight interval is shifted upward, producing values between 0.4 and 0.8. This expanded upper range increases the separation between indicators in terms of relative importance. From a computational perspective, a wider dispersion may reduce rounding-induced variance when applying floor and ceiling adjustments, thereby improving numerical stability in hierarchical aggregation.

Although assigning a minimum weight of 0.4 to indicators categorized as very unimportant may appear conceptually expansive, this interval was adopted based on weighting patterns reported in prior McCall-based studies. The objective of this study is not to validate the theoretical superiority of the interval, but rather to evaluate its computational sensitivity and influence on Total Quality outcomes under controlled conditions. Nevertheless, assigning a relatively high minimum weight of 0.4 may introduce a potential floor effect, whereby indicators assessed as least important continue to contribute proportionally to the aggregated quality score. As a consequence, the differentiation between low-importance and high-importance indicators may become less pronounced than conceptually expected. Therefore, Scheme 3 should be interpreted as an experimental normalization configuration for sensitivity analysis rather than as a universally optimal weighting strategy.

Case Study

To support this research, data were collected to measure software quality, and the results were subsequently processed using various weighting schemes based on the McCall quality model. The software selected as the case study in this research is a web-based application called Simponi, which stands for Sistem Pembelajaran Online dan Interaktif (Online and Interactive Learning System), developed and implemented at Universitas Multi Data Palembang. Simponi serves as the university's official e-learning platform, facilitating online teaching and learning processes between lecturers and students. It provides features such as digital course materials, interactive communication tools, assignment submissions, and learning progress tracking. The system is publicly accessible via the following URL: https://simponi.mdp.ac.id.

The selection of Simponi as the case study in this research is based on practical considerations. Since the primary focus of this study is to compare the outcomes of software quality measurement using the McCall model under different weighting schemes, the specific software used as a case study is not a determining factor in the validity of the research. What matters most is that the software under evaluation has sufficient functionality and complexity to allow the application of McCall's quality criteria, particularly those within the Product Operation aspect. Simponi meets this requirement as a feature-rich, actively used academic web application. Therefore, it serves as a suitable and representative case for demonstrating how different weighting approaches may affect the final quality score in McCall-based evaluations.

Research Instrument

The research instrument used in this study consists of a questionnaire comprising 40 items, adapted from several previous studies such as [10], [12], [13], [14], [15], [16], [20], [21], [25], [29], which conducted software quality measurements across various case studies. The questionnaire items are presented in Table 1.

Table 1. Questionnaire's Items

No	Variables	Indicators	Code	Questionnaire's Items
1	Correctness	Completeness	CO1	The software can process data.
			CO2	All features available in the software function properly.
			CO3	The software can display information in every provided menu.
			CO4	The software displays relevant information for each menu.
			CO5	The information displayed in the software is complete, clear, and easy to find.
			CO6	The availability of required information in the software is always up to date.
		Consistency	CS1	The software has a consistent appearance.
			CS2	The software maintains consistent design (color, font type, layout) across all pages.
			CS3	The language used in the software is consistent across all pages.
			CS4	The form and button design in the software is

No	Variables	Indicators	Code	Questionnaire's Items
				consistent on each page.
			CS5	The table features and design in the software are consistent across all pages.
			CS6	Data management in the software is consistent across all forms.
		Traceability	TB1	The software can search data across the entire system content.
			TB2	The software provides the latest information and shows the last updated date or time.
2	Reliability	Error Tolerance	ET1	The software can function again after a system failure caused by a server downtime.
			ET2	The software experiences minimal damage when encountering a system failure.
		Accuracy	AR1	The software facilitates easy input entry required by the system.
			AR2	The software can display accurate data based on the searched keyword.
			AR3	The information produced by the software is accurate and error-free.
			AR4	The software provides data and information precisely according to user needs.
			AR5	Software users can obtain needed information in a timely manner.
		Simplicity	SP1	The software is easy to use and not confusing.
			SP2	The information in the software is easy to understand without difficulty.
			SP3	The menus in the software are easy to understand without difficulty.
3	Efficiency	Conciseness	CC	The language used in the software is easy and quick to understand.
		Execution Efficiency	EE1	The functions and data in the software menus match user needs.
			EE2	The software processes data storage quickly.
			EE3	The software efficiently processes data and information.
4	Integrity	Security	SR1	The software cannot be used by others except through individual user accounts.
			SR2	The software login process functions correctly and meets user expectations.
			SR3	The software provides a notification in case of login failure.
			SR4	The software can control user access by limiting access rights.
5	Usability	Operability	OB1	The software is easy to use.
			OB2	The software can be easily operated.
			OB3	The software provides the required information that can be found easily and accurately.
			OB4	The software menus and displayed information are easy to understand.
			OB5	Overall, the software provides user satisfaction and comfort.
		Training	TR1	New users can easily use the software.
			TR2	The software provides help services to assist new users in operating the software.

No	Variables	Indicators	Code	Questionnaire's Items
		Communicativeness	CM	The software has an attractive, neat, and user-friendly interface.

Population and Sample

The respondents of this questionnaire consist of two types. The first type includes students who evaluate the software quality based on the variables and indicators derived into the questionnaire items. The second type consists of experts who assess the importance level of each indicator. While student respondents provide ratings based on their experience in using the software, the experts determine the indicator weights required for McCall-based quality computation. The expert category consisted of two individuals selected through saturated sampling based on their qualifications and professional portfolios in software development. For student respondents, the population comprised 1,897 active students at Universitas Multi Data Palembang in the 2024/2025 academic year. Using the Yamane formula [34], also known as Slovin formula [35], with a 5% margin of error, the estimated minimum sample size was 330 respondents.

In practice, 166 valid student responses were successfully collected and used for analysis. Although this number is below the estimated sample size for full population inference, the primary objective of this study is to conduct a comparative computational evaluation of alternative weighting schemes within the McCall model rather than to generalize findings to the entire population. In experimental research, the focus lies in examining treatment effects under controlled conditions [32]. Therefore, the available dataset is considered sufficient to generate stable mean criteria values (c) for sensitivity analysis across weighting schemes.

RESULTS AND DISCUSSION

This section presents the results of the computational evaluation and discusses the practical implications of alternative weighting schemes within the McCall model. The analysis focuses on comparing Total Quality outcomes across different weight normalization intervals and examining their impact on interpretative consistency and computational stability.

Results of Expert Judgement and Data Collection

Before proceeding to further analysis, all collected responses were subjected to validity and reliability testing using SPSS software. The validity test was performed using the Pearson product-moment correlation technique at a significance level of 5 percent. The degree of freedom (df) was determined using the formula df = n − 2, where n represents the number of respondents. With 166 student respondents, the df value was calculated as 164. Based on df = 164 at the 0.05 significance level (two-tailed), the critical r_table value was 0.1524. Each item's calculated correlation coefficient (r_calculated) was compared with this threshold. An item was considered valid when r_calculated exceeded 0.1524, indicating a statistically significant correlation between the item score and the total construct score [36]. The results showed that all questionnaire items satisfied the validity criteria.

Reliability testing was conducted using Cronbach's Alpha to evaluate the internal consistency of the measurement instrument. Cronbach's Alpha assesses the extent to which items within a construct consistently measure the same underlying concept. In applied quantitative research, a Cronbach's Alpha value greater than 0.60 is considered acceptable, particularly in exploratory or model-application studies rather than pure scale development research [37]. The reliability test results indicated that all variables produced Cronbach's Alpha values above 0.60, confirming that the instrument demonstrated acceptable internal consistency. The detailed results of both validity and reliability testing are presented in Table 2 and Table 3, which summarize the r_calculated values, the r_table threshold of 0.1524, and the Cronbach's Alpha coefficients.

Table 2. Results of Validity Testing (Pearson Correlation, α = 0.05, df = 164, r_table = 0.1524)

Code	r_calculated	r_table	Decision	Code	r_calculated	r_table	Decision
CO1	0.611	0.1524	Valid	AR5	0.775	0.1524	Valid
CO2	0.698	0.1524	Valid	SP1	0.778	0.1524	Valid

Code	r_calculated	r_table	Decision	Code	r_calculated	r_table	Decision
CO3	0.748	0.1524	Valid	SP2	0.848	0.1524	Valid
CO4	0.746	0.1524	Valid	SP3	0.800	0.1524	Valid
CO5	0.647	0.1524	Valid	CC	0.841	0.1524	Valid
CO6	0.674	0.1524	Valid	EE1	0.806	0.1524	Valid
CS1	0.747	0.1524	Valid	EE2	0.782	0.1524	Valid
CS2	0.764	0.1524	Valid	EE3	0.744	0.1524	Valid
CS3	0.806	0.1524	Valid	SR1	0.513	0.1524	Valid
CS4	0.686	0.1524	Valid	SR2	0.674	0.1524	Valid
CS5	0.715	0.1524	Valid	SR3	0.672	0.1524	Valid
CS6	0.776	0.1524	Valid	SR4	0.660	0.1524	Valid
TB1	0.760	0.1524	Valid	OB1	0.754	0.1524	Valid
TB2	0.695	0.1524	Valid	OB2	0.798	0.1524	Valid
ET1	0.579	0.1524	Valid	OB3	0.783	0.1524	Valid
ET2	0.608	0.1524	Valid	OB4	0.848	0.1524	Valid
AR1	0.786	0.1524	Valid	OB5	0.811	0.1524	Valid
AR2	0.720	0.1524	Valid	TR1	0.702	0.1524	Valid
AR3	0.761	0.1524	Valid	TR2	0.687	0.1524	Valid
AR4	0.719	0.1524	Valid	CM	0.788	0.1524	Valid

All items have r_calculated values greater than 0.1524, indicating that all 40 indicators are statistically valid at the 5% significance level. The highest correlation coefficient was obtained for items SP2 and OB4 (0.848), while the lowest value was observed for SR1 (0.513). Although variation exists among indicators, all coefficients remain substantially above the minimum threshold, demonstrating adequate construct representation and measurement consistency across the instrument.

Table 3. Results of Reliability Testing

Variable	Number of Items	Cronbach's Alpha	Interpretation
Product Operation (All Indicators)	40	0.977	Reliable

The Cronbach's Alpha value of 0.977 indicates very high internal consistency. This value suggests that the instrument items are strongly interrelated and reliably measure the same underlying construct within the Product Operation dimension of the McCall model. Therefore, the questionnaire is statistically suitable for subsequent weight normalization and McCall-based quality computation.

Expert Agreement Analysis

To evaluate the consistency of expert judgments, an inter-rater agreement analysis was conducted by comparing the importance ratings assigned by the two experts across all indicators. The analysis showed that both experts assigned identical ratings to 33 out of 40 indicators, resulting in an agreement rate of 82.5%. Given the exploratory nature of the weighting assignment process and the limited number of experts involved, percentage agreement was considered sufficient to assess practical consistency between expert judgments.

The remaining differences occurred primarily between adjacent rating levels, particularly between scores 4 and 5, indicating minor variations in prioritization rather than fundamentally conflicting assessments. Therefore, the expert weighting results were considered sufficiently consistent to derive representative indicator weights for the subsequent comparative computational evaluation. Consequently, the averaged expert ratings were considered sufficiently reliable for subsequent weight normalization and McCall-based quality computation.

Because the purpose of the expert assessment in this study was to generate computational indicator weights rather than to establish or validate a psychometric measurement instrument, more advanced inter-rater reliability statistics, such as Cohen's Kappa or the Intraclass

Correlation Coefficient (ICC), were not considered necessary. Percentage agreement was deemed sufficient to demonstrate practical consistency between the two experts for the subsequent comparative weighting analysis.

Stability of Criteria Values

To evaluate the stability of the criteria values (c) derived from the 166 respondent dataset, descriptive statistical analysis was conducted for each McCall Product Operation variable. The analysis included mean, standard deviation, and standard error measurements to assess the consistency of the computed criteria values.

Table 4. Stability Analysis of Criteria Values

Variable	Mean	Std. Deviation	Std. Error
Correctness	4.4867	0.5709	0.0443
Reliability	4.3542	0.6212	0.0482
Efficiency	4.4608	0.6525	0.0506
Integrity	4.5783	0.6312	0.0490
Usability	4.5075	0.6311	0.0490

The results indicate relatively low standard error values ranging from 0.044 to 0.051 across all variables. These findings suggest that the computed mean criteria values remain sufficiently stable for comparative computational evaluation despite the respondent count being lower than the Slovin-estimated minimum sample size. Furthermore, the consistency of mean values across variables indicates that the dataset provides adequate representational stability for examining the influence of alternative weighting schemes on Total Quality outcomes.

Comparative Results of Total Quality Across Weighting Schemes

This section presents a comparative analysis of the Total Quality scores generated under three alternative weighting schemes within the McCall model. Each scheme was evaluated using both floor and ceiling rounding adjustments to assess computational consistency. The criteria values (c), derived from student responses, were kept constant across all schemes to ensure that any variation in the final results was attributable solely to differences in weight normalization intervals. Table 5 summarizes the Total Quality results obtained from each weighting scheme.

Table 5. Total Quality Results Obtained from Each Weighting Scheme

Scheme	Weight Interval	Total Quality		Interpretation		Floor-Ceiling Difference
		Floor	Ceil	Floor	Ceil	
1	0.1 - 0.5	89.4658%	89.4635%	Highly Qualified	Highly Qualified	0.0023%
2	0.1 - 0.4	71.7424%	71.7403%	Qualified	Qualified	0.0021%
3	0.4 - 0.8	89.4619%	89.4604%	Highly Qualified	Highly Qualified	0.0015%

The results indicate that Scheme 1 (0.1-0.5) and Scheme 3 (0.4-0.8) produce nearly identical Total Quality scores, both exceeding 89%, which fall within the “Highly Qualified” category. In contrast, Scheme 2 (0.1-0.4) yields a substantially lower Total Quality score of approximately 71%, categorized as “Qualified.” The variation between weighting schemes exceeds 17 percentage points, whereas rounding adjustments produce differences below 0.003%. This difference is substantial in interpretative terms, as it leads to a change in the quality classification category. Although the evaluated software system, criteria values, and aggregation structure remain unchanged, the selection of a compressed weight interval (0.1-0.4) results in a noticeable decrease in the overall quality score and alters the final classification. As a consequence, high-performing indicators contribute less proportionally to the final aggregate score under the compressed scheme.

From a practical standpoint, this implies that two evaluators assessing the same system using identical data but different weight normalization intervals may arrive at different quality classifications. In institutional or organizational contexts, such discrepancies could influence strategic decisions, including system improvement prioritization, accreditation reporting, or quality benchmarking. Overall, the comparative results confirm that weight interval selection has a substantially greater influence on Total Quality outcomes than rounding adjustments.

Therefore, standardization of weighting schemes is essential to ensure consistency and comparability in McCall-based software quality evaluations.

The similarity between Scheme 1 and Scheme 3 can be attributed to the preservation of relative weight proportions after normalization within the hierarchical aggregation process. Although the absolute weight ranges differ, the relative ordering and proportional contribution of indicators remain largely unchanged. Consequently, the resulting weighted aggregation produces nearly identical Total Quality scores despite differences in the original normalization interval.

Practical Impact of Weight Interval Selection

The findings demonstrate that the selection of weight intervals significantly affects the final interpretation of software quality within the McCall framework. Although the evaluation data and aggregation structure remain constant, modifying the normalization range of indicator weights produces materially different Total Quality outcomes. The compressed weighting scheme (0.1-0.4) yields a Total Quality score approximately 17 percentage points lower than the moderate (0.1-0.5) and expanded (0.4-0.8) schemes. More importantly, this reduction leads to a shift in the classification category from "Highly Qualified" to "Qualified." From a practical perspective, such a shift is not merely numerical but interpretative, potentially influencing managerial decisions, performance reporting, and quality benchmarking.

For example, a university evaluating the same learning management system using Scheme 1 may classify the software as "Highly Qualified", whereas another evaluator applying Scheme 2 could classify the same system only as "Qualified." Although the underlying evaluation data remain identical, the resulting interpretation may influence software improvement priorities, accreditation documentation, and institutional benchmarking decisions. In institutional contexts, software quality evaluations are often used to support strategic decisions such as system enhancement prioritization, internal audits, accreditation documentation, or funding justification. If different weighting intervals are applied without standardization, identical systems evaluated using the same criteria values may receive different quality classifications. This inconsistency may reduce comparability across studies, institutions, or evaluation periods.

Furthermore, the compressed interval reduces the dispersion between indicator weights, limiting the proportional influence of high-priority indicators in the additive aggregation process. As a result, even when certain indicators are rated as highly important by experts, their relative impact on the final score becomes constrained. Conversely, moderate and expanded intervals preserve differentiation among indicator priorities, allowing expert judgments to exert stronger influence on the aggregated outcome. These findings highlight the necessity of explicitly defining and standardizing weighting schemes when applying the McCall model in practice. Without such standardization, variations in weight normalization intervals may introduce interpretative bias, affecting the reliability and comparability of software quality assessments.

Rounding Stability and Its Practical Relevance

In addition to examining the influence of weight interval selection, this study also evaluates the impact of rounding adjustments, specifically floor and ceiling, on the final Total Quality score. Across all three weighting schemes, the numerical differences between floor and ceiling computations are extremely small, ranging from 0.0015% to 0.0023%. These differences are negligible in practical terms and do not alter the resulting quality classification in any scheme. Both rounding approaches consistently produce identical interpretative categories (e.g., "Highly Qualified" or "Qualified") within each weighting scheme. This indicates that the McCall weighted aggregation model demonstrates high computational stability with respect to rounding adjustments.

From a practical standpoint, this finding has important implications. In real-world applications, evaluators may apply different rounding conventions depending on software tools, spreadsheet configurations, or reporting standards. The results of this study suggest that such minor computational differences are unlikely to meaningfully affect the final evaluation outcome. Therefore, concerns regarding floor versus ceiling rounding appear to be less critical

than the choice of weight interval itself. Moreover, the smallest rounding difference was observed under the expanded weighting scheme (0.4-0.8), further indicating its numerical stability. This suggests that wider weight dispersion may slightly reduce rounding-induced variance in hierarchical aggregation processes. Overall, while rounding strategy has minimal influence on Total Quality results, weight normalization interval selection plays a significantly larger role in shaping the final interpretation of software quality. For practitioners, this implies that standardization efforts should prioritize weight interval definition rather than rounding conventions.

Recommended Weighting Scheme for Applied McCall Evaluations

Based on the comparative analysis, Scheme 1 (0.1-0.5) and Scheme 3 (0.4-0.8) demonstrate practically equivalent classification consistency and computational stability. Both schemes produce "Highly Qualified" outcomes with only negligible differences in rounding variance. Although Scheme 3 produces slightly lower rounding-induced variance compared to Scheme 1, the empirical difference remains marginal. Therefore, the findings do not support a definitive claim that Scheme 3 is universally superior. Instead, the results suggest that both moderate and expanded weighting intervals are more suitable for preserving interpretative consistency than compressed intervals such as Scheme 2. However, practitioners should also consider that the relatively high lower bound adopted in Scheme 3 may reduce the distinction between indicators assigned the lowest and highest importance levels. Consequently, the selection of weighting intervals should remain consistent with the intended evaluation context.

In contrast, Scheme 2 (0.1-0.4) produces substantially lower Total Quality outcomes and changes the resulting classification category. This indicates that compressed weight dispersion may reduce the amplification effect of highly prioritized indicators within the additive aggregation structure of the McCall model. Accordingly, practitioners and researchers should be aware that compressed weighting intervals may produce substantially lower Total Quality outcomes and may affect quality classification consistency. More importantly, weighting schemes should be explicitly documented and standardized to ensure comparability across software quality evaluations.

Study Limitations and Future Work

Despite the meaningful findings obtained in this study, several limitations should be acknowledged.

1. First, the student sample size, although sufficient for computational comparison and model application, was lower than the ideal number estimated using the Slovin formula. While the primary objective of this study was not population generalization but comparative weighting analysis, future research may increase respondent participation to enhance statistical representativeness and strengthen external validity.
2. Second, this study focused exclusively on the Product Operation dimension of the McCall model. The Product Revision and Product Transition dimensions were not included in the analysis. Incorporating all three dimensions in future studies may provide a more comprehensive evaluation of weighting sensitivity across the full McCall framework.
3. Third, only three predefined weighting intervals were examined (0.1-0.5, 0.1-0.4, and 0.4-0.8), based on patterns identified in prior empirical studies. Other normalization strategies or alternative dispersion configurations may produce different stability characteristics. Future research could explore adaptive weighting ranges, nonlinear transformations, or comparative analysis with other quality models such as ISO 25010 to assess cross-model sensitivity behavior.
4. Fourth, the case study was limited to a single web-based academic system (Simponi). Although the computational structure of the McCall model remains consistent across systems, applying the same experimental design to other types of software—such as enterprise systems, mobile applications, or critical systems—may provide broader insights into the generalizability of the findings.

Future studies may also integrate statistical sensitivity analysis techniques or simulation-based approaches to further examine how variations in expert weight assignments influence aggregated quality scores. Such extensions would strengthen methodological standardization efforts in McCall-based software quality evaluation.

CONCLUSIONS

This study demonstrates that weight normalization is not merely a technical preprocessing step in McCall-based software quality measurement but a methodological decision that can materially influence the resulting quality assessment. Under identical criteria values, evaluation objects, and aggregation procedures, the 0.1–0.5 and 0.4–0.8 weighting schemes produced comparable Total Quality scores above 89% and consistently classified the evaluated system as "Highly Qualified." In contrast, the 0.1–0.4 scheme reduced the Total Quality score to approximately 71% and changed the classification to "Qualified." These findings indicate that differences in weight normalization intervals can lead to substantively different interpretations of the same software system. By comparison, floor and ceiling rounding produced differences below 0.003% across all scenarios and did not alter any quality classification, indicating that rounding conventions have negligible practical influence relative to weight interval selection.

The findings provide empirical support for greater methodological transparency and consistency in McCall-based software quality evaluation. Researchers and practitioners should explicitly report the weight normalization procedure used and apply it consistently when comparisons are made across systems, institutions, or evaluation periods. Rather than identifying a universally optimal weighting interval, this study shows that alternative normalization schemes may generate different evaluation outcomes and therefore should be justified according to the intended analytical context. The contribution of this study lies in demonstrating the sensitivity of hierarchical McCall aggregation to weight normalization and in highlighting its implications for the comparability and reproducibility of software quality assessments. Nevertheless, the findings should be interpreted within the scope of a single academic web application, the Product Operation dimension, three predefined normalization intervals, and two expert assessors. Future research should replicate the sensitivity analysis across different software domains, larger and more diverse expert panels, the complete McCall framework, and broader normalization strategies, including simulation-based sensitivity analysis, to establish more generalizable recommendations for weighting standardization..

ACKNOWLEDGMENT

The authors gratefully acknowledge Universitas Multi Data Palembang for funding this research through the Internal Research Grant Scheme, and for supporting its implementation. Appreciation is also extended to the software engineering experts who provided indicator-importance assessments and to the student respondents who participated in the evaluation of the Simponi system. Their contributions were essential to the completion of the data collection and software quality analysis.

AUTHOR CONTRIBUTION STATEMENT

AF conceptualized the study, developed the research design, performed data analysis, conducted the weighting experiments, interpreted the results, and drafted the manuscript. DF assisted in data collection, including distributing and administering questionnaires to experts and student respondents, and contributed to manuscript review and refinement. All authors approved the final version of the manuscript.

AI DISCLOSURE STATEMENT

The authors used an AI-based language model (ChatGPT) during the preparation of this manuscript for language refinement, structural improvement, and clarity enhancement. All analytical decisions, research design, data processing, and interpretation of results were conducted independently by the authors. The authors thoroughly reviewed and edited the content and take full responsibility for the final version of the publication.

REFERENCES

[1] D. Sofronas, D. Margounakis, M. Rigou, E. Tambouris, and T. Pachidis, "SQMetrics: An Educational Software Quality Assessment Tool for Java," *Knowledge*, vol. 3, no. 4, pp. 557–599, Sep. 2023, doi: 10.3390/knowledge3040036.

[2] A. V. Kopyltsov, "Selection of metrics in software quality evaluation," *Journal of Physics: Conference Series*, vol. 1515, no. 3, p. 032018, Apr. 2020, doi: 10.1088/1742-6596/1515/3/032018.

[3] Y. Thamilarasan, R. R. R. Ikram, M. Osman, and L. Salahuddin, "A Review on Software Quality Models for Learning Management Systems," *Journal of Advanced Research in Applied Sciences and Engineering Technology*, vol. 32, no. 2, pp. 203–221, Oct. 2023, doi: 10.37934/ARASET.32.2.203221.

[4] J. P. Miguel, D. Mauricio, and G. Rodríguez, "A Review of Software Quality Models for the Evaluation of Software Products," *International Journal of Software Engineering & Applications*, vol. 5, no. 6, pp. 31–53, Nov. 2014, doi: 10.5121/ijsea.2014.5603.

[5] P. Nistala, K. V. Nori, and R. Reddy, "Software Quality Models: A Systematic Mapping Study," in *2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*, IEEE, May 2019, pp. 125–134. doi: 10.1109/ICSSP.2019.00025.

[6] W. F. R. Assidiq, F. W. Putro, and A. M. Amri, "Evaluating Software Quality in a Point of Sales System in a Fast-Food Restaurant Using the McCall Model," *Jurnal Teknik Informatika (Jutif)*, vol. 6, no. 4, pp. 2317–2330, Aug. 2025, doi: 10.52436/1.jutif.2025.6.4.4860.

[7] S. N. Adillah, T. K. Ahsyar, Syaifullah, Anofrizen, and A. Marsal, "Quality Evaluation of The SITASI Final Project System using Selected McCall Software Quality Factors," *Edumatic: Jurnal Pendidikan Informatika*, vol. 9, no. 2, pp. 512–521, Aug. 2025, doi: 10.29408/edumatic.v9i2.30675.

[8] S. A. Tabrizi, "Engineering-Centered Quality Assurance in Software Localization: Aligning GUI and MT Evaluation with the McCall Software Quality Model," *Emerging Technologies and Engineering Journal*, vol. 2, no. 2, pp. 69–80, Oct. 2025, doi: 10.53898/etej2025225.

[9] Z. A. H. Alnaish and S. O. Hasoon, "Software Quality Measurement Analysis based on Techniques, Criteria, Metrics, Models, and Datasets," in *2022 8th International Conference on Contemporary Information Technology and Mathematics (ICCITM)*, IEEE, Aug. 2022, pp. 176–181. doi: 10.1109/ICCITM56309.2022.10031990.

[10] J. K. Mulia and R. Sutomo, "Evaluation Of The Quality Of The Online Sales Application Of Light Steel Material Using The Mccall Quality Factors Method," *International Journal of Science, Technology & Management*, vol. 4, no. 5, pp. 1244–1250, Sep. 2023, doi: 10.46729/ijstm.v4i5.930.

[11] E. Ronchieri and M. Canaparo, "Assessing the impact of software quality models in healthcare software systems," *Health Systems*, vol. 12, no. 1, pp. 85–97, Jan. 2023, doi: 10.1080/20476965.2022.2162445.

[12] A. Farisi, R. Teguh, and R. Lestari, "Analisis Kualitas Sistem Informasi Haji Terpadu Menggunakan Metode McCall," *JOINTECS (Journal of Information Technology and Computer Science)*, vol. 7, no. 2, p. 83, May 2022, doi: 10.31328/jointecs.v7i2.3725.

[13] S. Budi, W. Gata, M. Noor, S. Pangabean, and C. S. Rahayu, "News Portal Website Measurement Analysis Using ISO/IEC 25010 And McCall Methods," *Journal of Applied Engineering and Technological Science (JAETS)*, vol. 4, no. 1, pp. 273–285, Oct. 2022, doi: 10.37385/jaets.v4i1.1094.

[14] A. Riswanto, R. P. Sari, and N. Mutiah, "Measuring the Quality of District Government Websites Only Using the McCall and EUCS Method (End User Computing Satisfaction)," *JURTEKSI (Jurnal Teknologi dan Sistem Informasi)*, vol. 11, no. 1, pp. 123–130, Dec. 2024, doi: 10.33330/jurteksi.v11i1.3206.

[15] B. A. Herlambang, Muhtarom, and M. S. Zuhri, "Analysis of Edu-Smart Learning Management System Software," *KnE Social Sciences*, vol. 7, no. 19, pp. 183–191, Dec. 2022, doi: 10.18502/kss.v7i19.12440.

[16] F. Panca Juniawan *et al.*, "E-Voting Software Quality Analysis with McCall's Method," in *2020 8th International Conference on Cyber and IT Service Management (CITSM)*, IEEE, Oct. 2020, pp. 1–5. doi: 10.1109/CITSM50537.2020.9268854.

[17] D. Setiadi, T. Sumitra, A. Karim, and Ritzkal, "Software Quality Measurement Analysis on Academic Information Systems," *Ingenierie des Systemes d'Information*, vol. 29, no. 4, pp. 1453–1460, Aug. 2024, doi: 10.18280/isi.290418.

[18] A. J. Al-Nawaiseh, F. B. Ahmad, and S. J. Al-Nawaiseh, "A Critical Review of Software Quality Models in Academic Information System," *Journal of Theoretical and Applied Information Technology*, vol. 102, no. 24, pp. 9048–9059, Dec. 2024.

[19] A. M. Saleh and O. Enaizan, "Framework for selecting the best software quality model for a smart health application based on intelligent approach," *Bulletin of Electrical Engineering and Informatics*, vol. 12, no. 3, pp. 1711–1727, Jun. 2023, doi: 10.11591/eei.v12i3.4945.

[20] Y. Andriyani, J. A. Dewana, and I. D. Id, "Implementasi McCall's Framework dalam Pengujian Kualitas Perangkat Lunak (Studi Kasus Portal Kuliah Kerja Nyata Universitas Riau)," *Jurnal Teknik Informatika*, vol. 13, no. 2, pp. 201–212, Feb. 2021, doi: 10.15408/jti.v13i2.16986.

[21] C. Juliane, R. Dzulkarnaen, and W. Susanti, "Metode McCall's untuk Pengujian Kualitas Sistem Informasi Administrasi Tugas Akhir (SIATA)," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 3, no. 3, pp. 488–495, Dec. 2019, doi: 10.29207/resti.v3i3.1170.

[22] A. Farisi, Dafid, and R. Teguh, "Analisis Metode Pengukuran Kualitas Perangkat Lunak: Sebuah Tinjauan Literatur Sistematis," *SATESI: Jurnal Sains Teknologi dan Sistem Informasi*, vol. 4, no. 1, pp. 10–16, Apr. 2024, doi: 10.54259/satesi.v4i1.2551.

[23] S. M. Rubejes-Silva, "Bridging the Gap Between Universities and Alumni: A User-Centered Evaluation of a Digital Alumni Engagement Platform," *Journal of Innovative Technology Convergence*, vol. 6, no. 2, pp. 49–58, Jun. 2024, doi: 10.69478/JITC2024v6n002a05.

[24] A. Farisi and H. Saputra, "Analisis Kualitas Sistem Informasi Menggunakan Metode McCall: Studi Kasus SPON MDP," *Techno.Com*, vol. 21, no. 2, pp. 237–248, May 2022, doi: 10.33633/tc.v21i2.5970.

[25] Z. A. Santosa, M. Y. P. Chusnani, R. Purbaningtyas, and S. A. Wulandari, "Implementasi Profile Matching untuk Mengukur Kualitas Website Sistem Informasi Desa Sidokerto Menggunakan Model McCall," *Jurnal Masyarakat Informatika*, vol. 15, no. 1, pp. 67–80, May 2024, doi: 10.14710/jmasif.15.1.63273.

[26] Malahayati and E. Nadeak, "Implementasi Mc Call dalam Pengukuran Kualitas SIAKAD (Sistem Informasi Akademik)," *Teknomatika*, vol. 13, no. 01, pp. 63–69, 2023.

[27] F. Sulaiman, N. Suarna, and Iin, "Pengukuran Kualitas Perangkat Lunak Sistem Informasi Pengarsipan Dokumen Laporan Jalan Tol Menggunakan Metode Mccall," *INFOTECH journal*, vol. 8, no. 1, pp. 34–40, Mar. 2022, doi: 10.31949/infotech.v8i1.2234.

[28] A. Andrianti, "Pengukuran Kualitas Aplikasi Rekap Indikator Mutu Harian RS Bhayangkara Jambi Menggunakan Metode McCall," *Jurnal Ilmiah Media Sisfo*, vol. 14, no. 1, pp. 24–34, Apr. 2020, doi: 10.33998/mediasisfo.2020.14.1.716.

[29] F. Navalia and U. K. Nisak, "Evaluasi Kualitas Rekam Medis Elektronik Menunjang Kualitas Data Pasien di Rumah Sakit Umum Kota Mojokerto," *Jurnal Ilmiah Pamenang*, vol. 6, no. 2, pp. 148–157, Dec. 2024, doi: 10.53599/jip.v6i1.252.

[30] R. O'Shea, P. Deeney, E. Triantaphyllou, L. Diaz-Balteiro, and S. Armagan Tarim, "Weight stability intervals for multi-criteria decision analysis using the weighted sum model," *Expert Systems with Applications*, vol. 296, p. 128460, Jan. 2026, doi: 10.1016/j.eswa.2025.128460.

[31] J. Więckowski and W. Sałabun, "Sensitivity analysis approaches in multi-criteria decision analysis: A systematic review," *Applied Soft Computing*, vol. 148, p. 110915, Nov. 2023, doi: 10.1016/j.asoc.2023.110915.

[32] J. W. Creswell and J. D. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, 5th ed. Los Angeles, CA, USA: SAGE Publications, 2018.

[33] E. Triantaphyllou, *Multi-criteria Decision Making Methods: A Comparative Study*, vol. 44. in Applied Optimization, vol. 44. Boston, MA: Springer US, 2000. doi: 10.1007/978-1-4757-3157-6.
[34] T. Yamane, *Statistics: An Introductory Analysis*, 2nd ed. New York, NY, USA: Harper and Row, 1967.
[35] C. G. Sevilla, J. A. Ochave, T. G. Punsalan, B. P. Regala, and G. G. Uriarte, *Research Methods*, Revised Edition. Quezon City, Philippines: Rex Bookstore, 2007.
[36] J. F. Hair, *Multivariate Data Analysis*, 8th ed. Andover, UK: Cengage, 2019.
[37] K. S. Taber, "The Use of Cronbach's Alpha When Developing and Reporting Research Instruments in Science Education," *Res Sci Educ*, vol. 48, no. 6, pp. 1273–1296, Dec. 2018, doi: 10.1007/s11165-016-9602-2.