Journal of Software Engineering Research and Development, 2026, 14:6, doi: 10.5753/jserd.2026.5953

TechDebt Tracker: Towards a Method for Monitoring Technical Debt in Agile Projects

Mayra Pereira [**Universidade Federal da Paraíba** | ***mayra.daher@academico.ufpb.br***]
Adriana Damasceno [**Universidade Federal da Paraíba** | ***adriana.damasceno@academico.ufpb.br***]

Abstract

Context: Software companies that adopt agile methods face numerous challenges in sustaining the long-term evolution of software systems. Technical debt is a key contributor to poor maintainability, often leading to failures in agile software projects. This situation becomes even more problematic when managers do not adequately address technical debt items. *Objective:* This paper proposes the TechDebt Tracker, a method for supporting the documentation and monitoring of technical debt items within technical debt management activities in the context of agile projects. *Method:* We employed Design Science Research to develop and evaluate the proposal, following the summarized steps of related work review, problem definition, design and development, demonstration, and evaluation. In the related work review, we examined studies with similar research questions across multiple digital libraries. During the design and development phase, we used Design Thinking, the Business Model Canvas, and the Value Proposition Canvas to identify vulnerabilities and opportunities for improvement in the emerging solution. The proposal was demonstrated in a small software company, from which feedback was gathered to refine the method. The evaluation phase consisted of a small-scale study assessed through questionnaires. *Results:* The proposal comprises three components: two formulas, a kanban board, and a flow. The formulas are used to measure the impact of technical debt on the project and incorporate data such as the developer's hourly rate, severity, and penalty associated with each technical debt item. The kanban board includes several columns—such as monitoring, technical debt backlog, and testing—as well as a card template used to register and prioritize each technical debt item. The flow consists of states and actions that, when used together with the kanban board, define a method for monitoring technical debt. The company evaluated the proposal positively, highlighting its technical adequacy. *Conclusions:* We recommend evaluating the proposal in additional contexts and hope that this proposal will be adopted by software companies that employ agile methodologies in diverse scenarios. Our goal is to take a step toward developing a method that can be used in the daily operations of companies to simplify technical debt management.

Keywords: *technical debt, agile methods, project management*

1 Introduction

Low test coverage, poorly defined requirements, and outdated documentation are examples of practices that violate software engineering principles (Gomes et al., 2023). Making these decisions can lead to consequences that compromise software quality and hinder its maintenance and growth (Freire et al., 2024). To measure the effects of such poor practices, the term technical debt is used.

The technical debt metaphor was introduced by Ward Cunningham in 1992 (Ciolkowski et al., 2021) to facilitate communication between the technical team and other organizational stakeholders. Several definitions of technical debt are available in the literature. According to Seaman and Guo (Seaman and Guo, 2011), this concept refers to the impact of any incomplete, immature, or inadequate artifact within the software development life cycle. In 2016, during the Dagstuhl Seminar 16162 (Stochel et al., 2020), technical debt was formally defined by Avgeriou et al (Avgeriou et al., 2016) as a design or implementation construct that is beneficial or convenient in the short term but may impose substantial costs or hinder long-term evolution. More recently, a broader and more stable understanding has emerged, recognizing that technical debt may also include documentation, process, and business-related decisions (Wiese and Borowa, 2023). This work adopts the definition of technical debt from Avgeriou et al (Avgeriou et al., 2016). Even so, despite advances toward consensus, some practitioners still use the term incorrectly, vaguely, or as a catch-all label for any problem encountered in a project (Junior and Travassos, 2022) (Rios et al., 2018).

There are different classifications of technical debt, such as unavoidable technical debt, which arises when software design does not evolve well, or naive technical debt, which stems from the immaturity of the business or the development team. The accumulation of technical debt can have serious consequences for a company, including product atrophy, frustration among clients and developers, increased development costs due to rework, and, in extreme cases, system failure (Gomes et al., 2023).

The various stages of software development—such as requirements, architecture, implementation, and testing—can generate different types of technical debt (Ciolkowski et al., 2021). In projects using agile methods, it is uncommon to adopt approaches for managing technical debt due to a lack of documentation and a focus on visual methods (Wang et al., 2022). Additionally, the processes used to identify, prioritize, and monitor technical debt are often complex or not well-suited to agile projects (Freire et al., 2024).

Several studies have addressed technical debt in the context of agile teams; however, none have proposed a visual approach to its management and prioritization. For example, Seaman and Guo (Seaman and Guo, 2011) propose a technical debt management framework composed of three

stages: identification, measurement, and monitoring. However, this framework does not consider agile methods. Other approaches focus on identifying, measuring, or supporting decision-making regarding the repayment of technical debt items (Santos et al., 2016) (Chicote, 2017) (Pacheco et al., 2018). However, few studies have explored the relationship between agile methods and technical debt (Gomes et al., 2024) (Freire et al., 2024) (Gomes et al., 2023) (Avgeriou et al., 2020).

This work aims to answer the following research question: "Can a visual, action-flow-based approach be proposed as support for documentation and monitoring activities within technical debt management?". To address this question, we propose the TechDebt Tracker method to assist in the documentation and monitoring of technical debt items. Its main feature is its simplicity, designed to facilitate adoption among companies using agile methods. The proposal comprises a Kanban board, a flow, and formulas for measuring technical debt items recorded on the board. It was applied in a small company and evaluated through a small-scale study, which yielded positive feedback on the proposed approach.

2 Background

Among the software engineering practices that could be more effectively leveraged in software development, technical debt management stands out as a critical concern. To understand the challenges surrounding this topic, several studies have been conducted over the years. According to Seaman and Guo (Seaman and Guo, 2011), the technical debt metaphor has been broadened to encompass any imperfect artifact within the software development life cycle. Giardino et al. (Giardino et al., 2015) argue that software companies often face a trade-off between the pressure to deliver rapidly and the need to maintain quality across different phases of development, a tension that frequently results in the accumulation of technical debt. Other relevant concepts associated with technical debt include the cost of repayment and the interest incurred as a result of accepting the debt.

Technical debt can yield short-term advantages or lead to detrimental long-term consequences. Strategic decisions at both technical and business levels can positively influence a software solution and even enhance its competitive advantage. For example, a team may choose to shorten a product's delivery timeline to capitalize on a market opportunity. To do so, it might adopt technologies that increase productivity while intentionally reducing test coverage. This enables an earlier release and the acquisition of new customers. According to Junior and Travassos (Junior and Travassos, 2022), such benefits are typically short-lived, closely aligned with organizational business goals, and may be classified as project management–related, business-driven, or team-oriented.

However, decisions that generate immediate gains may evolve into long-term risks if not properly monitored and mitigated. As reported by Rios et al (Rios et al., 2020), issues such as degraded product performance, customer dissatisfaction, increased refactoring demands, low-quality code, and diminished developer morale may arise in projects that fail to understand, manage, or that accumulate technical debt. These consequences can be severe, including increased maintenance effort, degradation of software quality, and even product stagnation. Furthermore, as noted by Junior and Travassos (Junior and Travassos, 2022), negative impacts tend to manifest over extended periods and are directly associated with the internal quality of the software.

Given that the accumulation of technical debt can lead to significant challenges—such as increased costs, reduced performance, and customer dissatisfaction—its primary management objective is to mitigate such negative outcomes. However, as emphasized by Rubin (Rubin, 2018), not all debt should be repaid immediately. Instead, debts with higher interest should be prioritized and addressed incrementally while maintaining the delivery of customer value.

To support technical debt management, several approaches have been proposed over the years, including portfolio-based strategies and techniques for estimating technical debt interest. Ampatzoglou et al. (Ampatzoglou et al., 2015) highlight ongoing contradictions in the literature regarding interest management and present a framework based on contemporary economic theories as a means of addressing these challenges. Silva et al (Silva et al., 2019) identify eight activities associated with technical debt management: identification, measurement, prioritization, prevention, monitoring, repayment, documentation, and communication. Throughout these activities, it is necessary to adopt an appropriate prioritization method. Among those available in the literature, the work of De Almeida et al (De Almeida et al., 2021) stands out, as they present a business-oriented approach to prioritizing technical debt. These approaches share a common set of foundational activities that support the effective management of accumulated technical debt.

these advances, managing technical debt continues to present several challenges. Foremost among these is the lack of a consolidated and unified understanding of technical debt, its characteristics, and its impacts among software engineers. According to Jeronimo Junior and Travassos (Junior and Travassos, 2022), the Dagstuhl Seminar represented a significant step toward conceptual consolidation. Nevertheless, existing research still exhibits important gaps: studies often focus on a limited subset of technical debt types, fail to fully capture their impacts, and do not convincingly demonstrate the effectiveness of current tools and methods for technical debt management. Another challenge concerns improving communication and the dissemination of technical debt–related knowledge. It is essential that technical debt be addressed not only within development teams but also in academic environments, particularly in software engineering and software quality courses.

Some authors, such as Caires et al. (Caires et al., 2019), have shown that the use of agile methods can yield both positive and negative effects on technical debt. One major concern is the level of interactivity, which has been identified as a key factor in contributing to adverse outcomes.

Agile methods encompass development practices that adhere to the Agile Principles for Software Development. The Agile Manifesto, created by a group of software practitioners seeking to propose alternatives to traditional engineering processes, presents four core values (Fowler et al., 2001):

1. Individuals and interactions over processes and tools;
2. Working software over comprehensive documentation;
3. Customer collaboration over contract negotiation; and
4. Responding to change over following a fixed plan.

3 Related Work

As far as is known, there are no studies that enable the management of technical debt using agile methods in a visual approach. Several initiatives addressing this issue have been identified, some aiming to characterize the use of technical debt for different audiences and others focusing on managing technical debt in projects that employ agile methods.

Among the studies characterizing the use of technical debt, we can cite Seaman and Guo (Seaman and Guo, 2011), Gomes *et al.* (Gomes et al., 2024), Freire *et al.* (Freire et al., 2024), and Gomes *et al.* (Gomes et al., 2023).

Seaman and Guo (Seaman and Guo, 2011) proposed a framework that uses the relationship between cost and benefit to prioritize technical debt items to be addressed. In their approach, metrics are initially estimated as low, medium, or high and subsequently quantified numerically, enabling a decision on whether to correct each item. These initial estimates are based on historical data or expert opinion. The TechDebt Tracker draws inspiration from this comparison between principal and interest and adopts the formula by Seaman and Guo (Seaman and Guo, 2011) for calculating interest. However, it incorporates the approach by Curtis et al. (Curtis et al., 2012) for measuring the principal. Additionally, it adopts a Kanban board for integration with agile methods and a flow to assist in monitoring each technical debt item.

Junior and Travassos (Junior and Travassos, 2023) proposed the *isTDM*, a solution developed from an evidence-based conceptual framework to support Technical Debt management. The framework comprises three technologies: a set of guidelines (TDM Guidelines), a Glossary of Terms (TDM Glossary), and a Technology Portfolio (Portfolio of TDM Technologies). These three instruments can be used together or independently and are supported by a web platform. It presents itself as a complete and comprehensive solution to support both the academic community and the software industry. On the other hand, TechDebt Tracker positions itself as a visual and simplified tool to support the monitoring of technical debt items.

Gomes *et al.* (Gomes et al., 2024) conducted a multi-method study to evaluate ninety-seven tools designed to identify self-admitted technical debt. The majority of these tools provide access to upper management rather than focusing on repaying technical debt. In most cases, technical debt is identified through comments in the project code, leading to results with low precision.

Freire *et al.* (Freire et al., 2024) present a study on how software practitioners monitor technical debt items. The study investigates the practices used for monitoring technical debt items and the main reasons for not doing so. To this end, the authors analyzed 653 responses from software engineers distributed across six countries. They identified 46 monitoring practices for technical debt items and 35 reasons for not monitoring them. Among the findings, we observed that technical debt is most frequently tracked in the backlog, using specialized tools, or discussed in team meetings. The main reasons for not monitoring technical debt include a lack of interest or time and a focus on short-term goals. However, no visual techniques for monitoring and managing technical debt were presented.

Gomes *et al.* (Gomes et al., 2023) analyzed technical debt from the perspective of project managers. To achieve this, they examined over one hundred discussions on the StackExchange website. The study found that the perspective on technical debt differs between project managers and software developers. Technical debt indicators were identified and analyzed using a Sankey diagram. However, no visual management approach for technical debt was proposed.

Regarding projects that employ agile methods and deal with technical debt, we can cite Oliveira (Oliveira, 2015), Santos (Santos et al., 2016), Chicote (Chicote, 2017), Pacheco *et al.* (Pacheco et al., 2018), and Freire *et al.* (Freire et al., 2023). Oliveira (Oliveira, 2015) aimed to evaluate the application of the technical debt management framework proposed by Seaman and Guo (Seaman and Guo, 2011). For this purpose, an action research study was conducted in two software projects using Scrum. The study proved relevant as it presented a real-world experience of implementing the framework, identified success factors and areas for improvement, and recommended adaptations for agile software projects.

Santos (Santos et al., 2016) proposed an extension to Scrum to support technical debt management and conducted an evaluation of this approach. The proposal was initially developed based on evidence from the literature (secondary methods) as well as insights from industry professionals (field research). However, no specific method for managing and prioritizing technical debt was presented.

Chicote (Chicote, 2017) introduced a technical debt management technique based on using graphical elements to organize information and ideas, aiming to increase visibility into the process and support technical debt management in early-stage startups employing lean processes. The proposed technique consists of using adhesive tape cut to a size proportional to the corresponding technical debt. However, the study did not present a method for prioritizing technical debt items.

Pacheco *et al.* (Pacheco et al., 2018) proposed the design and implementation of technical debt visualizations to facilitate communication among stakeholders and support decision-making. The study focused on visualization techniques such as graphs, trees, and Sankey diagrams. However, it did not address methods for prioritizing technical debt.

Freire *et al.* (Freire et al., 2023) evaluated the use of IDEA diagrams as support for technical debt management activities. IDEA diagrams are used to analyze practices applicable to the context of technical debt. When analyzing these diagrams from the perspective of software professionals in software development projects, most participants came from medium and large companies. However, adoption by companies using agile methods remains limited, and there is still a lack of guidance on how IDEA diagrams can be integrated into agile methods such as Scrum. Furthermore, the study did not define criteria for prioritization, nor did it specify which

practices should be represented in the diagrams, given the broad scope of technical debt and its associated types and practices.

The studies mentioned above and the present research share the goal of simplifying technical debt management by providing tools that can be adapted to the realities of software development teams. The works of Oliveira (Oliveira, 2015) and Santos (Santos et al., 2016) provided the foundation for the technical debt backlog mechanism adopted in our study. Likewise, the studies by Chicote and Pacheco, together with the present research, aim to enhance understanding, improve communication among stakeholders, and facilitate the overall management of technical debt.

4 Methods

This is a qualitative research study aimed at understanding the subjective aspects of the use of technical debt in companies that adopt agile software methods. It was developed using the Design Science Research method (Wieringa, 2014). This research method is used to build knowledge within the context of a scientific approach to design. According to Dresch *et al.* (Dresch et al., 2020), its main principles are relevance and rigor, ensuring the reliability of the results. Additionally, Design Science Research aims to develop artifacts, prescribe solutions, and study artificial constructs. Its main output is a man-made artifact. The stages proposed by Peffers *et al.* (Peffers et al., 2007) were adopted for this study, namely: (1) Problem identification and motivation, (2) Definition of objectives for a solution, (3) Design and development, (4) Demonstration, (5) Evaluation, and (6) Communication.

In stage 1 (problem identification and motivation), a state-of-the-art literature review was conducted using databases such as SBC OpenLib[1], IEEE Xplorer[2], TD Researcher Team[3], and ACM Digital Library[4] with the search keywords: "technical debt", "dívida técnica", "agile processes", and "processos ágeis". This review identified that the adoption of agile methods may be linked to negative effects on technical debt (Pompermaier, 2021). We concluded that the most commonly used agile methods in small software projects are Scrum, Kanban, and Lean, as pointed out by Mendes (Mendes, 2020).

In stage 2 (definition of objectives for a solution), Wieringa's (Wieringa, 2014) design problem template was used. This led to the following general objective: "How can rework costs be minimized by proposing a method to help project managers using agile methods document and monitor technical debt items?". The main challenges include the lack of software engineering training among team members, insufficient computational resources and environments, and inadequate business decisions that contribute to increasing the project's technical debt. We have expanded the scope of this work to encompass all types of technical debt. This is feasible because our focus is on the documentation and monitoring of technical debt items, and these activities are largely consistent across technical debt categories (Junior and Travassos, 2022) (Rios et al., 2020) (Alves et al., 2016). Once a technical debt item is identified and added to the technical debt backlog, it can be used as input for our proposal.

During the design and development phase (stage 3), the TechDebt Tracker was proposed. To build it, the Business Model Canvas and the Value Proposition Canvas were used to identify and validate strategic guidelines that would add value to the method's development. Then, Design Thinking (Ruschel, 2019) was applied in the creative idea-generation process. The goal was to understand project managers' needs and challenges and leverage their previous experiences to contribute to software development. This process followed four steps (Ruschel, 2019): preliminary immersion, where similar items were analyzed; deep immersion, with the definition of personas and a value map of stakeholders; synthesis and analysis of information (definition of insights and priorities); ideation and prototyping (using the Trello software[5]).

In the first stage, called preliminary immersion, an analysis of similar tools was conducted to identify their main strengths and limitations in alignment with the objectives of this study. The analyzed tools were TD Manager (RIBEIRO, 2016), Technical Debt Plugin (Vieira, 2014), TD Classifier (Tsoukalas et al., 2022), and TDxTeam Monitor (dos Santos et al., 2019). The second stage, in-depth immersion, was divided into two sub-stages: persona definition and stakeholder value mapping. Initially, two personas were defined, representing the development team and the business team, respectively. This definition allowed for a deeper understanding of the users and their specific needs. Subsequently, a stakeholder value map was created – including developers, the business team, and the startup – to identify and prioritize their needs and integrate them strategically into the final product development.

Based on the insights gained from the previous stages, we defined a set of requirements. They included: the use of a Kanban board, the ability to measure technical debt per item, continuous monitoring of these items, the use of cards for characterization and tracking, and the adoption of simplified metrics, such as interest and principal. Finally, the ideation stage was conducted using the brainstorming technique, leading to the development of the flow. This flow was later reflected in the Kanban board columns within Trello, serving as the outcome of the prototyping stage.

The demonstration phase (stage 4) resulted in three remote meetings via Google Meet[6] to present the proposal to a technology startup and collect feedback. The company adopts agile methods in its projects and specializes in secure videoconferencing, developing corporate solutions based on synchronous communication technologies. Founded in 2019 as a research project in a federal university in Brazil, the company had 29 employees at the time of this evaluation. Among its products are a video platform, a scheduling management system, and an online education platform. This company was chosen due to its adoption of agile methods and its approach to technical debt management. Additionally, it showed inter-

[1] https://sol.sbc.org.br/index.php/indice
[2] https://ieeexplore.ieee.org/Xplore/home.jsp
[3] https://www.tdresearchteam.com/
[4] https://dl.acm.org/
[5] https://trello.com/
[6] https://meet.google.com

est and availability to use the TechDebt Tracker.

The evaluation phase (Stage 5) was conducted through an exploratory small-scale evaluation (Wohlin and Rainer, 2022), which represents a simplified form of a case study. The evaluation was carried out by the company's development team over three months.

Stage 6 (communication) involved presenting the study at the university where this research was performed, preparing an activity report, and writing this article.

5 TechDebt Tracker

The design and development phase of the Design Science Research resulted in the TechDebt Tracker. The proposal was primarily based on the cost-benefit analysis approach proposed by Seaman and Guo (Seaman and Guo, 2011) and relies on three components: (1) a Kanban board, (2) formulas for measuring and prioritizing technical debt repayment, and (3) a technical debt management flow. Figure 1 provides an overview of the TechDebt Tracker method.

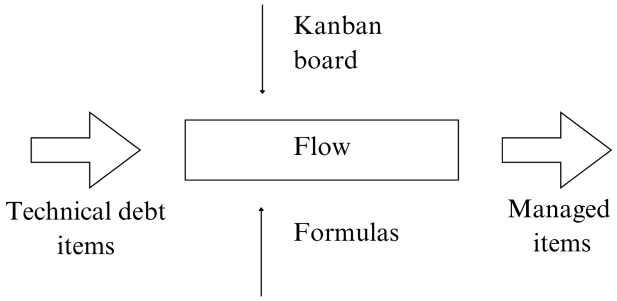

Figure 1. The TechDebt Tracker Method

5.1 The Kanban Board

The Kanban board (component 1) was adapted from the work of Santos (Santos et al., 2016). In this adaptation, we assume that the project is managed using agile methods. This artifact aims to facilitate adaptation and adjustments to existing team processes, as it can be used as a separate board solely for technical debt management or integrated with the sprint backlog.

To evaluate its use, a prototype was developed using the Trello tool[7]. The board consists of the following columns:

- Documents: where the user can access information about the artifacts used in our method;
- Technical Debt Backlog: where new technical debt items are recorded;
- Fixing (Sprint Backlog): containing the items selected for resolution during the sprint;
- Testing: where resolved items are tested;
- Monitoring: gathering items that need continuous tracking;
- Done: registering items that have been fully addressed in the project.

When this board is used alongside the project backlog, the items in the Fixing column can be incorporated into the sprint backlog. Figure 2 presents the Kanban board model, and its implementation is available at the link https://trello.com/b/T0TbIXdL/technical-debt-management.

5.2 Formulas

We use the formulas to manage technical debt and the estimates of: (1) principal and (2) interest. These formulas are based on the works of Curtis *et al.* (Curtis et al., 2012) and Seaman and Guo (Seaman and Guo, 2011) to prioritize technical debt items.

The principal represents the amount of effort required to repay a technical debt item at the current date and is calculated for each item. It is a simple multiplication of the percentage of technical debt to be fixed in each item, the hours required to correct the item, and the cost per hour. Definition 5.1 presents the formula for calculating the principal. This formula is not used when historical data on the effort required to correct a similar item is available; alternatively, an expert estimate may also be used to calculate the principal value.

Definition 5.1 (Principal) *Let i be the corresponding technical debt item in the sprint backlog, $percentage_i$ the percentage of this item to be fixed, $hours_i$ the hours required to correct this item, and $cost_i$ the cost per hour for fixing the item. We define the principal cost $principal_i$ as:*

$$principal_i = percentage_i \times hours_i \times cost\,per\,hour_i$$

The interest quantifies the potential consequences of a technical debt item and is used to prioritize items in the backlog. It incorporates the interest probability, severity, hours, and cost per hour. The interest probability refers to the likelihood of the technical debt item affecting and causing significant damage to the project in the future, with the following predefined values (Seaman and Guo, 2011): 0.8 for high probability, 0.5 for medium probability, and 0.2 for low probability. The severity of the interest measures the impact magnitude of the technical debt item, also ranging from 0.2 to 0.8, from low to high severity. The formula also includes the hours required to fix the item and the cost per hour, as defined in Definition 5.1. Definition 5.2 presents the interest formula.

Definition 5.2 (Interest) *Let i be the corresponding technical debt item in the Kanban board backlog, $interest_i$ its estimated interest cost for debt repayment, $probability_i \in \{0.1, 0.5, 0.8\}$ representing the interest probability, $severity_i \in \{0.1, 0.5, 0.8\}$ representing the penalty severity, $hours_i$ the hours required to fix the item, and $cost_i$ the cost per hour for fixing the item. The interest $interest_i$ is defined as:*

$$interest_i = probability_i \times severity_i \times hours_i \times cost\,per\,hour_i$$

The formulas are available in the "Documents" column of the Kanban board.

[7] https://trello.com/

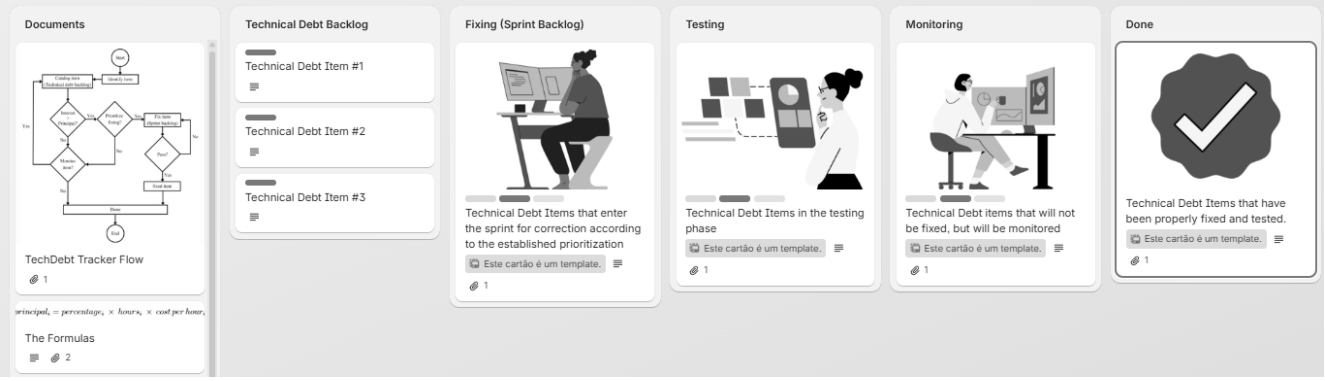

Figure 2. Kanban board

5.3 TechDebt Tracker **Flow**

The management flow defines the steps that are followed for each technical debt item in the project. We developed it to ease decision-making regarding technical debt management, improve communication among the team and stakeholders, and support the standardization of activities related to technical debt. We show its graphical representation in Figure 3.

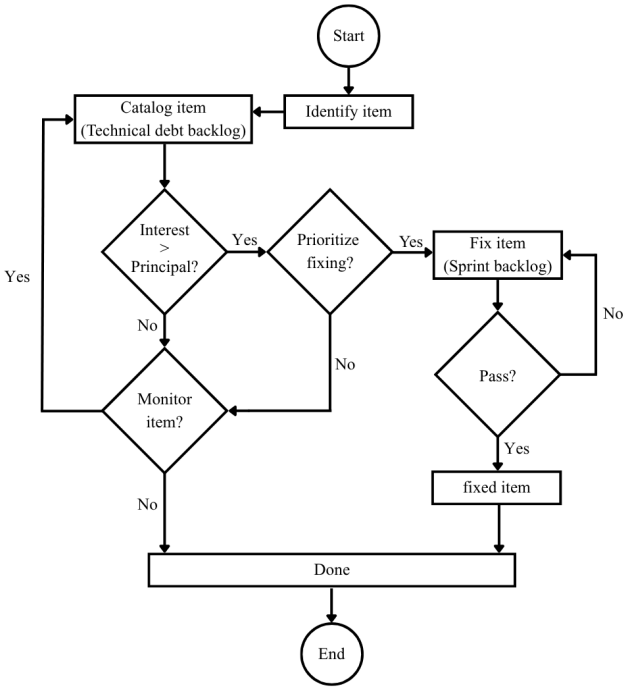

Figure 3. TechDebt Tracker Flow

Revisiting Figure 3, we observe that the flow is organized into activities and decision points, presented in the following sequence. It begins with the activity "identify items". This activity may include code reviews, automated static code analysis tools, best-practices checklists, manual detection through feedback from development or quality assurance teams, documentation review, requirements artifact analysis, and user feedback.

Next, the "catalog item" activity uses the Kanban board and creates a card in the "technical debt backlog" column. The card follows a model adapted from Oliveira's work (Oliveira, 2015) and aims to provide details of a new technical debt item to ease decision-making. It also helps identify the most common types of technical debt generated by the team and developers who frequently deliver immature artifacts, allowing for the development of prevention or mitigation strategies. The card is divided into three sections: characterization, measurement, and prioritization of technical debt. Figure 4 presents this card model.

In the characterization section, the following fields are provided: id (identifier), identification date, responsible person, type of technical debt (design, requirements, architecture, testing, etc.), tracking (location in the code), description, and intentionality (whether the technical debt was deliberate or conscious).

The Technical Debt Measurement section contains the principal and interest fields, calculated using the formulas from Definitions 5.1 and 5.2. For illustration purposes, suppose a technical debt item where the percentage to be corrected is 60%, the hours spent on correction are 16, and the cost per hour is 20. The principal for this item is 192. Now, assuming the probability and severity values are both 0.8, the hours spent to address the debt is still 16, and the cost per hour remains 20, the interest is 204.8.

In the Prioritization section, the decision-making field compares the absolute values of principal and the interest. The prioritization of debt repayment is also visually managed using labels representing high, medium, and low categories.

If the interest is higher than the principal, as in the previous example, this indicates that the technical debt item should be repaid. In this case, the manager proceeds to the new decision point "Prioritize fixing?", which allows them to determine how to handle self-admitted technical debt items. Next, the "fix item" activity is executed. Otherwise, there is no indication to correct the item, and the manager decides whether to monitor it based on their expertise. If the item is not monitored, the card moves to the "Done'' column, and the flow concludes. If the item is monitored, the corresponding card is moved from the "Fixing'' column to the "Monitoring'' column, and the manager determines when to re-evaluate the item. Upon re-evaluation, the card returns to the "catalog item'' activity.

The "fix item" activity may involve refactoring the code, removing dependencies among backlog items, updating requirements documentation, or increasing test coverage. At the start of this activity, the card is moved from the "Technical Debt backlog" column to the "fixing (sprint backlog)" column. From there, it is prioritized within the sprint back-

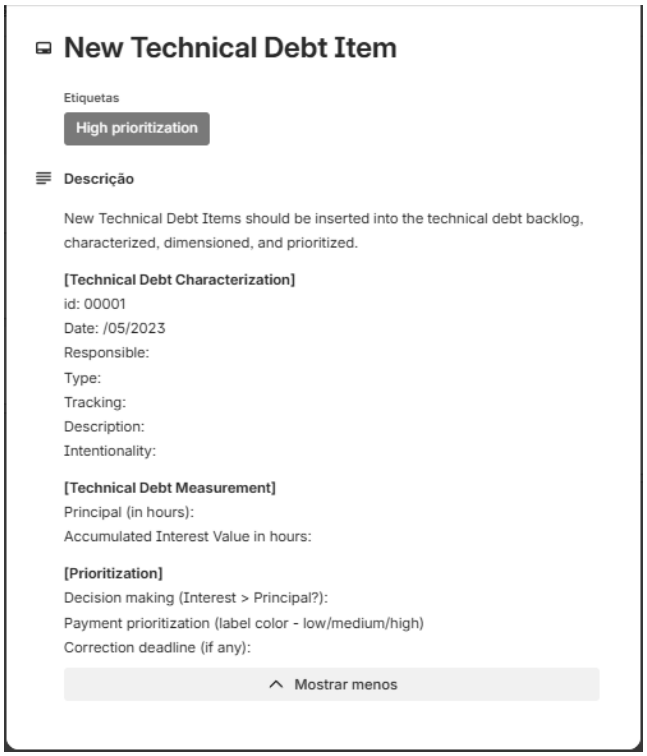

Figure 4. Card model for item registration

log according to its assigned labels (high, medium, or low prioritization). If two cards share the same label, the interest value calculated for the technical debt item is used as a tiebreaker. Once the necessary corrections have been implemented, the card is transferred to the "testing" column, where the manager decides whether to approve or reject the correction. If the correction is not approved, the item returns to the "fix item" activity. If the correction is approved, the item proceeds to the "fixed item" activity. Finally, the card is moved to the "Done" column, concluding the flow.

The "monitor item" activity ensures that the item remains in the technical debt backlog. It differs from the "identify item" activity because the interest estimates have already been calculated. Additionally, monitoring is conducted based on the project manager's needs.

In summary, in the flow, when the team identifies and registers a new technical debt item, it is measured and added to a technical debt backlog. Then, a decision-making process takes place, determining whether the technical debt should be repaid based on whether the interest is greater than the principal. If the answer is yes, the technical debt is repaid, meaning that the problems arising from the previously unaddressed activity must be resolved. Next, the item is evaluated through tests, and if approved, it moves to paid status. Additionally, the manager decides whether to monitor the item. If so, the card is placed in the 'monitoring' column. The TechDebt Tracker Flow is available in the "Documents" column of the Kanban board.

6 Exploratory Small-scale Evaluation

For the evaluation stage of the proposal, we conducted a small-scale assessment involving a development team from a software company. We used this method to analyze the value, usefulness, and innovative character of the newly introduced artifact. The evaluation consisted of three phases. In the first phase, a questionnaire was administered and completed by the project's technical lead before the usage of the proposed approach. In the second phase, the development team used the TechDebit Tracker, identified technical debt items, calculated their costs, prioritized them, and monitored them. In the third and final phase, a second questionnaire was answered by the company's technical lead. Both questionnaires used the Likert scale (Dawes, 2008), with responses ranging from "strongly agree" to "strongly disagree". The goal of applying these questionnaires was to compare the level of knowledge and practical application of technical debt before and after using the TechDebt Tracker based on the respondent's perception and allowing the measurement of benefits and assessment of the proposal's impact.

1. On a scale from 1 to 5, how much do you agree with the following statements?
   (a) I have extensive knowledge of Technical Debt and its impact on software quality;
   (b) I successfully apply approaches for measuring the cost of Technical Debt using metrics;
   (c) I successfully apply approaches for monitoring Technical Debt;
   (d) I successfully apply approaches for prioritizing the repayment of Technical Debt;
   (e) I successfully calculate the cost (hours/year) of rework required to correct Technical Debt items;
   (f) I successfully calculate the cost ($/year) of rework required to correct Technical Debt items.
   (g) I successfully apply approaches for identifying Technical Debt;

Figure 5. Questionnaire applied before the adoption of the proposed method

The initial questionnaire contained seven statements to measure participants' initial knowledge of technical debt concepts. Figure 5 presents the questionnaire items.

The questionnaire applied after executing the proposal comprised ten questions capturing participants' perceptions of the acquired benefits and the techniques adopted by the company, such as prioritization, monitoring, and stakeholder communication within the project. Figure 6 presents the questionnaire items.

6.1 Discussions

We consider that there are indications that TechDebt Tracker adheres to the agile principles described in Section 2. First, the proposal can be used in association with the main agile methods in the labor market, such as Scrum, Lean, and Kanban. This is because its primary component is the Kanban board, which is widely used in agile methods. As a result, interactions among team members are prioritized over the flow and tools used, as decisions regarding technical debt management fall to the manager in collaboration with other team members. At the end of each iteration, complete ver-

1. On a scale from 1 to 5, how much do you agree with the following statements regarding the use of the TechDebit Tracker method?
   (a) Applying the method increased my knowledge of Technical Debt;
   (b) Applying the method increased the use of approaches for measuring Technical Debt;
   (c) Applying the method increased the use of approaches for monitoring Technical Debt;
   (d) Applying the method increased the use of approaches for prioritizing repayment;
   (e) Applying the method increased the calculation of the cost (hours/year) of rework required to correct Technical Debt;
   (f) Applying the method increased the calculation of the cost ($/year) of rework required to correct Technical Debt;
   (g) The method is relevant for assisting managers in Technical Debt management;
   (h) The method's proposal is easy to understand;
   (i) The method is easy to execute;
   (j) Applying the method helped communication among stakeholders in the project when making decisions about Technical Debt.

Figure 6. Questionnaire applied after the method adoption

Table 1. TechDebt Tracker **Evaluation: Pre- and Post-Use Comparative**

Item	Evaluated Aspect	Initial Score	Final Score
(a)	Knowledge	3	5
(b)	Measuring Cost	4	4
(c)	Monitoring	2	5
(d)	Prioritization of Repayment	3	5
(e)	Calculation of Rework Cost (Hours/Year)	2	5
(f)	Calculating Rework Cost ($/Year)	2	5

Note: The scores range from 1 to 5

Table 2. TechDebt Tracker **Evaluation: Post-Use Assessment**

Item	Evaluated Aspect (Post-Use)	Score
(g)	Relevance of the Method	5
(h)	Simplicity of Understanding the Approach	5
(i)	Ease of Execution	4
(j)	Ability to Assist in Communication between Stakeholders	5

Note: The scores range from 1 to 5

sions of technical debt items are delivered, identified, and updated according to the project's progress.

To enhance the presentation of the empirical study, we included Table 1, which provides a comparative analysis of the questionnaire responses collected before and after the application of the TechDebt Tracker method. The observed results reflect the users' experience and contribute to assessing perceived value as well as measuring the practical impact achieved through the use of the proposed approach.

Table 1 assesses the impact and effectiveness of the proposal by comparing users' responses before and after applying the method in their projects, specifically evaluating items (a–f) from the questionnaire in Figure 5 and their corresponding items in Figure 6. The table comprises the columns 'Item,' 'Evaluated Aspect,' 'Initial Score,' and 'Final Score'. The results obtained in the evaluation phase (Step 5 of the Design Science Research) reflect user experience and help to evaluate perceived value and measure the impact and the practical change achieved by using the proposal. They suggest positive impacts on technical debt management. The responses ranged from 1 to 5, where 1 indicated strong disagreement and 5 indicated strong agreement. Regarding knowledge of technical debt, the rating increased from 3 to 5. On the question of measuring the cost of technical debt, the rating remained at 4. When asked about monitoring technical debt, the rating increased from 2 to 5. Regarding prioritization, the rating rose from 3 to 5. Concerning the calculation of the cost (hours/year) of rework needed to fix items, the rating increased from 2 to 5. Similarly, for the question about calculating the cost ($/year) of rework, the initial rating was 2, increasing to 5.

Table 2 was also included to present the method evaluation results. These results did not use comparisons because they belong solely to the questionnaire administered after the application of the TechDebt Tracker method. These items helped to measure different dimensions of the quality of the proposal. The relevance of the method was rated a 5. The simplicity of understanding the approach was also rated a 5, as was its ability to assist in communication between stakeholders. However, its ease of execution received a rating of 4.

Despite the critical limitation in generalizing the results due to the sample size and the possibility of limitations in construct validity, the questionnaire results suggest that the proposal had a positive impact on technical debt management. Using the TechDebt Tracker improved team knowledge about technical debt, enhanced technical debt monitoring, prioritized repayment, and facilitated cost estimation for corrections. Additionally, the proposal was considered relevant for technical debt management. It was also rated as easy to understand and execute, as well as beneficial for supporting stakeholder communication. Moreover, the interviewee suggested an improvement: incorporating aspects of the product roadmap into the decision-making process.

Two threats to the validity of this study were identified, both related to the study participants and their impact on the generalization of the results. The first concerns participant selection, as only one individual was chosen for the small-scale evaluation, which may be considered a low number. Therefore, the results cannot be generalized but remain relevant. The second threat relates to the participant's level of education—a Ph.D. candidate in Computer Science. This limits the generalization of the results to populations with different educational backgrounds, though it does not inval-

idate them. In both cases, the results positively indicate the usefulness of the small-scale evaluation used to validate the study.

7 Concluding Remarks

This study presented a step toward a method for assisting in the documentation and monitoring of technical debt items to be used by agile teams. The proposal consists of a flow, a Kanban board model, and formulas for measuring technical debt items. These components were adapted for practical use in agile software development companies.

The proposal was evaluated through a small-scale study conducted within a software company. Based on the evaluation results, there is evidence that the proposal had a positive impact on technical debt management, improved communication among project stakeholders, and facilitated the dissemination of knowledge about technical debt within the team.

These findings have important implications for advancing the understanding and dissemination of technical debt practices, particularly in agile environments, where few studies have thoroughly examined the interplay between agility and technical debt. The limitations of this research include the small scale of the evaluation and the limited number of participants.

For future work, we suggest applying the proposal in additional companies and testing it with teams of different maturity levels and sizes, or with teams that utilize Continuous Integration and Continuous Delivery (CI/CD) practices, to deepen the understanding of both the limitations and the strengths of our method. We also propose exploring alternative formulas to support technical debt calculation and decision-making, as well as assessing the effectiveness of different practices for companies adopting agile methods. Additionally, analyzing the similarity among technical debt items in the proposed method could further minimize rework when updating records on the Kanban board.

References

Alves, N. S., Mendes, T. S., De Mendonça, M. G., Spínola, R. O., Shull, F., and Seaman, C. (2016). Identification and management of technical debt: A systematic mapping study. *Information and Software Technology*, 70:100–121.

Ampatzoglou, A., Ampatzoglou, A., Avgeriou, P., and Chatzigeorgiou, A. (2015). Establishing a framework for managing interest in technical debt. In *5th International Symposium on Business Modeling and Software Design, BMSD*.

Avgeriou, P., Kruchten, P., Ozkaya, I., and Seaman, C. (2016). Managing technical debt in software engineering (dagstuhl seminar 16162). *Dagstuhl reports*, 6(4):110–138.

Avgeriou, P. C., Taibi, D., Ampatzoglou, A., Fontana, F. A., Besker, T., Chatzigeorgiou, A., Lenarduzzi, V., Martini, A., Moschou, A., Pigazzini, I., et al. (2020). An overview and comparison of technical debt measurement tools. *Ieee software*, 38(3):61–71.

Caires, V., Rios, N., Holvitie, J., Leppänen, V., Licorish, S. A., MacDonell, S. G., Buchan, J., Mendonça, M. G., and Spinola, R. O. (2019). Processos e práticas ágeis sensíveis à dívida técnica-comparação dos resultados de um survey executado no brasil, finlândia e nova zelândia.

Chicote, M. (2017). Startups and technical debt: managing technical debt with visual thinking. In *2017 IEEE/ACM 1st International Workshop on Software Engineering for Startups (SoftStart)*, pages 10–11. IEEE.

Ciolkowski, M., Lenarduzzi, V., and Martini, A. (2021). 10 years of technical debt research and practice: Past, present, and future. *IEEE Software*, 38(6):24–29.

Curtis, B., Sappidi, J., and Szynkarski, A. (2012). Estimating the principal of an application's technical debt. *IEEE software*, 29(6):34–42.

Dawes, J. (2008). Do data characteristics change according to the number of scale points used? an experiment using 5-point, 7-point and 10-point scales. *International journal of market research*, 50(1):61–104.

De Almeida, R. R., do Nascimento Ribeiro, R., Treude, C., and Kulesza, U. (2021). Business-driven technical debt prioritization: An industrial case study. In *2021 IEEE/ACM International Conference on Technical Debt (TechDebt)*, pages 74–83. IEEE.

dos Santos, S. L., Rios, N., Mendonça, M. G., and Spínola, R. O. (2019). Monitoramento da contribuição de equipes de desenvolvimento na evolução de itens da dívida técnica em projetos de software.

Dresch, A., Lacerda, D. P., and Junior, J. A. V. A. (2020). *Design science research: método de pesquisa para avanço da ciência e tecnologia*. Bookman Editora.

Fowler, M., Highsmith, J., et al. (2001). The agile manifesto. *Software development*, 9(8):28–35.

Freire, S., Rios, N., Pérez, B., Castellanos, C., Correal, D., Ramač, R., Mandić, V., Taušan, N., López, G., Pacheco, A., et al. (2024). Hearing the voice of software practitioners on technical debt monitoring: Understanding monitoring practices and the practices' avoidance reasons. *Journal of Software Engineering Research & Development*, 12(1).

Freire, S., Rocha, V., Mendonça, M., Izurieta, C., Seaman, C., and Spínola, R. (2023). Assessing idea diagrams for supporting analysis of capabilities and issues in technical debt management. In *International Conference on Product-Focused Software Process Improvement*, pages 243–258. Springer.

Giardino, C., Paternoster, N., Unterkalmsteiner, M., Gorschek, T., and Abrahamsson, P. (2015). Software development in startup companies: the greenfield startup model. *IEEE Transactions on Software Engineering*, 42(6):585–604.

Gomes, F., Santos, E., Freire, S., Mendes, T. S., Mendonça, M., and Spínola, R. (2023). Investigating the point of view of project management practitioners on technical debt-a study on stack exchange. *Journal of Software Engineering Research and Development*, 11(1):12–1.

Gomes, T. B. S., de Moura Loiola, D. A., and Santos, A. R. (2024). Technical debt tools: a survey and an empirical evaluation. *Journal of Software Engineering Research and*

Development, 12(1):8–1.

Junior, H. J. and Travassos, G. H. (2022). Consolidating a common perspective on technical debt and its management through a tertiary study. *Information and Software Technology*, 149:106964.

Junior, H. J. and Travassos, G. H. (2023). istdm: An evidence-based framework to support the management of technical debts in software projects. In *Workshop Anual do MPS (WAMPS)*, pages 73–75. SBC.

Mendes, J. G. P. (2020). Uma análise sobre o uso de metodologias ágeis de desenvolvimento de software em startups.

Oliveira, F. (2015). *Gerenciamento de dívida técnica em projetos de software utilizando Scrum: Uma pesquisa-ação*. Novas Edições Acadêmicas.

Pacheco, A., Marín-Raventós, G., and López, G. (2018). Designing a technical debt visualization tool to improve stakeholder communication in the decision-making process: A case study. In *Research and Practical Issues of Enterprise Information Systems: 12th IFIP WG 8.9 Working Conference, CONFENIS 2018, Held at the 24th IFIP World Computer Congress, WCC 2018, Poznan, Poland, September 18–19, 2018, Proceedings 12*, pages 15–26. Springer.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of management information systems*, 24(3):45–77.

Pompermaier, L. B. (2021). Modelo press: evoluindo a adoção de práticas de engenharia de software em startups digitais.

RIBEIRO, L. F. (2016). Uma estratégia baseada em critérios para apoiar a tomada de decisão sobre o pagamento de itens de dívida técnica. *DIss. de mestrado. Universidade Salvador*.

Rios, N., de Mendonça Neto, M. G., and Spínola, R. O. (2018). A tertiary study on technical debt: Types, management strategies, research trends, and base information for practitioners. *Information and Software Technology*, 102:117–145.

Rios, N., Spínola, R. O., Mendonça, M., and Seaman, C. (2020). 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: Rios et al. *Empirical Software Engineering*, 25(5):3216–3287.

Rubin, K. S. (2018). *Scrum Essencial: Um guia prático para o mais popular processo ágil*. Alta Books Editora.

Ruschel, B. (2019). *Guia Prático do Design Thinking: Aprenda 50 ferramentas para criar produtos e serviços inovadores.* Ebook Kindle.

Santos, C. G. d. et al. (2016). Um estudo empírico sobre a gerência de dívida técnica em projetos de desenvolvimento de software que utilizam scrum.

Seaman, C. and Guo, Y. (2011). Measuring and monitoring technical debt. In *Advances in Computers*, volume 82, pages 25–46. Elsevier.

Silva, V. M., Junior, H. J., and Travassos, G. H. (2019). A taste of the software industry perception of technical debt and its management in brazil. *Journal of Software Engineering Research and Development*, 7:1–1.

Stochel, M. G., Chołda, P., and Wawrowski, M. R. (2020). On coherence in technical debt research: Awareness of the risks stemming from the metaphorical origin and relevant remediation strategies. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 367–375. IEEE.

Tsoukalas, D., Chatzigeorgiou, A., Ampatzoglou, A., Mittas, N., and Kehagias, D. (2022). Td classifier: Automatic identification of java classes with high technical debt. In *Proceedings of the International Conference on Technical Debt*, pages 76–80.

Vieira, I. R. (2014). Avaliando a dívida técnica em produtos de código aberto por meio de estudos experimentais. *Diss. de mestrado. Instituto de Informática-UFG*.

Wang, C., Dai, M., Fang, Y., and Liu, C. (2022). Ideas and methods of lean and agile startup in the vuca era. *International Entrepreneurship and Management Journal*, 18(4):1527–1544.

Wieringa, R. J. (2014). *Design science methodology for information systems and software engineering*. Springer.

Wiese, M. and Borowa, K. (2023). It managers' perspective on technical debt management. *Journal of Systems and Software*, 202:111700.

Wohlin, C. and Rainer, A. (2022). Is it a case study?—a critical analysis and guidance. *Journal of Systems and Software*, 192:111395.