UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Tecnura

https://revistas.udistrital.edu.co/index.php/Tecnura/issue/view/1327

DOI: https://doi.org/10.14483/22487638.24849

INVESTIGACIÓN

Metrics for the evaluation of documentation format in agile software projects

Métricas para la evaluación del formato de la documentación en proyectos de software ágil

Juan Carlos Narváez [1] y César Jesús Pardo Calvache [2]

Fecha de Recepción: 12 de noviembre de 2025 **Fecha de Aceptación**: 24 de febrero de 2026

Cómo citar: J.C. Narváez, C.J. PArdo Calvacho, «Metrics for evaluation of documentation format in agile software projects», *Tecnura*, vol. 30, n.° 88, jun. 2026. 38–57. https://doi.org/10.14483/22487638.24849

Abstract

Objective: This paper proposes a metrics model for assessing and making documentation debt visible in agile software development, with a specific focus on the Format dimension.
Methodology: Based on a systematic literature review, the study identified critical risks and applied the Goal-Question-Metric (GQM) paradigm to design indicators. Both subjective metrics, based on perception, and objective metrics, based on structural analysis, were defined to evaluate attributes such as readability, conciseness, and clarity.
Results: A five-dimensional architecture was defined (Structure, Format, Usability, Correlation, and Auditability) and specific metrics were developed for the Format dimension, addressing readability, conciseness, and clarity. These metrics integrate subjective user evaluation with objective structural complexity indicators—such as Average Words per Sentence (PPO)—to quantify risks associated with code misunderstanding and communication failures.
Conclusions: Format deficiencies can give rise to serious risks, including communication failures. The combined use of subjective and objective metrics is essential for a holistic diagnosis, transforming documentation quality into a measurable engineering attribute that can be managed proactively
Keywords: Documentation Debt, Metrics, Agile software development, Technical Debt, Software Quality

Resumen

Objetivo: este trabajo propone un modelo de métricas para evaluar y visibilizar la deuda de la documentación en el desarrollo ágil de software, enfocándose específicamente en la dimensión de Formato.
Métodología: a partir de una revisión sistemática de literatura, se identificaron riesgos críticos y se aplicó el paradigma Goal-Question-Metric (GQM) para diseñar indicadores. Se definieron métricas subjetivas (basadas en percepción) y objetivas (análisis estructural) para evaluar atributos como legibilidad, concisión y claridad.

1 Magister en computación. Docente en la Universidad del Cauca. Email: juanarvaez@unicauca.edu.co
2 PhD. en Informática. Docente en la Universidad del Cauca. Email: cpardo@unicauca.edu.co

Resultados: se definió una arquitectura de cinco dimensiones (Estructura, Formato, Usabilidad, Correlación y Auditabilidad) y se desarrollaron métricas específicas para el Formato, abordando la Legibilidad, Concisión y Claridad. Estas métricas integran la evaluación subjetiva del usuario con indicadores objetivos de complejidad estructural, como el Promedio de Palabras por Oración (PPO), para cuantificar riesgos asociados a la incomprensión del código y fallas de comunicación.
Conclusiones: las deficiencias de formato son causantes de riesgos graves como fallas de comunicación. La sinergia entre métricas subjetivas y objetivas es indispensable para un diagnóstico holístico, transformando la calidad documental en un atributo de ingeniería medible para su gestión proactiva.
Palabras clave: Deuda de la Documentación, Métricas, Desarrollo Ágil de Software, Deuda Técnica, Calidad de Software

Introduction

High-quality documentation is widely regarded as a strategic resource in software engineering because it is essential for understanding a system holistically and maintaining it over the long term [1, 2]. However, there is a clear gap between its recognized importance and its practical implementation: producing pragmatic and effective documentation remains a persistent challenge in the field [3]. This problem is exacerbated in the context of agile approaches where, despite their aim of improving value delivery in short development cycles [4], the principle of documenting only what is necessary is often misinterpreted as a justification for relegating software documentation to a secondary role [5, 6, 7].

Far from being a harmless omission, this practice creates an insidious type of technical debt known as documentation debt [8]. This concept does not simply refer to a lack of text, but to an ecosystem of deficient information characterized characterized by scarce, inconsistent, or outdated artifacts. The fundamental problem with this type of technical debt is its invisible nature: it accumulates silently until its consequences become prohibitively expensive [4, 8].

The progressive accumulation of technical debt in software documentation increases the likelihood of risks that compromise project viability, maintenance costs, the efficiency of onboarding new team members, and overall software quality [9]. As a result, the personnel involved in building a software system remain trapped in a cycle of reactive management, facing negative consequences only once they have reached a critical state. This research is therefore motivated by the need for a formal mechanism that allows the level of technical debt in software documentation to be visualized, measured, and controlled.

This paper responds to that need by proposing a structured metrics model designed to quantify the likelihood of risks associated with software documentation. The model seeks to transform an abstract and subjective problem—such as poor documentation—into concrete and actionable indicators. It addresses Documentation Debt from five fundamental dimensions: Structure, Format, Usability, Correlation, and Auditability. This allows organizations not only to identify weaknesses but also to justify the

allocation of resources and proactively manage the quality of their knowledge assets. This article introduces the general architecture of the model and, as an initial result, details the set of metrics designed for the Format dimension [10].

The remainder of the article is organized as follows. Section 2 describes the five dimensions of documentation and their characteristics. Section 3 presents the methodology followed for the construction of the model. Section 4 presents the metrics of the Format dimension. Section 5 discusses the results. Section 6 presents the conclusions and future work.

Dimensions and Characteristics of Documentation

To evaluate documentation debt systematically, a hierarchical model is proposed that organizes documentation quality into two levels: **Dimensions**, which represent the main evaluation perspectives for a documentation artifact, and **Characteristics**, which correspond to the specific and measurable attributes within each dimension.

As shown in Figure 1, the model comprises five dimensions—Structure, Format, Usability, Correlation, and Auditability—which group a total of seventeen quality characteristics, providing a comprehensive basis for diagnosing documentation quality and identifying areas in which debt may accumulate.

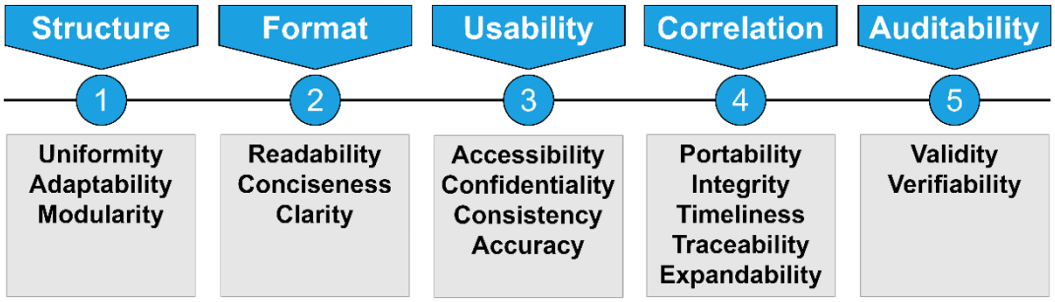

Figure 1. Dimensions and characteristics of software documentation

Source: own elaboration.

The dimensions and characteristics associated with each are described below.

Structure

Following Koznov [11], this dimension focuses on the logical distribution of documentation content to enable effective understanding and use. Each component—indexes, chapters, glossary, images, tables, error descriptions and solutions, usage instructions, and references—must have a specific purpose

and be coherently related, either internally (with parts of the same artifact) or externally (with other artifacts), to maintain the overall meaning of the document. Its characteristics are:

- **Uniformity:** Consistency of the information within the artifact, ensuring that there are no conflicts, contradictions, or duplications, and maintaining coherence in its format, structure, storage, and updating [11, 12].

- **Adaptability:** The artifact's ability to evolve and remain relevant as the needs and requirements of the project change [13].

- **Modularity:** The capacity to divide an artifact into independent, cohesive sections with specific objectives, facilitating the understanding of specific system details without requiring review of the entire document [1].

Format

The Format dimension focuses on the visual presentation, language, and grammar of the content. Although its deficiencies may go unnoticed in the short term thanks to the team's tacit knowledge, they seriously affect the future understanding of the information in the long term [1, 10]. The characteristics of this dimension are:

- **Readability:** The ease of reading an artifact, which can be hindered by abstract, overly technical, or overloaded information, or by the presence of typographical errors. This characteristic evaluates the effort required by the reader to access the information and achieve the purpose for which the document was created [14].

- **Conciseness:** The ability to use only the information strictly necessary to explain the system without sacrificing accuracy. The presence of useless or redundant sections makes the document cumbersome and incomprehensible [1].

- **Clarity:** The ability of content to be understandable and intelligible without generating doubts. It requires appropriate language and simple wording that allows information to be found quickly without the need to re-read, adapting the structure and visual design to the user's needs [1].

Usability

The Usability dimension assesses how easily a reader can obtain the information necessary to meet their immediate objectives. This perspective analyzes the effective use of documentation, considering factors such as navigation, format, structure, and preservation mechanisms that condition access [15]. Its characteristics are:

- **Accessibility:** The ability of the documentation to be located and operational when needed, allowing for its retrieval and reading according to the user's availability and access level [12].

- **Confidentiality:** Based on the ISO 27001 standard, this ensures that documentation is available only to authorized personnel or processes, requiring visible classification of artifacts —such as public, internal, or restricted.

- **Consistency:** The ability of documentation to avoid discrepancies between its content and the actual operation of the system. It depends on the uniformity and integrity of the content and is evaluated through elements such as the level of formality, semantics, visual content, and organization [12, 16, 17].

- **Accuracy:** The ability of the artifact to convey accurate and unambiguous information that faithfully reflects the actual state of the system, given that inaccurate documentation can be more harmful than its absence [15, 18].

Correlation

The Correlation dimension refers to the correspondence required between the software system and its documentation. Its main objective is to ensure consistency between recorded information and the actual behavior of the system, keeping artifacts up to date whenever changes occur. Its characteristics are:

- **Portability:** The quality that allows documentation to be transferred efficiently between different technological or operational environments, using standard formats that ensure independence and interoperability [12].

- **Integrity:** The degree to which the documentation remains correct and complete after changes to the system, retaining sufficient information to support development, maintenance, and use tasks [12].

- **Timeliness:** The degree to which documentation remains current throughout the software lifecycle, depending on adequate traceability to ensure the simultaneous evolution of the system and its documents [12].

- **Traceability:** The record of the artifact's evolution, allowing identification of the context of changes—where, when, who, and why. This facilitates tracking of modifications to maintain consistency with the system and understand dependencies between artifacts [16].

- **Expandability:** The ability to add or modify information in an artifact—increasing its size or scope—without negatively affecting other characteristics, which is essential for system evolution and maintenance [15].

Auditability

The Auditability dimension involves inspecting and verifying documentation with respect to software behavior and established review criteria. Documentation must serve as eliable evidence of the procedures, actions, and decisions taken during development. Its characteristics are:

- **Validity:** The ability to prove the authenticity, veracity, legality, and reliability of information through mechanisms such as signatures and seals, allowing agreements to be formalized and the relevance of the content to be certified [1, 16].

- **Verifiability:** The ability to corroborate whether the documentation provides accurate and reliable information, avoiding conflicts with other artifacts. It relies on readability and structure to ensure effective review, since incorrect or unverified information can lead to errors [1, 16].

Model construction methodology

The construction of the model began by consolidating empirical evidence through systematic literature mapping and causal analysis of documentation debt [19]. This process made it possible to derive and formulate the associated risks, design the overall architecture, and define the metrics under the Goal-Question-Metric (GQM) paradigm.

Risk definition process

To ensure that the model addresses significant Documentation Debt issues in agile software development, a formal risk identification process was carried out in three phases.

Phase 1 (Empirical Evidence Collection) consolidated the findings of a systematic literature mapping on documentation debt in agile software development conducted in 2022 [20], which provided the knowledge base for designing the metrics model.

Phase 2 (Causal Analysis), involved the extraction and analysis of the most frequently reported factors and consequences in the literature. A total of 45 causes and 42 effects of documentation debt were identified, which were then refined to remove redundancies and consolidate the main concepts.

Phase 3 (Synthesis and Risk Formulation), used the refined causes and effects to formulate 15 specific documentation risks. Each risk was formulated following a structure inspired by the ISO 31000 standard, detailing four elements: the affected actor, the nature of the risk, its root cause, and the potential impact.

Table 1 presents the risks defined during this process according to the following format:

For (who is affected by the risk) + **there is a risk** (specification of the risk) + **due to** (one or more causes of the risk) + **this may** (the consequences of the risk materializing)

Table 1. *Documentation risks in agile software development*

Id	Risk description
R1	For development and maintenance teams, there is a risk of misunderstanding the source code in the future, due to a lack of code comments and technical documentation that is scarce, incomprehensible, outdated, or non-existent. This may affect the maintainability and modifiability of the system, teamwork, knowledge transfer, and problem solving.
R2	For administrators and end users, there is a risk that the system will be highly vulnerable because its documentation does not correctly describe its architecture, design, functionalities, and restrictions. This may hinder the understanding of risks and threats, impede the implementation or improvement of security practices, and increase the complexity of incident management and response.
R3	For the company, there is a risk that the system will be very difficult to maintain in the future because its documentation does not support an adequate understanding of its operation, architecture, design, and the decisions made during its construction. This may affect code refactoring, decrease consistency in system development, hinder the traceability of changes, and increase the effort required to identify, understand, and resolve incidents.
R4	For the company, there is a risk that the system will be of poor quality because its documentation does not provide clear and accurate information for understanding its design, architecture, and operation. This may hinder quality control, scalability and maintainability, knowledge transfer to new staff, and increase dependence on the personnel who originally participated in its development. It may also cause dissatisfaction among stakeholders and damage the company's business relationships, reducing competitiveness.
R5	For the company, there is a risk of rework during system development because the documentation is confusing, incomplete, inconsistent, or outdated. This may lead to misunderstandings in the interpretation of requirements, increase the likelihood of errors, and cause important information to be omitted during design or implementation, resulting in rework and delivery delays.
R6	The company faces the risk of progressively accumulating unmanageable levels of technical debt due to unclear documentation, the absence of best practices and standards, a lack of details on dependencies and risks, and non-existent or outdated design and architecture documentation. This may lead to low-quality software resulting from rushed or incorrect decisions.
R7	The company faces the risk of having informal development and quality control processes because the guidelines to be followed at each stage are not clearly documented. This may lead to a lack of uniformity and consistency in the execution, monitoring, and control of processes, and increase the likelihood of losing valuable knowledge, as processes are executed based on intuition rather than documented guidelines.
R8	The company faces the risk of not improving its processes because the documentation does not adequately record how they should be executed to be efficient, consistent, and improve the quality of operations. This may affect the visibility of processes, limit analysis and performance measurement, increase uncertainty, and constrain the capacity for continuous improvement.
R9	For the deployment team, there is a risk that the system cannot be correctly implemented in their environment because the documentation lacks precise information on its operation and how it should be installed, configured, and used. This may affect implementation planning, lead to errors or omissions during the process, and hinder the functioning of the system after deployment
R10	For the company, there is a risk of losing the knowledge acquired by the development team due to poor, incomplete, or non-existent documentation of design decisions, functionalities, configuration, integrations, requirements management, problem solutions, and lessons learned. This may increase dependence on the tacit knowledge of certain team members and reduce team productivity
R11	The development team faces the risk of implementing system requirements incorrectly due to ambiguous documentation. This may lead to requirements being misinterpreted, inconsistent, or contradictory, causing confusion during system construction and increasing the likelihood of omitting important details in functionalities, constraints, business rules, or non-functional requirements.
R12	For the company, there is a risk of not effectively integrating new staff, because the documentation does not support an adequate understanding of the system, its status, its operation, the requirements it must meet, its design, and its architecture. This may delay the adaptation of new personnel and their ability to fully contribute to the team's objectives.

Id	Risk description
R13	For the quality team, there is a risk that it will be very difficult to determine how to test the system because the documentation presents incomplete information on requirements and does not clearly explain the acceptance criteria. This may lead to test cases that do not cover all relevant scenarios, inadequate defect management, and loss of traceability.
R14	The company faces the risk of losing the ability to adequately track the evolution and changes made over time to documentation artifacts due to the deficiency or absence of version control. This may affect version history management, make it difficult to understand the context of modifications, and lead to confusion and loss of information
R15	For the development team, there is a risk of communication problems arising because the documentation is ambiguous, unclear, or outdated in its description of the system, its functionalities, and its conditions of use. This may lead to misinterpretations, misunderstandings, and confusion in internal and external communications, affecting collaboration and problem-solving.

Source: own elaboration.

To contextualize the metrics, risks R1, R4, and R15 were selected due to their high relevance to the Format dimension. These risks address issues such as code comprehension, low system quality, and communication failures, which directly affect the clarity, readability, and accuracy of the documentation. This selection allowed specific indicators to be defined to measure their impact on software viability.

Risk and dimension mapping

Once the risks had been identified, the next step was to establish their relationship with the dimensions to validate the conceptual coverage of the model and understand the multidimensional nature of the risks. Three steps were taken:

First, **a critical analysis and semantic deconstruction of each risk** was performed to extract its components: the affected actors (those who suffer the consequences), the nature of the risk (the central negative event), the root causes (identifying adjectives that denote documentary deficiencies such as "ambiguous" or "scarce"), and the potential impact. The objective was to obtain a set of concrete descriptors that directly relate each risk to specific failures or deficiencies in the documentation artifacts.

Second, **the link between each risk and the documentation quality characteristics was established** by comparing the previously extracted descriptors with the definitions in the model. The fundamental criterion was the existence of a direct correspondence between the problem described in the risk (cause or impact) and the specific quality attribute.

Third, **the affected characteristics were grouped into the corresponding dimensions** of the model. The criterion established was to associate a risk with a dimension if it affected at least one of its characteristics, producing a multidimensional impact profile for each risk. For example, risk R1 was associated with four dimensions, while R3 covered all five.

Table 2 presents the resulting mapping matrix, which summarizes the interrelationship between risks and the dimensions of the model.

Table 2. *Mapping of risks to documentation with the dimensions of the model.*

Risk	Structure	Format	Usability	Correlation	Auditability
R1	X	X	X	X	
R2		X	X		X
R3	X	X	X	X	X
R4		X	X		X
R5		X	X	X	X
R6	X	X	X	X	
R7	X	X	X	X	X
R8		X	X		
R9			X	X	X
R10	X		X	X	X
R11	X	X	X		X
R12		X	X		
R13	X	X	X	X	X
R14	X		X	X	X
R15	X	X	X		
Total	9	12	15	9	10

Source: own elaboration.

To illustrate the mapping process described above, the analysis performed for risk R1 is detailed below:

Definition of risk R1 (see Table 1): For development and maintenance teams, there is a risk of misunderstanding the system's source code in the future, because it has no comments to support the explanation of its intent and the technical documentation is scarce, incomprehensible, outdated, or non-existent. This may affect the maintainability and modifiability of the system, teamwork, knowledge transfer to new staff, and problem solving.

In this first step, the **causes** were identified (scarce, incomprehensible, outdated, or nonexistent documentation), the **central problem** was defined (the inability to correctly understand the source code), and the **impact** was determined (negative effects on system maintainability and knowledge transfer).

In the second step, direct links were established between the textual components of R1 and the quality characteristics. Phrases such as "failure to understand correctly" or "incomprehensible" indicate difficulty assimilating information, related to **Clarity** and **Readability**. Words such as scarce or non-existent correspond to a deficiency in **Integrity**, while outdated indicates failures in **Timeliness** and **Accuracy**.

Incomprehensibility was associated with a lack of Uniformity, and difficulties in maintainability or knowledge transfer were identified as **Modularity** problems.

In the third step, the identified characteristics were grouped with their respective dimensions. Uniformity and Modularity belong to the Structure dimension; Clarity and Readability correspond to Format; Accuracy is located in Usability; and Integrity and Timeliness are part of Correlation.

As a result, risk R1 would impact on the Structure, Format, Usability, and Correlation dimensions, as shown in Table 2.

The analysis of risk distribution (Table 2) was decisive in defining the scope of the research. Although the Usability dimension emerged as the most cross-cutting—impacted by all 15 risks—the initial study was strategically focused on the Format dimension, based on several considerations.

The Format dimension is critical, as it is the second most affected, directly linked to 12 of the 15 identified risks. Risks such as source code misunderstanding (R1), low system quality (R4), and critical communication failures (R15) are strongly associated with the presentation and language of the documentation. This suggests that improving Readability, Conciseness, and Clarity can substantially help mitigate serious project risks.

Beyond its frequency of impact, the Format dimension is foundational to documentation quality: if documentation lacks Readability or Clarity, it fails to fulfill its communicative purpose, regardless of its visual structure. Therefore, addressing format deficiencies is an indispensable step toward enabling and enhancing quality across the rest of the model.

Furthermore, the Format dimension allows for direct identification of problems and the implementation of concrete corrective measures. Problems in this area are tangible and observable, manifesting in specific elements such as excessively long sentences or the absence of visual aids. This advantage is reflected in the metrics developed (Section 4), which integrate objective indicators derived from linguistic and structural analysis (IFKE, IEC, IPP, PPO, PFO, ICP) with subjective assessments of user perception (PLD, PCD, PCLD). This combination provides a robust diagnosis that not only detects whether a problem exists but also identifies its structural causes, facilitating the implementation of specific and measurable improvements.

In conclusion, given the criticality, foundational role, and diagnostic tangibility of the Format dimension, this article focuses on the specification and discussion of its metrics. This approach seeks to establish a coherent methodology and provide concrete results that serve as a starting point for future research on the remaining dimensions.

General architecture of the model and use of GQM

Once the risks were established, the relational architecture of the model was designed (Figure 3), incorporating the dimensions and characteristics described above. Under the Goal-Question-Metric (GQM) paradigm [21] (Figure 2), each characteristic is associated with a group of specific metrics: metrics for **direct measurement** of objective attributes and metrics for **subjective perception** through surveys.

The GQM paradigm ensures that metrics are not arbitrary, but rather aligned with measurement objectives and the questions that evaluate their achievement. It operates at three levels:

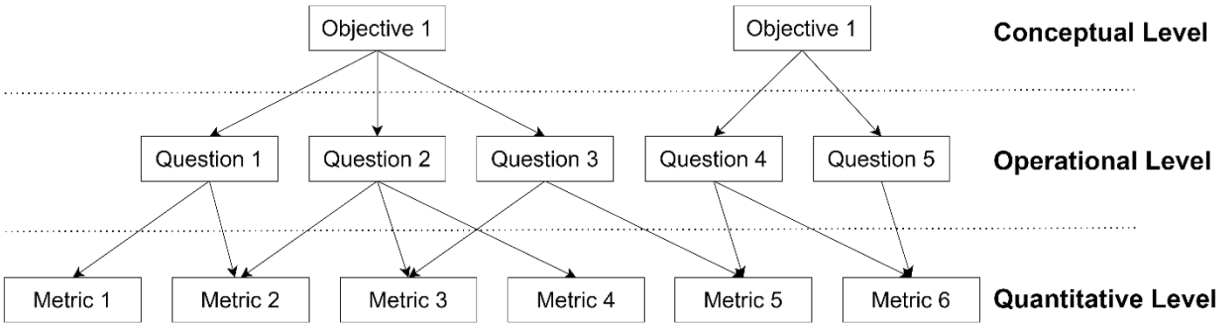

Figure 2. Basic structure of the GQM paradigm

Source: Basili et al. [21].

Conceptual Level (Goal): Defines clear and strategically aligned measurement objectives, specifying their purpose, perspective, object of analysis, and context.

Operational Level (Questions): Breaks down each objective into specific questions that characterize the object of study and suggests the information needed to determine whether the goal is being achieved.

Quantitative Level (Metric): Defines specific metrics to answer these questions. They can be objective —such as lines of code— or subjective —level of satisfaction— and must be clearly linked to the operational questions.

The five dimensions that make up the architecture are: (i) Structure, which focuses on the logical organization and distribution of content; (ii) Format, which evaluates the visual presentation, language, and grammar used; (iii) Usability, which measures the ease with which the reader can extract the information necessary for their objectives; (iv) Correlation, which examines the consistency and timeliness of the documentation with respect to the system it describes; and (v) Auditability, which determines the artifact's ability to be systematically inspected and verified as a reliable record.

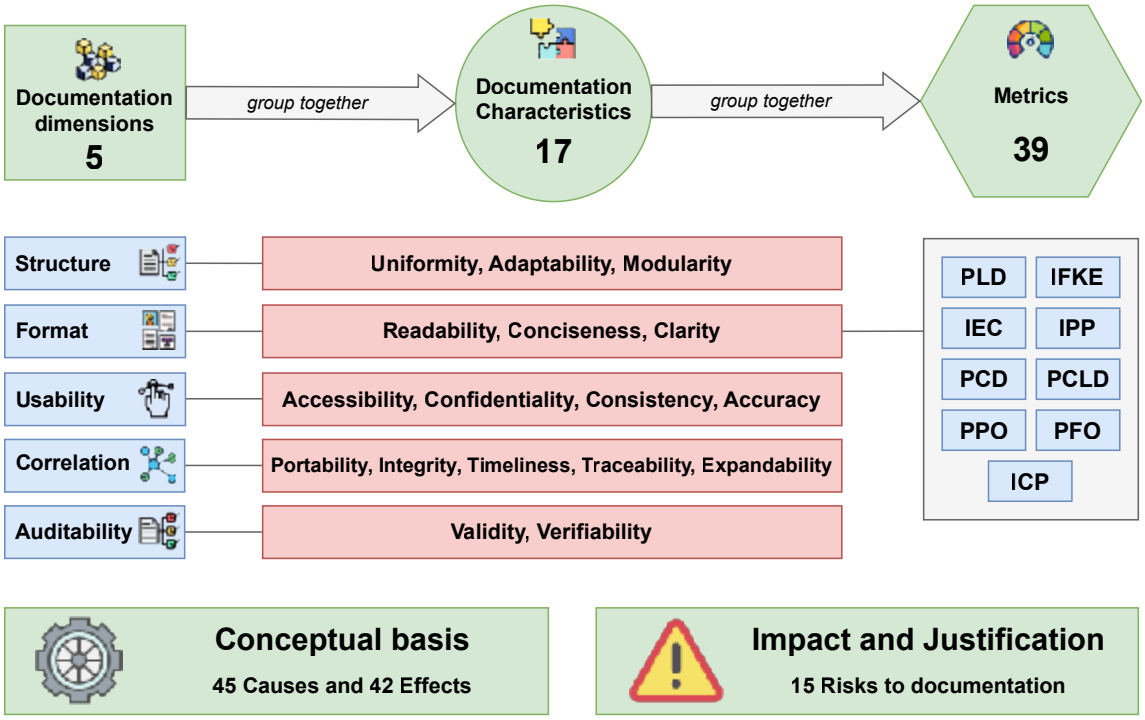

Figura 3. General architecture of the proposed metrics model

Source: own elaboration.

Metrics for the Format dimension

As defined in Section 2, the Format dimension encompasses the visual presentation and language of the artifact. Because deficiencies in this dimension can hinder comprehension and weaken long-term knowledge preservation—regardless of the technical accuracy of the content—this section details the specific metrics for its three key characteristics: Readability, Conciseness, and Clarity.

Readability Metrics

Readability is a subjective quality that determines the ease of reading. Since factors such as excessive technicality or typographical errors impair it, its evaluation requires a mixed approach: combining user perception with objective metrics that analyze the linguistic structure of the text.

The first metric (Equation 1) is the **Perceived Readability of Documentation (PLD)**, which quantifies users' subjective assessment of reading ease. It is measured on a scale of 0 to 100—where higher values indicate better quality—and is calculated using a four-question survey with ratings from 1 to 5, available at https://bit.ly/3ZZwvSz. For example, if a participant's responses yield a result of 62.5%, readability is classified as acceptable according to Table 3, indicating that the text is adequate but has shortcomings that require additional reader effort.

$$PLD = \left(\frac{(P21 + P22 + P23 + P24) - 4}{16}\right) * 100 \qquad (1)$$

Table 3. *Interpretation scale for the PLD metric*

Result	Classification	Interpretation
PLD ≥ 80	Excellent	Readability is satisfactory. The text is very easy to read and review.
60 ≤ PLD < 80	Good	Readabilityis good, although there are minor flaws that do not impede comprehension.
40 ≤ PLD < 60	Acceptable	Readability is fair, and significant shortcomings are identified that require effort to overcome.
10 ≤ PLD < 40	Poor	Readability is poor and there are critical shortcomings that seriously hinder reading and comprehension.
PLD < 10	Very poor	Readability is non-existent or very poor, and the text is practically incomprehensible or requires extreme effort to understand.

Source: own elaboration.

The second metric (Equation 2) is the **Flesch-Kincaid Index for the Spanish language (IFKE)** [22], which measures the linguistic complexity of the text to determine its ease of comprehension by an average reader. The result is expressed on a scale of 0 to 100, where a higher score indicates easier reading. For example, a fragment of a user story consisting of 72 words, 2 sentences, and 158 syllables would yield a result of 38.45, classifying the text as "Difficult" according to Table 4.

$$IFKE = 206.84 - 1.02\left(\frac{NTP}{NTO}\right) - 60\left(\frac{NTS}{NTP}\right) \qquad (2)$$

Table 4. *Interpretation scale for the IFKE metric*

Result	Classification	Interpretation
90 ≤ IFKE < 100	Very easy	The text is optimal and accessible to any reader without effort.
80 ≤ IFKE < 90	Easy	The text is simple and the risk of misunderstanding is very low.
70 ≤ IFKE < 80	Moderately easy	The text is clear and can be read without difficulty by most people.
60 ≤ IFKE < 70	Normal	The text is functional but requires additional effort for readers unfamiliar with its information.
50 ≤ IFKE < 60	Moderately difficult	The text is slightly complex, requires effort, and there is a risk of misinterpretation.
30 ≤ IFKE < 50	Difficult	The text is complex with a high risk of misunderstanding and requires considerable effort to understand its content.
0 ≤ IFKE < 30	Very difficult	The text is very dense or too technical.

Source: R. Flesch [22].

The third metric (Equation 3) is the **Content Structuring Index (IEC)**, based on the premise that objective readability depends not only on linguistic complexity (IFKE) but also on visual presentation. Its purpose is to measure structural richness by evaluating the density of formatting elements—headings, lists, and visual aids—that organize content and break up blocks of text. A higher value indicates better

structural readability. For example, an artifact with 20 paragraphs and 10 supporting elements would yield an IEC of 0.5, while the same content with only 1 heading would result in an IEC of 0.05, indicating dense text with very low structural readability. Table 5 provides the interpretation scale.

$$IEC = \frac{Nh+Nl+Nv}{Np} \qquad (3)$$

Table 5. *Interpretation scale for IEC*

Result	Classification	Interpretation
IEC = 0	Very Poor	No formatting elements exist. Structural readability is zero.
0 < IEC < 0.25	Poor	There are very few structural elements. The text remains dense and difficult to review.
0.25 ≤ IEC < 0.5	Acceptable	There are some structural elements. The text requires review.
0.5 ≤ IEC < 1.0	Good	There is at least 1 formatting element for every 2 paragraphs. The content is segmented and easy to review visually.
IEC ≥ 1	Excellent	There are one or more formatting elements (headings, lists, visuals) for every paragraph, with an excellent balance between content and visual structure.

Source: own elaboration.

The fourth metric (Equation 4) is the Paragraph Penalty Index (IPP), which quantifies excessively long paragraphs that increase cognitive load and hinder quick information retrieval. The metric compares the number of lines in each paragraph (LPi) against a predefined optimal threshold (LOP, e.g., 8 lines). A quadratic penalty is applied to lines exceeding this threshold, based on the premise that the negative impact on readability is non-linear: a very long paragraph is disproportionately more harmful than several medium-length ones. The result is the average of these penalties; a lower value indicates better quality (Table 6). For example, a document with three paragraphs (Np = 3) of 6, 10, and 14 lines, with an optimal threshold of 8, would generate an IPP of 13.33, classifying the format as Poor.

$$IPP = \frac{1}{Np}\sum_{i=1}^{Np}(\max(0, LPi - LOP))^2 \qquad (4)$$

Table 6. *Interpretation scale for IPP*

Result	Classification	Interpretation
IPP = 0	Optimal	No difficulties; all paragraphs are within the optimal threshold.
0 < IPP < 5	Good	Paragraph density is slight; paragraphs are moderately long, but their impact is minimal.
5 ≤ IPP < 20	Poor	The text is dense, with a notable presence of long paragraphs that hinder reading.
IPP ≥ 20	Very Poor	The text is very dense and contains extremely long paragraphs.

Source: own elaboration.

Conciseness metrics

Conciseness involves providing the necessary information without sacrificing accuracy; when absent, documentation becomes redundant, cumbersome, and difficult to understand. To measure it, the **Perception of Documentation Conciseness (PCD)** metric (Equation 5) is proposed, which determines whether users consider the content sufficiently detailed without including unnecessary information. It is expressed on a scale of 0 to 100 and is calculated from a six-question survey with ratings from 1 to 5 (https://bit.ly/46r9DxA). For example, if a user's responses yield a result of 75%, the perception of conciseness is classified as "Good" according to Table 7.

$$PCD = \left(\frac{(P25+P26+P27+P28+P29+P30)-6}{24}\right) * 100 \qquad (5)$$

Table 7. *Interpretation scale for PCD*

Result	Classification	Interpretation
PCD < 10	Very poor	There is excessive information overload, the content is redundant, and it lacks semantic efficiency.
10 ≤ PCD < 40	Poor	There is a high presence of superfluous information, making it difficult to extract relevant content.
40 ≤ PCD < 60	Acceptable	The documentation contains non-essential information that requires additional effort on the part of the reader.
60 ≤ PCD < 80	Good	The information is concise, with only minor redundancies that do not impede understanding.
PCD ≥ 80	Excellent	The information is accurate and efficient, presenting only the strictly necessary content.

Source: own elaboration.

Clarity metrics

Clarity is the ability of content to be intelligible without generating doubts or requiring reinterpretation. Given that its absence erodes trust, its evaluation requires a mixed approach that integrates the user's subjective perception with an objective analysis of the textual structure.

The first metric (Equation 6), **Perceived Clarity of Documentation (PCLD)**, assesses whether users perceive the content as understandable and intelligible. Results are expressed on a scale of 0 to 100, derived from a four-question survey with ratings from 1 to 5. For example, if a user's responses yield a result of 68.75%, clarity is classified as "Good" according to Table 8.

$$PCLD = \left(\frac{(P31 + P32 + P33 + P34)-4}{16}\right) * 100 \qquad (6)$$

Table 8. *Interpretation scale for PCLD*

Result	Classification	Interpretation
PCLD < 10	Very poor	The text is perceived as ambiguous and confusing, generating a high degree of uncertainty in the reader.
10 ≤ PCLD < 40	Poor	The content is perceived as difficult to understand and requires frequent reinterpretation, generating doubts about its meaning.
40 ≤ PCLD < 60	Acceptable	The text is moderately understandable but lacks the efficiency necessary to convey the message without generating uncertainty.
60 ≤ PCLD < 80	Good	The documentation is perceived as clear and intelligible, with only a few minor ambiguities that do not impede comprehension.
PCLD ≥ 80	Excellent	The content is perceived as fully understandable and intelligible, conveying the message efficiently and without raising doubts.

Source: own elaboration.

To complement the subjective evaluation, three objective metrics are proposed that analyze the structural complexity of the text.

The first is the **Average Words per Sentence (PPO)** metric (Equation 7), which quantifies excessively long sentences that increase cognitive load and hinder comprehension. It is expressed as a positive integer value; a lower number indicates greater clarity. For example, a PPO of 60 is classified as "Poor" (Table 9) because it requires significant memory effort to process.

$$PPO = \frac{TPC}{TOC} \qquad (7)$$

Table 9. *Interpretation scale for PPO*

Result	Classification	Interpretation
PPO < 15	Excellent	The text uses short, direct sentences. It is optimally clear.
10 ≤ PPO < 40	Good	The text uses sentences of standard length. It is easy to read for most people.
40 ≤ PPO < 60	Acceptable	The text uses moderately long sentences and requires more effort from the reader.
60 ≤ PPO < 80	Poor	The text uses long, complex sentences that increase cognitive load and the risk of ambiguity.
PPO ≥ 80	Very poor	The text uses excessively long sentences with a high risk of misunderstanding and reader fatigue.

Source: own elaboration.

The **Average Phrases per Sentence (PFO)** metric (Equation 8) assesses syntactic complexity by measuring the density of ideas per sentence. A low value is desirable, as an excess of clauses makes reading difficult. For example, if a text of 2 sentences (TOC) contains 7 clauses (TFC), PFO = 3.5, indicating poor clarity (Table 10).

$$PFO = \frac{TFC}{TOC} \qquad (8)$$

Table 10. *Interpretation scale for PFO*

Result	Classification	Interpretation
PPO = 1.0	Excellent	The syntactic structure is ideal; each sentence conveys a single idea, and the cognitive load is minimal.
1.0 < PFO ≤ 2.0	Good	Simple sentences that combine few ideas, are easy to process, and have a low risk of debt.
2.0 < PFO ≤ 3.0	Acceptable	Sentences are moderately dense, containing multiple ideas. Reading effort and the risk of debt due to ambiguity increase.
3.0 < PFO ≤ 4.0	Poor	Sentences are complex, syntactically nested, and there is a high density of ideas that increase cognitive load.
PFO > 4.0	Very poor	Sentences are excessively dense or chained, and the cognitive load is very high.

Source: Authors.

Finally, the **Prose Quality Index (ICP)** (Equation 9) measures the formal quality of the text by quantifying the presence of errors that generate uncertainty and affect clarity. Unlike PPO and PFO, which evaluate structure, ICP addresses correctness. The metric is normalized per 1,000 words and applies severity weighting, under the premise that grammatical errors are more detrimental to meaning than spelling errors. A lower value indicates higher prose quality.

$$ICP = \frac{(Eo*Wo)+(Eg*Wg)}{Pt/_{1000}} \qquad (9)$$

For example, a document with 2000 words (Pt) that has ten spelling errors (Eo) and four grammatical errors (Eg), using Wo=1 and Wg=3, the IPC would be 11, indicating acceptable prose (Table 11).

Table 11. *Interpretation scale for ICP*

Result	Classification	Interpretation
0 ≤ ICP < 2	Excellent	The prose is polished and virtually error-free.
2 ≤ ICP < 5	Good	There are some minor or slight errors that do not impede comprehension.
5 ≤ ICP < 15	Acceptable	There are noticeable errors that require revision and may compromise the clarity of the artifact.
ICP ≥ 15	Poor	The prose is sloppy and contains a high density of errors that negatively impact clarity and credibility.

Source: own elaboration.

Discussion

The main strength of the proposal lies in the combination of the team's subjective perception with the objective analysis of the artifacts. Perception metrics (such as PLD and PCLD) are crucial in agile environments to validate the practical value of documentation, because a technically sound document that is perceived as confusing is effectively unusable.

Objective metrics (such as IFKE and PPO) provide an empirical basis for diagnosing the root cause of poor perception. For example, a low subjective rating for clarity can be explained by a specific datum—an average sentence length that is too long (high PPO)—thereby transforming an abstract complaint into an actionable problem.

Beyond quality assessment, these metrics act as risk management tools. A deterioration in Format scores serves as an early warning that critical risks—such as source code incomprehensibility (R1), low system quality (R4), and communication failures (R15)—may materialize, enabling the team to move from reactive to proactive management.

The proposal also has limitations: the Format dimension is critical but insufficient on its own. A holistic diagnosis of documentation debt requires future evaluation of the remaining four dimensions: Structure, Usability, Correlation, and Auditability.

Conclusions and future work

This work addresses documentation debt in agile development as an engineering variable that has historically lacked formal measurement mechanisms because of its invisible nature. Its main contribution is a metrics model that transforms this abstract concept into measurable and actionable indicators.

The analysis of the Format dimension reveals that much project risk does not stem from a lack of information, but from how that information is presented. Objective metrics such as Average Words per Sentence (PPO) suggest that structural problems are often the root cause of low perceived clarity, increasing the likelihood of communication failures.

An exclusively objective or subjective approach is insufficient. True diagnostic capacity emerges from the combination of the two: while subjective perception (e.g., PCLD) identifies the existence of a problem, objective metrics (e.g., IFKE, PPO) diagnose its structural cause. This work therefore represents an initial step toward managing documentation quality not as an art, but as an engineering attribute that must be measured for proactive control.

Three lines of future work are proposed. First, the specification of metrics for the remaining four dimensions (Structure, Usability, Correlation, and Auditability) must be finalized to complete the model. Second, the model must be empirically validated in real industrial environments to assess its accuracy and practical usefulness. Third, the development of a software tool based on emerging technologies—such as Generative Artificial Intelligence (GenAI)—is planned to automate metric execution, facilitating adoption in industry and improving documentation processes throughout the software development lifecycle.

Referencias

[1] E. Aghajani *et al.*, "Software documentation issues unveiled," in *Proc. IEEE/ACM Int. Conf. Softw. Eng. (ICSE)*, Montreal, QC, Canada, 2019, pp. 1199–1210. https://doi.org/10.1109/ICSE.2019.00122

[2] G. Matturro, F. Raschetti, and C. Fontán, "A systematic mapping study on soft skills in software engineering," *J. Univers. Comput. Sci.*, vol. 25, no. 1, pp. 16–41, 2019. https://doi.org/10.3217/jucs-025-01-0016

[3] N. Rios, R. O. Spínola, M. Mendonça, and C. Seaman, "The practitioners' point of view on the concept of technical debt and its causes and consequences: A design for a global family of industrial surveys and its first results from Brazil," *Empir. Softw. Eng.*, vol. 25, no. 5, pp. 3216–3287, 2020. https://doi.org/10.1007/s10664-020-09832-9

[4] S. Al-Saqqa, S. Sawalha, and H. Abdelnabi, "Agile software development: Methodologies and trends," *Int. J. Interact. Mobile Technol.*, vol. 14, no. 11, pp. 246–270, 2020. https://doi.org/10.3991/ijim.v14i11.13269

[5] M. Fowler, "Technical debt quadrant," Oct. 2009. Accessed: Oct. 17, 2022. [Online]. Available: https://martinfowler.com/bliki/TechnicalDebtQuadrant.html

[6] A. Martini, T. Besker, and J. Bosch, "Process debt: A first exploration," in *Proc. Asia-Pacific Softw. Eng. Conf. (APSEC)*, Singapore, 2020, pp. 316–325. https://doi.org/10.1109/APSEC51365.2020.00040

[7] J. P. Zumba and C. A. L. Arreaga, "Evolución de las metodologías y modelos utilizados en el desarrollo de software," *INNOVA Res. J.*, vol. 3, no. 10, pp. 20–33, Oct. 2018. https://doi.org/10.33890/innova.v3.n10.2018.65

[8] N. Rios *et al.*, "Hearing the voice of software practitioners on causes, effects, and practices to deal with documentation debt," in *Lecture Notes in Computer Science*, vol. 12130. Cham, Switzerland: Springer, 2020, pp. 55–70. https://doi.org/10.1007/978-3-030-44429-7_4

[9] Y. Shmerlin, I. Hadar, D. Kliger, and H. Makabee, "To document or not to document? An exploratory study on developers' motivation to document code," in *Lecture Notes in Business Information Processing*, vol. 215. Cham, Switzerland: Springer, 2015, pp. 100–106. https://doi.org/10.1007/978-3-319-19243-7_10

[10] A. A. H. Alzahrani, "Software systems documentation: A systematic review," *Int. J. Adv. Comput. Sci. Appl.*, vol. 15, no. 8, 2024. https://doi.org/10.14569/IJACSA.2024.0150816

[11] D. V. Koznov, D. V. Luciv, and G. A. Chernishev, "Duplicate management in software documentation maintenance," *CEUR Workshop Proc.*, vol. 1989, pp. 195–201, 2017.

[12] J. Zhi, V. Garousi-Yusifoglu, B. Sun, G. Garousi, S. Shahnewaz, and G. Ruhe, "Cost, benefits and quality of software development documentation: A systematic mapping," *J. Syst. Softw.*, vol. 99, pp. 175–198, 2015. https://doi.org/10.1016/j.jss.2014.09.042

[13] J. Bayer and D. Muthig, "A view-based approach for improving software documentation practices," in *Proc. Int. Symp. Workshop Eng. Comput. Based Syst. (ECBS)*, Potsdam, Germany, 2006, pp. 269–278. https://doi.org/10.1109/ECBS.2006.18

[14] M. Zanoni, F. Perin, F. A. Fontana, and G. Viscusi, "Pattern detection for conceptual schema recovery in data-intensive systems," *J. Softw.: Evol. Process*, vol. 26, no. 12, pp. 1172–1192, 2014. https://doi.org/10.1002/smr.1656

[15] J. D. Arthur and K. T. Stevens, "Assessing the adequacy of documentation through document quality indicators," in *Proc. Conf. Softw. Maintenance (ICSM)*, Miami, FL, USA, 1989, pp. 40–49. https://doi.org/10.1109/icsm.1989.65192

[16] R. Plosch, A. Dautovic, and M. Saft, "The value of software documentation quality," in *Proc. Int. Conf. Quality Softw. (QSIC)*, Dallas, TX, USA, 2014, pp. 333–342. https://doi.org/10.1109/QSIC.2014.22

[17] M. Kajko-Mattsson, "A survey of documentation practice within corrective maintenance," *Empir. Softw. Eng.*, vol. 10, no. 1, pp. 31–55, 2004. https://doi.org/10.1023/B:LIDA.0000048322.42751.ca

[18] A. Forward and T. C. Lethbridge, "The relevance of software documentation, tools and technologies: A survey," in *Proc. 2002 ACM Symp. Document Eng.*, McLean, VA, USA: ACM, 2002, pp. 26–33. https://doi.org/10.1145/585058.585065

[19] N. Qamar, N. Sabahat, and A. Mosavi, "Evaluating the impact of pair documentation on requirements quality in agile software development," *IEEE Access*, vol. 13, pp. 45784–45794, 2025. https://doi.org/10.1109/ACCESS.2025.3550123

[20] J.-C. Narváez-Narváez, C.-J. Pardo-Calvache, and C.-E. Orozco-Garcés, "Deuda de la documentación en el desarrollo ágil de software: mapeo sistemático de la literatura," *Rev. Cient.*, vol. 46, no. 1, pp. 107–121, Jan. 2023. https://doi.org/10.14483/23448350.19670

[21] V. R. Basili and G. Caldiera, "The goal question metric paradigm," in *Encyclopedia of Software Engineering*, J. J. Marciniak, Ed. New York, NY, USA: Wiley, 1994, vol. 2, pp. 528–532. [Online]. Available: https://go.umd.edu/3ol7AtO

[22] R. Flesch, "A new readability yardstick," *J. Appl. Psychol.*, vol. 32, no. 3, pp. 221–233, 1948. https://doi.org/10.1037/h0057532