2024 14th International Conference on Software Technology and Engineering (ICSTE) | 979-8-3503-7895-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICSTE63875.2024.00009

Defining Metrics for Evaluating Transparency in Software Engineering Process

Paulinus Ofem
Department of Computer Science
North-West University
Mmabatho, South Africa
25831097@student.g.nwu.ac.za

Bassey Isong
Department of Computer Science
North-West University
Mmabatho, South Africa
bassey.isong@nwu.ac.za

Francis Lugayizi
Department of Computer Science
North-West University
Mmabatho, South Africa
francis.lugayizi@nwu.ac.za

***Abstract*—Transparency has become an important non-functional quality requirement of software products and processes. In software engineering, stakeholders, especially software developers and requirements engineers, do not fully know how to benefit from transparency because of the lack of suitable measures and metrics that can be utilised in transparency evaluation and its improvement. Few existing metrics are also poorly defined and, therefore, poorly understood. This paper proposes measures and metrics for evaluating and improving transparency. By integrating the "define-your-own-model," the Goal Question Metric and the model-driven engineering paradigms, we provide a comprehensive set of transparency quality measures and metrics that can be used in a transparency improvement programme.**

***Index Terms*—transparency, transparency factors, transparency metrics**

I. Introduction

Transparency in software engineering (SE) and information systems (IS) is an evolving concept that addresses stakeholders' demands for openness in processes, information, and software. Initially emerging from concerns over accessibility and process clarity, transparency is crucial for stakeholders to understand a software system's functions fully. Despite its recognition as a non-functional requirement (NFR) [1] for enhancing communication [2], [3], challenges in measuring and achieving transparency persist in academic research and industry practice. The concept's breadth encompasses various definitions, focusing on aspects like anomaly explanation and complexity masking, yet a unified theoretical framework for its evaluation remains underdeveloped.

Apart from the challenge of transparency measurement, efforts to embed transparency within the software development life cycle (SDLC) highlight the significance of effective communication between stakeholders, mediated through various software artefacts. Poor communication is a key factor in project failures [2], suggesting a gap in research and practice regarding transparency's role in facilitating better information exchange. A comprehensive approach to transparency should account for the content and the channels of communication, aiming for a shared understanding that can enhance project success and stakeholder productivity [4]. The push for transparency, therefore, is not just about making information available but ensuring that it fosters effective, efficient communication that addresses the multifaceted concerns of all project participants.

Measurement is an everyday life activity vital to scientific and engineering disciplines. Establishing measures facilitates the acquisition of information that can be applied to developing theories and models [5]. To this end, several measurements or metric models have been proposed in the literature. Existing models do not always cater to varying measurement needs and contexts. This necessitates developing targeted measurement models that fulfil an organisation's interest [5], [6]. To define a transparency measurement model, we combine and adopt Gilb and Finzi's "define-your-own model" [6] approach and the Goal Question Metric (GQM) methodology [7]. Therefore, the overarching goal of this paper is to establish transparency evaluation metrics. This paper is organised as follows: Section II provides related works. Section III presents the research objective and methodology. The proposed metrics are provided in Section IV, and in Section V, we conclude the study and provide future work.

II. Related Work

Transparency has become a critical quality factor that software project stakeholders, especially requirements engineers and developers, must consider during software development and after software deployment. Transparency commonly refers to fully opening or disclosing information and processes to stakeholders. In SE, transparency refers to the extent of openness of software products and process information to stakeholders [4]. Professor Mylopolous opines that "transparency is an interesting quality because it makes it necessary to attach requirement models to software" [1]. We consider requirements models to be stakeholders' transparency requirements that must be fulfilled to make the software or process transparent.

There is an apparent lack of comprehensive research on transparency in SE, as evidenced by a systematic literature review (SLR) conducted as part of our transparency research programme in [8]. Due to this under-research, transparency is often treated as a heuristic or anomaly without solid theoretical or empirical foundations, leading to stakeholder misunderstandings, particularly in SE. Consequently, there's limited research and scarce publications on the topic. Prior

979-8-3503-7895-5/24/$31.00 ©2024 IEEE
DOI 10.1109/ICSTE63875.2024.00009

Authorized licensed use limited to: North West University. Downloaded on January 08,2026 at 06:52:55 UTC from IEEE Xplore. Restrictions apply.

works (e.g., [1], [9]–[18]) have focused on understanding transparency from the perspectives of fully functional software, data, and business processes, proposing models like the transparency ladder but lacking definitive measures for achieving transparency, especially in SE. SE studies (e.g., [2], [3], [19], [20]) did not provide a well-defined measurement for transparency. However, they offer good hindsight and background that supports further research on how SEP can achieve transparency. The studies have not equally provided transparency requirements that developers and end-users may use to assess the degree of transparency achievement. By adopting a formal approach, we believe the outcome of our study can shed more light on transparency measurement.

In order to bridge these gaps, in [21], we introduced a conceptual framework to understand, evaluate, and improve transparency in SEP. This framework identifies critical components such as stakeholders, their transparency requirements, transparency factors, metrics, and improvement schemes. Because we consider transparency achievement a requirements engineering activity, we elicited initial transparency requirements to be considered in the early software development phase in [4]. The authors populated the framework stakeholders' transparency component, emphasising the need for a systematic approach to conceptualise and measure transparency. This paper aims to populate the metrics construct of our framework.

III. Methodology

We integrated and adopted the Goal Question Metric (GQM) [7] paradigm, the "Define-Your-Own" model [5], [6], and the model-driven engineering [22] approach for the development of a transparency high-level quality model. The GQM is a framework for developing software metrics. The goal in GQM defines the overarching target a project aims to achieve. A measurement goal can be defined for objects such as products and processes. The questions define the way an assessment of a goal is performed. The metric is data associated with each question that quantitatively answers the question. The GQM assumes that purposeful measurement by organisations can be achieved when goals are specified. These goals should be traceable to data that defines the goals.

A. Research Objective

The primary objective of this paper is to identify and define appropriate transparency measures and metrics for evaluating and improving transparency in SEP.

IV. Goal, questions and metrics for transparency metrics model

This section implements GQM and the "Define-Your-Own" model approach in developing a high-level quality transparency metrics model. Again, in this paper, we focus on software product-related transparency metrics.

According to [23], the execution of a measurement activity requires precise and testable goals to be clearly defined. The clear goals we set will enable us to define the dependent and independent attributes of a measurement model clearly. The goal and questions for our metrics are derived from the knowledge gained from the two SLR activities we performed and a general literature review as part of our work in [4], [8], [21].

Given the limits of human cognition and the reliance on human judgement, it is reasonable to keep the number of metrics to a manageable size. Also, one of the major findings from the SLR we performed is that implementing transparency poses several challenges. These challenges include the cost of implementing transparency in SEP [1]. To arrive at a set of metrics that, to some extent, overcomes some of the challenges, we established the following requirements:

- **R1:** The metric should provide a clear indication of the transparency of an SDLC product in terms of its reflective indicators. The achievement of R1 ensures that the metric assesses transparency when used to evaluate an SDLC product.
- **R2:** The metric should provide logical outcomes when applied to each SDLC product during SEP. The attainment of R2 ensures that the metric applies to all SDLC products and is, therefore, not restricted to only one.
- **R3:** The metric should be independent of the software development methodology or language used in creating the SDLC product.
- **R4:** The metric should be easily applied without encumbering the development process regarding budget and delivery schedule.

The measures and metrics for transparency will be evaluated against these established requirements.

A. Goal

Purpose: *To evaluate, predict, compare, and improve*
Quality focus: *the transparency (in relation to effectiveness and efficiency of communication, software artefact maintainability and stakeholders' productivity) of*
Object of study: *software product (software requirements specifications)*
Viewpoint: *from the viewpoint of developers, requirements engineers, and the researcher in the context of the*
Environment: *transparency of SDLC products*

Goal: To evaluate, predict, compare, and improve the transparency of software artefacts via accessibility, usability, understandability, modifiability, and usability transparency factors from the perspective of the researcher, requirements engineers, developers, and in the context of SDLC products.

Owing to its complex nature, it is challenging to measure transparency directly. However, it can be indirectly measured via quality factors that help explain and achieve its measurement. In [1], these attributes are known as soft goals because they help explain and achieve transparency. To identify the key transparency factors and existing transparency metrics, we conducted an SLR, as previously stated. These critical

Authorized licensed use limited to: North West University. Downloaded on January 08,2026 at 06:52:55 UTC from IEEE Xplore. Restrictions apply.

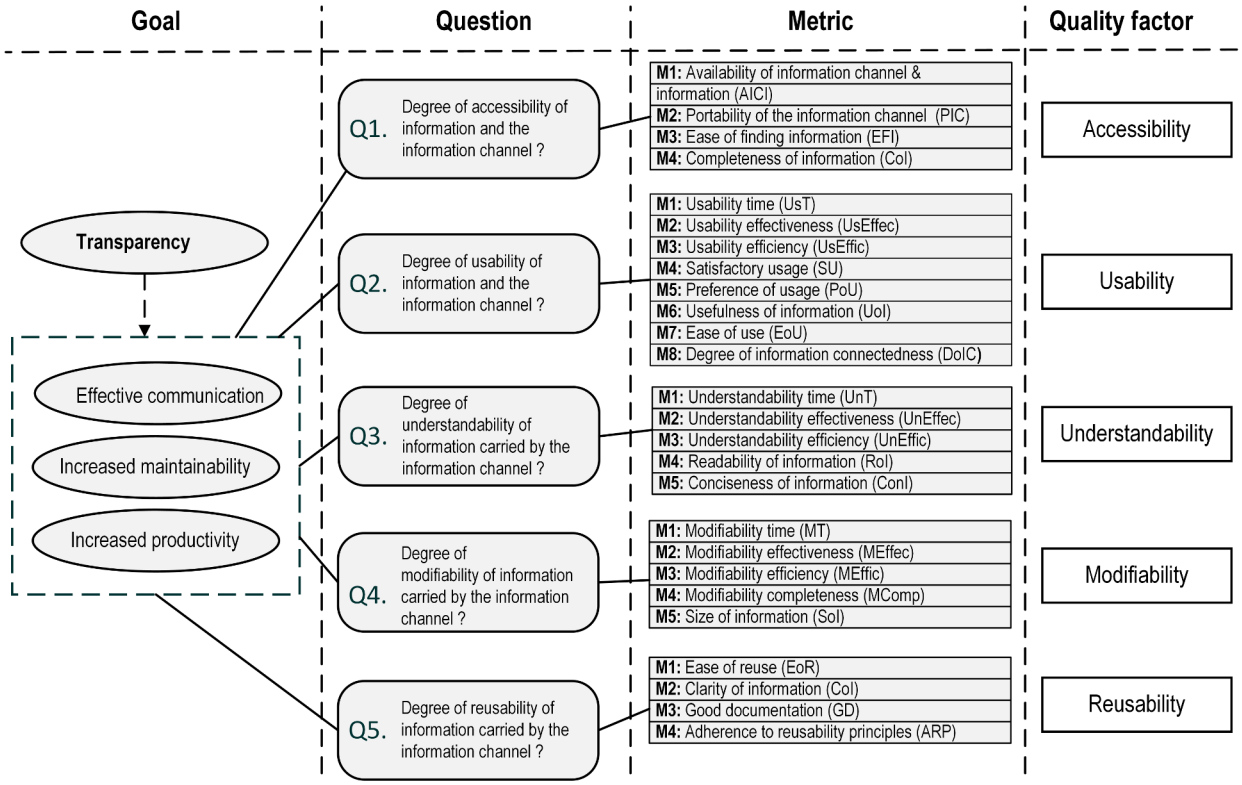

Fig. 1. High-level GQM model for assessing transparency

transparency factors were carefully chosen for their generality for adoption in this study.

B. Questions

Based on the goal definition and the key transparency factors we identified from the literature, we propose the following questions:

Q1: What is the degree of accessibility of a software artefact?
Q2: What is the degree of usability of a software artefact?
Q3: What is the degree of understandability of a software artefact?
Q4: What is the degree of modifiability of a software artefact?
Q5: What is the degree of reusability of a software artefact?

Professor Mylopoulous opined that "transparency is an interesting quality because it makes it necessary to attach requirement models to software" (*personal communications cited in* [1]). This implies that software is transparent when transparent requirement models are attached to the software. In light of this, determining stakeholders' SEP transparency concerns or requirements remains a reasonable and crucial step in transparency implementation. Following the first step in IEEE's [24] methodology, the SEP transparency requirements were generated as reported in our previous work [4]. Furthermore, transparency quality factors, measures, and metrics are needed to measure these requirements. Therefore, transparency metrics can measure the degree to which each specified requirement is achieved via its corresponding quality factor.

C. The Proposed Candidate Metrics for Transparency

The last phase in the GQM modelling methodology is to determine and define the metrics for the measurement. Maximising existing data sources is considered a key factor in selecting metrics to be used [5]. Existing metrics, transparency attributes, and the proposed metrics were derived through an SLR. Our proposed quality GQM model for measuring transparency is presented in Fig.1. In the introductory section of this paper, a reference is made to what constitutes a communication channel. In this study, we consider SDLC products or artefacts as communication channels. We equally consider these channels to be information carriers. The communication channels, therefore, must have information-carrying capacity. In measuring transparency, we consider the information-carrying medium and the information itself. As indicated in Fig. 1, the goal is to measure transparency in the first instance. Secondly, evaluate the impact of transparency on effective and efficient communication, software maintainability, and stakeholders' productivity. The questions captured in the model enabled us to find measures that will potentially help us achieve the goal of measurement. The metrics are derived from

Authorized licensed use limited to: North West University. Downloaded on January 08,2026 at 06:52:55 UTC from IEEE Xplore. Restrictions apply.

quality factors recognised as having the potential to explain and achieve transparency. The full definition of each metric is presented in Section IV-E. As mentioned earlier, we present product-related transparency metrics in this paper.

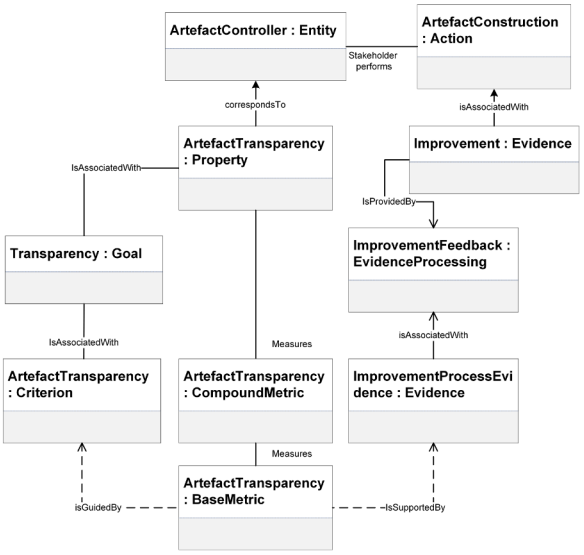

Fig. 2. A Generic model of transparency in software engineering

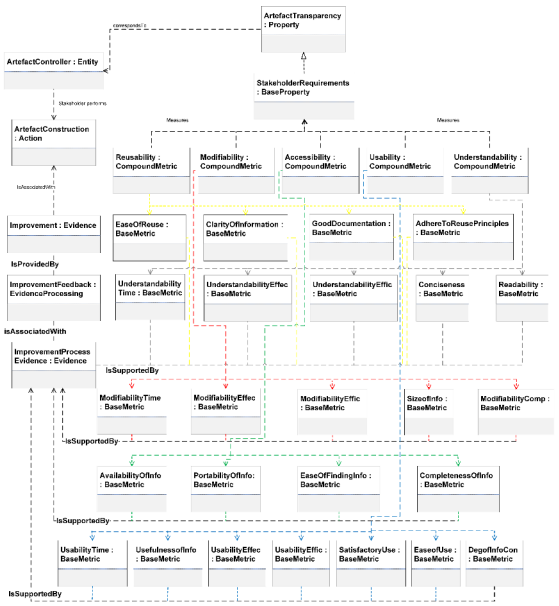

Fig. 3. The Specified model for transparency quality factors and metrics

D. Modelling Transparency - The MDE Approach

This section attempts to model transparency and its properties based on Fig. 2. The model presented in this section enables the understanding and description of transparency factors and metrics interrelationships from MDERA's [22] viewpoint. In producing these models, we adapted a meta-model [25] initially proposed for achieving transparency in the cloud. A similar adaptation is provided in the medical systems domain [17]. We briefly describe the original meta-model properties and how we adapt some of these properties in our current study.

As provided in [25], the first element of the meta-model is the *Goal*, which captures a high-level description of a modelled property. The *Property*, another element of the model, is a high-level concept, such as quality factors that characterise an entity. A property is equally the object of study. The *Property* element, being a top-level concept, could further be decomposed into a *BaseProperty* and a *CompoundProperty*. The *Entity* element performs actions that achieve the modelled property. The *Action* element refers to a process that is executed, which produces an effect on an Entity. The *Evidence* and *EvidenceProcessing* are the tangible and observable effects of actions on the property. *EvidenceProcessing* produces the *Evidence*. We, therefore, note that the procedure for gathering and analysing evidence is equally considered as vital as the evidence being processed. This implies that another piece of evidence would be associated with each process. The Metric element represents the methods that measure the property. A metric can be decomposed into a *BaseMetric* and *Compound-Metric*. The *Criterion* element represents a constraint to what the metric measures.

We next present how we adapted and explicated the original meta-model elements to model transparency in SEP context. In Fig. 2, we present a generic model of transparency in SE. Based on the generic model, we present a more specified model for the present study in Fig. 3. Fig. 3 indicates *BasesMetric* for the *Reusability* and *Understandability* compound metrics. Other base metrics can, however, be modelled in the same way.

Goal. The Goal element represents a high-level description of the property that is modelled. In this paper, transparency of software artefacts is the designated overall goal of the study. This is indicated in the generic model in Fig. 2.

Property. As indicated in Fig. 3, the major property of the model is *Transparency*. This property is defined as a software artefact. A metric for the property would, therefore, evaluate how transparent the artefact is during SEP. In Fig. 3, this property is further decomposed into sub-property of *StakeholdersRequirements*, as elements that inherit from the *BaseProperty*.

Entity and Action. As previously stated, a *property* is achieved by an *Entity*. An entity itself performs and *Action*, or is affected by an action. In the context of this study, the main goal is to make software artefacts transparent. Since what is evaluated are artefacts, we designate the ArtefactController as

Authorized licensed use limited to: North West University. Downloaded on January 08,2026 at 06:52:55 UTC from IEEE Xplore. Restrictions apply.

the entity. This entity indirectly performs actions within the *Action* element designated as *ArtefactConstruction*. Therefore, we subsume the entity's actions into the business process of constructing software artefacts. It is intended that stakeholders will perform activities that better model the property under consideration.

Criterion. This element refers to a constraint that might help the refinement of the property that is being measured. Since our focus is evaluating the transparency of software artefacts, we consider stakeholders' preferences. Also considered are object-oriented design techniques and reusability principles during the actual definition of the metrics. The criterion element is considered part of the goal and is, therefore, dealt with within the goal specification.

Evidence and evidence processing. Transparency is a high-level, complex, and multi-faceted concept; observing and measuring it is difficult. The implication is that any solution to transparency evaluation is difficult to generalise across different domains and contexts. Previously in [4], the authors elicited some transparency requirements that are applicable to SEP. These requirements represent the *Evidence* a stakeholder is expected to observe and measure with the corresponding transparency metric as indicated in Fig. 3. According to [1], transparency requirements that are fulfilled in the software are what make software transparent. The metrics, therefore, enable us to determine to what extent the requirements are attained. In the model, we model evidence as *ImprovementProcess*. The evidence is subsumed into the improvement process intended to make the artefact more transparent. In this paper, we do not cover the improvement process.

The *Evidence*, though associated with the *Action* performed by the *Entity*, is not produced by the *EvidenceProcessing* element. In Fig. 3, the *Improvement* represents evidence that the *Action* of artefact construction is in progress, and its outcome is informed by the *EvidenceProcessing* element. The relationship between the *EvidenceProcessing* and the Evidence element describes the nature of the improvement provided by the process. The improvement process could be diagnostic, preventive or corrective. These are part of the improvement mechanisms in the proposed evaluation framework. Again, these aspects are beyond the scope of this paper.

E. Description of Metrics

Transparency is an NFR, which can be realised in a software product or SEP [1], [3]. This paper focuses on the early software product quality metrics and measurements considered useful during the requirements engineering phase. We specifically take into consideration the quality metrics that have the potential to assess transparency quality factors. In addition, we consider other metrics and measures that can equally assess the quality factors of transparency. We adopt the metrics description and evaluation framework proposed in [26] and the property-based software engineering measurement [27] to fully define our metrics.

It is important to note that some proposed metrics are objective or subjective. Objective metrics are always desirable to have because they ensure consistency in measurement. According to [5], subjective measurements can be determined by the environment in which they are defined, and they are equally useful if their imprecision is understood. The subjectiveness is because communication in SE and other disciplines is largely a human endeavour. This view also applies to some transparency requirements, as demonstrated in previous studies [4], [8], [18], [21]. Given the multifaceted and complex nature of transparency and its orthogonality to software functioning, subjective metrics are sometimes necessary. Previous studies show that some transparency-improving factors or notions such as availability, relevance, simplicity, and understandability hint at the need for subjective measures because the transparency requirements to be evaluated, in some instances, are human-centred and not software-related attributes. Subjective measures are usually represented as either a nominal or ordinal scale. The nominal and ordinal measures are unsuited to the recommended property-based measurement framework. Also, some descriptive properties (e.g., metric attribute, attribute variability) proposed in [26] are not equally suited to ordinal and nominal measures. These measures are, therefore, not in conformity with formal properties and, thus, omitted in the description of some subjective metrics.

However, a measure is internally valid if it correctly represents a numerical characterisation of the attribute [5]. A theoretical validation is a basic form of validation to show that a measure measures the attribute it purports to measure [28]. Theoretical validation is considered a prerequisite to empirical validation [5]. The framework [28] enabled us to theoretically validate the objective metrics we proposed, especially the properties related to length and size. It also helped in cases where properties do not typically represent software properties but are reasonably applicable. Next, we outline the proposed transparency metrics under five main categories: accessibility, usability, understandability, modifiability and reusability. Because of the limited paper length, Tables I-V under the Appendix provide a complete definition of the proposed metrics.

Accessibility metrics
Metric 1: Availability of information channel and information (AICI)
Metric 2: Portability of the information channel (PIC)
Metric 3: Ease of finding information (EFI)
Metric 4: Completeness of information (ComI)

Usability metrics
Metric 1: Usability time (UsT)
Metric 2: Usability effectiveness (UsEffec)
Metric 3: Usability efficiency (UsEffic)
Metric 4: Ease of use (EoU)
Metric 5: Satisfactory usage (SU)
Metric 6: Preference of usage (PoU)
Metric 7: Usefulness of information (UoI)

Authorized licensed use limited to: North West University. Downloaded on January 08,2026 at 06:52:55 UTC from IEEE Xplore. Restrictions apply.

Metric 8: Degree of information connectedness (DIC)

Understandability metrics
Metric 1: Understandability time (UnT)
Metric 2: Understandability effectiveness (UnEffec)
Metric 3: Understandability efficiency (UnEffic)
Metric 4: Readability of information (RoI)
Metric 5: Conciseness of information (ConI)
Metric 6: Readability notion (RNotion)
Metric 7: Understandability notion (UnNotion)

Modifiability metrics
Metric 1: Modifiability time (MT)
Metric 2: Modifiability effectiveness (MEffec)
Metric 3: Modifiability completeness (MComp)
Metric 4: Modifiability efficiency (MEffic)
Metric 5: Size of information (SoI)

Reusability metrics
Metric 1: Ease of reuse (EoR)
Metric 2: Clarity of information (CoI)
Metric 3: Good documentation (GD)
Metric 4: Adherence to reusability principles (ARP)
Metric 5: Ease of understanding (EoU)

It is also worth noting that less objective or formal metrics dominate the metrics we have identified and proposed. This outcome aligns with the fact that measuring some quality factors, including those provided in existing quality models via objective metrics, is problematic. This situation justifies the use of subjective measures [5]. The nature of the metrics we have proposed is such that, in reality, they do not impact the functionality of software. This evidence supports the orthogonality of transparency as a quality that may not be fully measured in the traditional sense of measurement, as Leite and Cappelli [1] opine.

F. Validation of Quality Metrics

In SE, measures must undergo theoretical (i.e., to show that the measures measure what they intended to measure) validation. They also demand empirical (i.e., to demonstrate that the measures are useful in their relationship with other variables defined as part of a theory under investigation) validation [5]. The transparency factors and metrics have undergone face-validity checks as part of previous work [21]. Face-validity checks to see that the proposed constructs and metrics are logically sound and fit for purpose. A panel of six experts in SE reviewed the constructs and metrics. The review feedback enabled us to improve our definition of transparency metrics.

Concerning metrics's theoretical validation, we used Kaner and Bones' framework [26] to describe and validate the metrics. We have equally relied on a property-based validation framework [27] to evaluate the formal metrics. This is presented in Appendix A. Therefore, this evaluation approach is not considered for subjective metrics.

In addition, we considered theoretical validation by studying and applying a metrics validity scheme provided by Meneely *et al.* [29]. A total of 47 metrics validity criteria were provided. Of the original 47, we adopted a set of 25 that we deemed applicable and applied them in the current study. The criteria adopted are namely: *applicability, mathematical properties, definition validity, priori validity, actionability, scale validity, unit validity, usability, internal validity, content validity, construct validity, causal model validity, monotonicity, association, empirical validity, economic productivity, external validity, instrument validity, internal consistency, instrument validity, reliability, nonexploitability, product or process relevance, repeatability, underlying theory validity*.

Main Contribution
This paper's main contribution is adequately defining and theoretically validating transparency measures and metrics. Additionally, it models how these measures and metrics can help transparency evaluation and improvement in SEP. Given its continuous emergence as a concept, transparency still needs to be understood, especially when measuring its achievement. Our initial transparency measurement proposals are critical to its practical application in SEP.

V. Conclusion

We require well-defined metrics to assess transparency requirements. This paper focuses on the metrics component of a transparency evaluation framework. Because transparency may not be measured directly, we used an integrated approach to develop and theoretically validate metrics for measuring it. To measure transparency, we identified vital constructs of accessibility, usability, understandability, modifiability, and reusability from which the transparency metrics were derived. Thus, we provided a set of metrics for each construct that assesses the targeted construct. An experiment is necessary to empirically validate and demonstrate the usefulness of transparency constructs and metrics. The experiment was conducted, and the results were submitted for consideration for publication in a journal.

Appendix: Complete Description of Metrics and Measures for Evaluating Transparency

The complete description of the metrics is available at the following link.
https://drive.google.com/file/d/
1acLowuYbw1eUFAEMbzioox3qt-xL4rHL/view?usp=sharing

VI. Acknowledgements

This research is part of the first author's PhD study, which is supported by the Data Science and Computing research unit of the Department of Computer Science, North-West University, and the author's primary institution, the Federal University of Lafia, Nigeria, with support from the Tertiary Education Trust Fund.

Authorized licensed use limited to: North West University. Downloaded on January 08,2026 at 06:52:55 UTC from IEEE Xplore. Restrictions apply.

References

[1] J. C. S. d. P. Leite and C. Cappelli, "Software transparency," *Business & Information Systems Engineering*, 2010.

[2] Y.-C. Tu, C. Thomborson, and E. Tempero, "Illusions and perceptions of transparency in software engineering," in *18th Asia-Pacific Software Engineering Conference*. Ho Chi Minh city, Vietnam: IEEE, 2011, Conference Proceedings, pp. 365–372.

[3] Y.-C. Tu, E. Tempero, and C. Thomborson, "An experiment on the impact of transparency on the effectiveness of requirements documents," *Empirical Software Engineering*, vol. 21, pp. 1035–1066, 2016.

[4] P. Ofem, B. Isong, and F. Lugayizi, "Stakeholders' transparency requirements in the software engineering process," in *IECON 2022–48th Annual Conference of the IEEE Industrial Electronics Society*. Brussels, Belgium: IEEE, 2022, Conference Proceedings, pp. 1–6.

[5] N. Fenton and J. M. Bieman, *Software Metrics: A Rigorous and Practical Approach*, 3rd ed. New York, USA: CRC Press, 2014.

[6] T. Gilb and S. Finzi, *Principles of software engineering management*. Addison-wesley Reading, MA, 1988, vol. 11.

[7] R. Van Solingen, V. Basili, G. Caldiera, and H. D. Rombach, "Goal question metric (gqm) approach," *Encyclopedia of software engineering*, 2002.

[8] P. Ofem, B. Isong, and F. Lugayizi, "On the concept of transparency: A systematic literature review," *IEEE Access*, vol. 10, pp. 89 887–89 914, 2022.

[9] C. Cappeli, J. C. S. d. P. Leite, and A. d. P. A. Oliveira, "Exploring business process transparency concepts," in *15th IEEE International Requirements Engineering Conference (RE 2007)*, 2007, Conference Proceedings, pp. 389–390.

[10] J. C. S. d. P. Leite and C. Cappeli, "Exploring i* characteristics that support software transparency," in *CEUR Workshop Proceedings*, Recife, Brazil, February 2008, Conference Proceedings, pp. 51–54.

[11] C. Cappeli, "An approach for business processes transparency using aspects," Thesis, school=Dissertation, Departamento de Informática, PUC-Rio, Ago (in Portuguese), 2009.

[12] M. Hosseini, A. Shahri, K. Phalp, and R. Ali, "Towards engineering transparency as a requirement in socio-technical systems," in *2015 IEEE 23rd International Requirements Engineering Conference (RE)*, Ottawa, Canada, 2015, Conference Proceedings, pp. 268–273.

[13] M. Hosseini, A. Shahri, K. Phalp, and R. Ali, "Foundations for transparency requirements engineering," in *Proceedings of the 22nd International Working Conference on Requirements Engineering: Foundation for Software Quality*, vol. 9619. Gothenburg, Sweden: Springer-Verlag, 2016, Conference Proceedings, pp. 225–231.

[14] M. Hosseini, A. Shahri, K. Phalp, and R. Ali, "A modelling language for transparency requirements in business information systems," in *International Conference on Advanced Information Systems Engineering*. Ljubljana, Slovenia: Springer, 2016, Conference Proceedings, pp. 239–254.

[15] M. Hosseini, A. Shahri, K. Phalp, and R. Ali, "Four reference models for transparency requirements in information systems," *Requirements Engineering*, 2018.

[16] D. Spagnuelo, C. Bartolini, and G. Lenzini, "Metrics for transparency," in *Data Privacy Management and Security Assurance*. Heraklion, Crete, Greece: Springer International Publishing, 2016, Conference Proceedings, pp. 3–18.

[17] D. Spagnuelo, C. Bartolini, and G. Lenzini, "Modelling metrics for transparency in medical systems," in *Trust, Privacy and Security in Digital Business*. Lyon, France: Springer International Publishing, 2017, Conference Proceedings, pp. 81–95.

[18] D. Spagnuelo and G. Lenzini, "Transparent medical data systems," *J. Med. Syst.*, vol. 41, no. 1, pp. 1–12, 2017.

[19] Y.-C. Tu, "Transparency in software engineering," Thesis, University of Auckland, New Zealand, 2014.

[20] Y.-C. Tu, E. Tempero, and C. Thomborson, *Evaluating presentation of requirements documents: Results of an experiment*. Springer, 2014, pp. 120–134.

[21] B. Isong, P. Ofem, and F. Lugayizi, "Towards a framework for improving transparency in the software engineering process," in *2022 12th International Conference on Software Technology and Engineering (ICSTE)*. Osaka, Japan: IEEE, 2022, pp. 19–28.

[22] B. Baudry, C. Nebut, and Y. Le Traon, "Model-driven engineering for requirements analysis," in *11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007)*. IEEE, 2007, pp. 459–459.

[23] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Heidelberg New York Dordrecht London: Springer Science & Business Media, 2012.

[24] IEEE, *IEEE Standard for a Software Quality Metrics Methodology(1061-1998(R2009)*. IEEE, 2009.

[25] C. Fernández-Gago and D. Nuñez, "Metrics for accountability in the cloud," in *Summer School on Accountability and Security in the Cloud*. Springer, 2014, pp. 129–153.

[26] C. Kaner and W. P. Bones, "Software engineering metrics: What do they measure and how do we know?" in *10th International Software Metrics Symposium, METRICS 2004*. IEEE CS Press, 2004, Conference Proceedings, pp. 1–12.

[27] K. Srinivasan and T. Devi, "Software metrics validation methodologies in software engineering," *International Journal of Software Engineering & Applications*, vol. 5, no. 6, p. 87, 2014.

[28] H. Zuse, *Software complexity: measures and methods*. Walter de Gruyter GmbH & Co KG, 2019, vol. 4.

[29] A. Meneely, B. Smith, and L. Williams, "Validating software metrics: A spectrum of philosophies," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 21, no. 4, p. 24, 2012.

Authorized licensed use limited to: North West University. Downloaded on January 08,2026 at 06:52:55 UTC from IEEE Xplore. Restrictions apply.