

Beyond the Code: The Value of Practicing and Evaluating Technical Debt Management

Mili Orucevic
Visma
Oslo, Norway

Maren Maritsdatter Kruke
Visma
Oslo, Norway

Antonio Martini
University of Oslo
Oslo, Norway
antonio.martini@ifi.uio.no

Abstract

Background: Effective management of Technical Debt (TD) is essential for improving software quality and sustainability. However, there is a lack of evidence about the benefits of TD management practices.

Objective: This experience report involving a large international company, Visma, aims to assess the impact of employing TD practices on an organization's capabilities and outcomes related to TD.

Method: This experience report describes the usage of two tools, TDMA and TD Pulse, to respectively track the practices and assess TD management capabilities. Over eight years, some teams in Visma adopted recommended TD practices (TDMA), while others either lacked formal TD practices or relied on ad hoc, informal methods. The TD Pulse questionnaire was then administered to both groups to evaluate their TD management capabilities and two key performance indicators (KPIs): time wasted due to TD and missed business opportunities attributed to TD.

Results: The analysis indicates that the implementation of TD practices has significantly enhanced various TD management capabilities and reported a better TD management outcome according to the KPIs, demonstrating clear benefits as shown by the results from TD Pulse.

Conclusions: This experience report suggests that organizations generally experience positive outcomes from adopting TD management practices, indicating that companies stand to benefit substantially from their implementation.

CCS Concepts

• **Software and its engineering** → **Software design tradeoffs; Capability Maturity Model.**

Keywords

Technical Debt Management, survey, empirical research, case study

ACM Reference Format:

Mili Orucevic, Maren Maritsdatter Kruke, and Antonio Martini. 2026. Beyond the Code: The Value of Practicing and Evaluating Technical Debt Management. In *International Conference on Technical Debt (TechDebt '26)*, April 12–13, 2026, Rio de Janeiro, Brazil. ACM, New York, NY, USA, 11 pages. <https://doi.org/10.1145/3794915.3795785>



This work is licensed under a Creative Commons Attribution 4.0 International License. *TechDebt '26, Rio de Janeiro, Brazil*

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2485-5/2026/04

<https://doi.org/10.1145/3794915.3795785>

1 Introduction

Technical Debt (TD) presents a daily challenge for software practitioners, involving both technical and non-technical dimensions. While TD manifests in suboptimal code solutions, its roots and impacts often stem from business and organizational factors. Therefore, a comprehensive approach is necessary for effective TD management, extending beyond tools for identifying code deficiencies [9].

A critical inquiry for software companies is whether it is beneficial to use TD management practices, and how to assess their effectiveness to ensure that resolving one issue today doesn't lead to new, potentially worse problems tomorrow.

Especially in the age of AI, where code is generated at an high pace, teams must become even better able to evaluate the Technical Debt (TD) they are accumulating (faster).

In order to manage technical debt, several practices have been proposed, related to identification, prioritization, etc. [7, 1, 13]. However, choosing and adopting practices is not easy and organizations are hindered by several factors [8]: it's difficult to provide a clear Return of Investment for managing TD. How can we assess whether using TD practices leads to effective TD management?

On the other hand, several capabilities (abilities) managing TD can be self-assessed by organizations [9], and some outcomes can be estimated (if not measured). For example, practitioners can be asked if they are able to infer the risk of the existing TD, if they know who is responsible for the budget spent on TD, or if they feel encouraged to repay TD. These are not practices, but are capabilities of an organization.

Finally, companies use Key Performance Indicators (KPIs) to understand if they are successful in achieving their goals. Two key achievements in managing TD are wasting less time and missing less business opportunities because of TD.

In essence, managing technical debt is an individual but also an **organizational skill**: it needs **practice**, its **ability** must be **assessed**, and it must lead to clear **benefits**.

Although some studies have investigated the existing practices that are used by practitioners to manage TD, no large study has shed light on whether such practices are effective. The main research question in this study is:

RQ1: *Does employing a set of TD practices lead to better perceived capability to manage TD and improved TD KPIs?*

In this experience report, we collected data from 131 practitioners from 91 teams on the use over 8 years of a suite of practices (TDMA) to manage TD in a large company. We also evaluated the self-assessed capabilities of such teams managing TD, including estimating the results in terms of wasted time and missed business opportunities (KPIs).

The lessons learned indicate that employing these practices is linked to several benefits, including improved guidance, increased motivation, and perception of reduced wasted time. However, not all aspects showed improvement with the implementation of these practices, though none were negatively affected either. In conclusion, the evidence underscores the positive impact of Technical Debt practices on software organizations.

2 Background and Related Work

2.1 TD management

The concept of technical debt (TD) originated as a metaphor for expedient design or implementation choices that accelerate short-term delivery while incurring future costs [3]. Over time, TD has been elaborated as a lens for understanding trade-offs across multiple levels of software engineering, from code and architecture to process, highlighting the need for structured decision-making and governance [5].

Systematic mappings and tertiary reviews synthesize the fast-growing TD literature, outlining themes, practices, and open challenges. A mapping up to 2014 finds empirical work centered on code-level and architectural debt, with fewer studies on process and test debt, and highlights inconsistent definitions and measurement gaps [7]. An extended mapping on identification/management reports static and code metrics as common for code debt, while architectural debt demands design-centric, expert-driven techniques [1]. A tertiary review of secondary studies reinforces persistent issues: inconsistent terminology, limited longitudinal evidence, and a disconnect between academic measures and practitioner adoption [11].

Practitioner studies show broad recognition of TD but uneven management: many teams track TD informally, while systematic quantification and portfolio-style approaches remain uncommon; time pressure and business priorities often drive debt decisions [4]. Large-scale investigations of architectural TD find it impactful yet under-detected by code-centric tools, with ad hoc practices, poor traceability, and a need for explicit policies [10]. Longitudinal evidence links TD to reduced productivity and morale, underscoring the need for proactive management [2]. Managerial perspectives emphasize visibility, prioritization, and alignment with business value-framing repayment as investment under uncertainty [12].

As for measurement, industry relies on proxies such as static analysis warnings, code smells, complexity metrics, and maintainability indices for code debt (e.g. the SQALE method [6]). However, systematic mappings and practitioner reports warn these proxies miss architectural, requirements, process and other types of debts, and that “debt interest” remains hard to quantify reliably [7, 1, 4].

Adoption of systematic TD tracking at scale appears challenging. Evidence from a survey and multiple case study across 15 large organizations reports low maturity and heterogeneity in TD tracking practices, limited integration with portfolio governance, and organizational impediments (e.g., competing priorities, measurement misalignment, and tool/process fragmentation) that make sustained TD tracking difficult to adopt broadly [8]. This aligns with broader observations that industrial uptake concentrates on code-level indicators integrated into CI/CD, with less mature support for architectural and organizational debt [1, 5], and that constraints

such as limited time, evolving requirements, and difficulty linking TD repayment to near-term business value impede proactive management [4, 12].

Another study shows how teams manage technical debt as part of routine agile work, combining preventive practices, ongoing monitoring, and opportunistic repayment that is often tied to feature delivery and release deadlines [13]. Reported impediments include scarce actionable metrics, competing product priorities, and limited time, which push repayment to be deferred or bundled into planned changes rather than handled through dedicated initiatives [13].

In summary, existing surveys and reviews establish that:

- TD is a multifaceted concept extending beyond code to architecture, requirements, and process [5, 7].
- Practitioners value the TD concept but face organizational and economic barriers to systematic management [4].
- Systematic TD tracking is difficult to adopt at scale due to low maturity, heterogeneity of practices, and organizational impediments [8].
- There is a sustained need for rigorous, longitudinal, and context-aware empirical studies, better operationalization of “interest” and methods that align TD decisions with business value [11, 1].
- These insights motivate further empirical investigation into TD practices, especially around scalable identification of non-code debt, decision support for repayment prioritization, and integration of TD management into product and portfolio governance.

It is therefore crucial to understand if TD practices can indeed lead to a better TD management and consequently to better outcome in managing TD. However, to the best of our knowledge, there are no large-scale studies, surveys or case-studies, where the impact of adopting TD practices has been investigated and substantial evidence has been reported.

2.2 Company details participating in the study

The studied company, Visma, develops and delivers mission-critical business software to all type of businesses both to the private and public sector, improving the work-life of millions of people around the world. The software simplifies and automates complex and manual work processes, empowering people’s everyday lives.

Visma is a collection of entrepreneurs and innovators, each bringing their own drive, vision, and leadership, while thriving within a dynamic and collaborative ecosystem. All entities in the overarching company have business autonomy and freedom, a key ingredient to the company’s success.

Visma focuses heavily on providing cloud-based software systems that enable organizations to automate and streamline financial, administrative, and business processes. Through numerous multinational subsidiaries organized using self-managed teams, the company develops solutions encompassing a wide range of areas, including accounting, payroll, human resources, procurement, and customer relationship management.

To support individual subsidiaries of the company, global processes are created as guidance for best practices. The individual subsidiaries adopt these processes as they see fit.

Such setting provides an excellent opportunity to study and reach out to several autonomous teams with various backgrounds using different practices.

Most teams are responsible for mission- and business-critical, customer-facing systems, including CRM/ERP platforms, payroll systems, healthcare services, and software for schools and kindergartens. A minority of teams maintain internal tools; however, these internal systems typically support the primary mission-critical applications, and high levels of technical debt in those systems would have comparably serious consequences.

3 Overview of the Approaches

3.1 Technical Debt Management Assessment (TDMA) at Visma: Practicing TD management

Technical Debt Management at Visma has the goal to establish a certain level of maturity of technical debt management, keep a technical debt risk register in the backlog and ensure technical debt is continuously tracked and mitigated.

The TDMA: To prevent and manage the technical debt of the software, the service delivery team need to have in place a processes for technical debt management. The Technical Debt Management Assessment will assess the team's current technical debt management practices, and guide the team towards how to improve it, by keeping track of technical debt in a structured way, and do this over time. In the assessment the teams go through how to; prevent, identify, document, analyze, monitor, communicate, plan and repay technical debt.

The first version of the TDMA was inspired by the research done in [13]: this research used the company as the case study with contribution from dedicated resources from Visma. TDMA was created as part of a digital transformation from yearly to daily continuous releases. The framework has since undergone multiple iterations—refining capabilities and assessment criteria—while preserving its core aim of producing a transparent, well-managed technical-debt process. The TDMA is reported in Table I and in Fig 1.

At the beginning, each year the teams performed a self-assessment of how they work with technical debt (**practices**). After a few rounds, the teams assessed their practices every 18 months, and finally it is done on an occasional basis, when the team has become mature enough. If the team does not adhere to the suggested practices, they create a follow up ticket in their backlog to implement the suggestion.

Figure 1 gives a simple overview of the flow of the technical debt process and the annual self-assessment they perform. Technical debt can be prevented after its identification or it can be documented, analyzed, planned, and then repaid. The sections monitoring and communication are continuous during the technical debt process. We will go more into depth on the different sections of the technical debt process below.

Each team works with technical debt differently. Some teams are quite strict while others tend not to be as strict.

Table 1 gives an overview of the technical debt process.

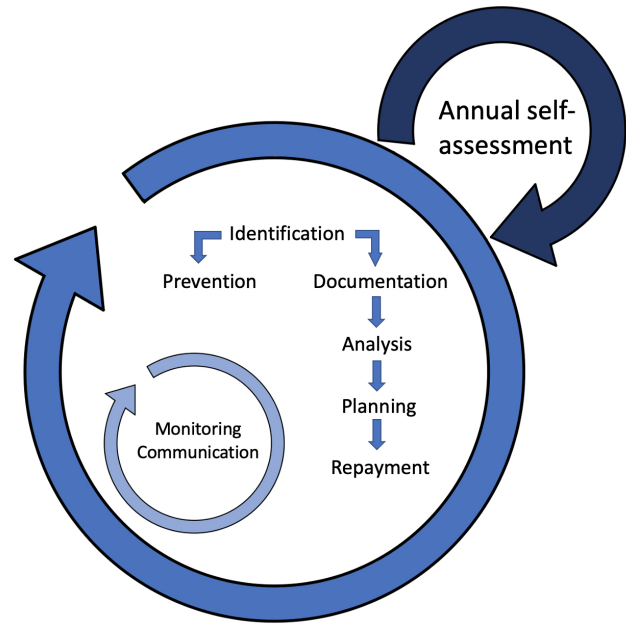


Figure 1: Technical Debt Process and practices assessed by the TDMA

3.1.1 Prevention. Code reviews are recognized as a crucial activity for preventing technical debt. They provide a platform for identifying and addressing technical debt before merging code into the main branch. Adhering to coding standards is considered a vital part of managing technical debt throughout the development process.

3.1.2 Identification. Teams identify technical debt through manual efforts during implementation and code reviews, using static analysis tools, during testing, and in response to incidents. The majority of teams actively look for technical debt while working on the code, aiming to improve the code quality as they go, similar to the "boy scout" principle of leaving the area in better condition than it was found. Tools like SonarQube and Coverity are widely used, which are effective in detecting technical debt.

3.1.3 Documentation. Technical debt is documented in the same backlog as other tasks, ensuring that they are prioritized alongside features.

Many teams utilize Jira to track technical debt, but not all identified technical debt is recorded. Some teams only log medium or long term debt, and sometimes close the tickets with "won't fix" resolution (basically use JIRA as a ADR (architectural decision record) while others omit that, and only log issues that will be fixed. Teams also fix minor issues directly without creating tickets to avoid unnecessary administrative overhead. Recording technical debt in Jira can provide leverage in discussions with management, allowing teams to justify time spent on addressing these issues.

When documenting technical debt, many teams use a specific TD label to categorize and filter items. While labeling practices are common, not all teams consistently apply labels for every ticket.

Section	Approaches
Prevention	The team follows coding standards and performs code review for all non-trivial changes
Identification	The team identifies technical debt during implementation and code reviews
	The team identifies technical debt using static code analysis tools (e.g. SonarQube or similar)
	The team identifies technical debt as part of operational processes like incident reviews and problem management
Documentation	Technical debt is documented in the same backlog as everything else
Analysis	Technical debt is analyzed and a severity score is calculated and documented in the teams backlog system
Monitoring	Different types of technical debt metrics is measured over time by looking at the sum of all severity scores
Communication	The team continuously communicates technical debt risks to stakeholders, focusing on the business and/or customer value of paying down the technical debt with the highest severity score (that has a ROI)
Planning	The team decides on what technical debt should be paid down next based on the severity scores
Repayment	Some of the team's capacity is always reserved for paying down technical debt (20% or more is the industry standard and highly recommended for technical debt), but only if there is a clear value for the team or the product

Table 1: Technical debt process

Some teams focus more on labeling during assessments rather than upon initial entry.

Including technical debt (TD) items in the shared product backlog rather than a separate or informal list increases visibility and accountability across all stakeholders (product management, business, and delivery). This prevents TD from remaining an internal engineering concern or being lost in personal notes, and makes the scope and recurrence of TD—whether a few large issues, many small ones, or recurring patterns—transparent to decision-makers.

This transparency brings TD into strategic discussions: it clarifies emergent dependencies, explains why some changes are costlier than they appear, and supports informed prioritization. TD should not be an afterthought; like security, it must be considered throughout product development. Representing TD in the backlog also enables deliberate trade-offs (e.g., “slow down to speed up”): allocating effort to debt repayment can increase long-term delivery capacity and reduce the risk of future firefighting.

3.1.4 Analysis. Teams can assess technical debt using a priority field (low, medium, high, critical) or by calculating a severity score, defined as the product of impact and likelihood: ($S = I$ times L).

While some teams emphasize calculating severity for technical debt, others prioritize using the priority field based on their experience rather than engaging in detailed analysis. The application of severity scores can vary; some teams do not rigorously calculate severity for every technical debt entry, with certain teams only conducting these assessments annually.

Teams often rely on their collective experience to establish severity and likelihood ratings. Some organize team meetings to collaboratively calculate these scores and decide on prioritization. Custom scales for assessing likelihood and impact are utilized, but the usual range is from 1-5, which provides a final severity score of 1-25.

3.1.5 Monitoring. Monitoring of technical debt is done a bit differently from team to team. Some teams have a continuous approach

where they even set some thresholds for either amount of TD issues open, or total severity of the TD issues, others are a bit more informal about this. All teams that use the specific label, and have a severity set on a TD issue, gets an “out-of-the-box” TD dashboard, that is automatically created for the teams, as soon as they start using TDMA. This shows the teams cadence of TD issues open, closed, days TD issues are opened, severity, trends over time, and heat maps. How often the teams review and use these dashboards differ from team to team but many use it as an easy overview towards stakeholders to signal if more time is needed to spend on repaying technical debt, and for other strategic decisions regarding the product.

3.1.6 Communication. Teams regularly communicate about technical debt, with some designating specific days or meetings to focus on discussing, analyzing, and prioritizing it.

Some teams need to involve stakeholders in discussions about technical debt, noting its resource implications during the prioritization process. In particular, stakeholders should be informed about technical debt as it affects the overall project.

3.1.7 Planning. Larger technical debt items that cannot be addressed promptly are included in a roadmap typically spanning over one year. Some teams engage in backlog grooming sessions to plan for upcoming periods. During these sessions, they prioritize technical debt based on its calculated severity.

Respondents employ different methods for prioritization: The majority indicate that prioritization is based on either severity or priority levels. While some teams follow a structured approach, others adopt a more flexible strategy, addressing easier fixes first to build a foundation for tackling larger issues later. Some teams hold regular meetings to discuss backlog prioritization, indicating a proactive approach to managing technical debt.

3.1.8 Repayment. The teams can decide to allocate from 5% to as much as 30% of the team's capacity for repayment, depending on the focus and the level of TD.

Only a few teams specifically use percentages for technical debt repayment, though most integrate repayment into their daily tasks.

Larger teams are generally better positioned to manage and repay technical debt, as smaller teams may struggle to balance production issues, new features, and technical debt.

3.2 TD Pulse: assessing the TD capabilities and TD KPIs

TD Pulse [9] is a novel, holistic approach created to assess the management of Technical Debt (TD) in large software companies, developed over a decade of research and practical experience. It addresses both technical and non-technical aspects of TD, recognizing that its roots lie not just in the code, but also in business, organizational, and process-related factors. This comprehensive method aims to help companies evaluate their current TD management capabilities and identify context-specific improvement areas.

The methodology includes a questionnaire designed to gather insights from both technical and non-technical roles regarding eight key aspects of TD management. This questionnaire takes about 12 minutes to complete and uses a Likert scale for responses, enabling a nuanced understanding of perceptions related to their organization's practices.

The assessed **capabilities** are:

- Awareness: Familiarity with the concept of TD and its impact, along with how TD may hinder the exploitation of business opportunities.
- Knowledge: Understanding what constitutes TD in a specific company, including identification, avoidance, estimation, measurement, and repayment.
- Education: The importance of how TD is communicated and discussed within the company to enhance awareness and knowledge.
- Encouragement: Support and motivation from high-level management for teams to engage in TD management and improvement activities.
- Process: Procedures in place for managing TD, including the integration of TD considerations into budget management and development processes.
- Organization: Clarity of roles and responsibilities regarding TD management within the organization.
- Tools: The tools and practices employed for identifying, tracking, and managing TD effectively.
- Prioritization: how do practitioners prioritize TD with respect to various prioritization aspects such as delays in feature delivery or cost of refactoring.

Finally, TD Pulse asks about two **KPIs**:

- Wasted time because of TD
- Missed business opportunities because of TD

TD Pulse was used by 232 participants to assess TD management capabilities in three large international companies in a previous study [9]. Several managers and technical roles also were surveyed

after reporting the results of the assessment to evaluate the usefulness of the tool (for more information, see [9]). The general evaluation was that the tool is useful to assess TD management and to provide concrete insights and suggestions for improvement.

The evaluation of TD Pulse showed how assessing the key aspects of Technical Debt (TD) management in a holistic manner reveals context-dependent gaps. TD accumulation and refactoring depend on several essential factors, with no single aspect in TD Pulse being deemed less important, as mentioned in the evaluation study [9].

Overall, TD Pulse demonstrated its effectiveness as a lightweight assessment method that facilitates meaningful evaluations of TD management, helping organizations to take actionable steps towards improvement.

In this work, TD Pulse was used to obtain a **snapshot assessment of the capabilities and the outcome of TD management (KPIs) of the studied teams at Visma**.

4 Study Design

4.1 Purpose and studied concepts

The purpose of the study is to understand if using **practices** is associated to improvement in TD management **capabilities** and if they can be associated with less time wasted because of TD and with less missed business opportunities (**KPIs**).

Practice is defined by Oxford Languages as *"the repeated exercise in or performance of an activity or skill so as to acquire or maintain proficiency in it."*

In contrast, **capability** is defined by Merriam-Webster as *"the ability or power to do something"*.

In summary, practice involves the repeated performance to improve a skill, while capability refers to the inherent ability to perform it. For example, a practice is to monitor code debt using SonarQube. A capability is the actual ability of the team to effectively use tools to identify and prioritize the right TD.

While TDMA, at the studied company, focuses on the use of practices, the TD Pulse also asks questions such as *"It's clear for the software development teams who is responsible to approve or deny budget"*, which is a capability.

In the survey, a simple question was asked about TDMA: *"Have you completed the company Technical Debt Management Assessment (TDMA)?"*. This question was answered with Yes/No and added to the survey described below.

The capabilities [9] consist mostly of a statement that is answered as a 4-level agreement Likert scale (strongly disagree to strongly agree). This involves various areas of TD management, as reported in Fig 2.

Finally, to understand the impact of the practices, we study KPIs. A KPI, or Key Performance Indicator, is, according to Merriam-Webster, *"a (measurable) value that demonstrates how effectively a company is achieving key business objectives"*. KPIs can be quantitative or qualitative and are used to track progress in various areas of (software) development. In this study, the two KPIs that we used are self-reported by respondents: The wasted time and the missed business opportunities caused by the presence of Technical Debt.

In particular, the questions asked in the survey were: *"On average, how much % of the total development time do you estimate was spent*

Aspects common to managers and technical		
Category	Sub-category	Input
Background	Experience	# years
	Role	Predef.
	Size of team(s)	number
Knowledge	anonymous ID	string
	Use of TD in the organization	4-Likert
	Existing guidelines to manage TD	4-Likert
Awareness	Test sentence to assess familiarity	4-Likert
	% Waste produced by TD	0-100
Encouragement	Missed business opportunities	4-Likert
	team encouraged by manager	4-Likert
	encouragement from other managers	4-Likert
Prioritization	encouragement from Product owners	4-Likert
	when are teams encouraged	4-Likert
Education	ranking of prioritization aspects	ranking
Organization	facilitation of education session	4-Likert
Process	clear responsible for budget	4-Likert
	TD decisions are followed upon	4-Likert
Tools	TD risk understanding	4-Likert
	systematic and iterative process	4-Likert
	dedicated time to manage TD	4-Likert
Tools	data and measures usage	4-Likert
	tools chosen by the teams	4-Likert
Aspects specific for one role		
Specific	Sub-category	Input
Managers		
Background	# teams managed	number
Knowledge	existing guidelines (specific)	qual, optional
Encouragement	high-level management encouragement	4-Likert
Process	usage of TD in budget	4-Likert
Tools	possible to compare TD across teams	4-Likert
	visualization of TD costs and risks	4-Likert
	perception if teams use tools	4-Likert
Specific	Sub-category	Input
Technical		
Knowledge	existing guidelines (specific)	qual, optional
Awareness	Hurtfulness of various types of TD	4-Likert
Organization	Clear technical responsible	4-Likert
Process	TD is removed	4-Likert
	Bigger TD is tracked	4-Likert
	code reviews are helpful	4-Likert
Tools	confidence in code analyzers	4-Likert
	confidence in risk assessment (tools)	4-Likert
	tools supporting TD level and planning	4-Likert

Figure 2: The questions asked in the capability and KPIs survey. Awareness is the area where we asked about KPIs. Picture extracted from [9]

by your team(s) on extra activities due to the presence of Technical Debt in the system(s)?" where the respondents needed to answer a number from 1 to 100.

The second one was "We have missed business opportunities because of existing Technical Debt in the system(s)" and the respondents could select Strongly disagree to Strongly agree (4 levels Likert scale) which were coded to 1-4.

4.2 Overall design

Our research design is showed in Fig 3. In 2015 the TD practices were introduced at the studied company. Such suite of practices have been named together as TDMA, or Technical Debt Management Assessment. Over the years, several teams at company have

adopted such recommended practices while others have not yet or have their own defined practices.

In 2023, we took a snapshot of the company with TD Pulse, evaluating the current TD capabilities and KPIs. In TD Pulse, we also added a question on whether the participant had used TDMA and a question to identify the teams and other company-related information (which was not analyzed as confidential). A total of 131 participants took the survey after a webinar explaining Technical Debt was attended by hundreds of participants. Right after the webinar, the questionnaire was opened for answers, and it remained open for three weeks. The average respondent time was 12 minutes to go through around 30 questions (varied a bit from managers to technical roles).

We then compared the answers from those participants answering that they used TDMA with those who didn't. This would give us insights on whether the practitioners using TDMA would give significantly different answer than those who didn't use it.

For space reasons, we report boxplots of the distribution of answers only for those variables for which there was an obvious difference across the groups of teams with the exception of missed business opportunity, since it's an important KPI. All the associations were manually investigated by all authors and discussed upon.

5 Results

5.1 Demographics of the study

In total, 40 Management Roles and 91 Technical Roles participated in the study.

Of these, the ones reporting using the TD management practices suite called TDMA were 12 managers (30% of the respondents) and 23 technical roles (25% of the respondents). The number of more specific roles is reported in Table II.

The technical participants were all from distinct teams (which means, there were 91 teams participating in the survey), and, although the size of the teams varied, most of the teams were composed of between 5 and 10 members (Fig. 4). We didn't report in the picture, for visualization purposes, 8 respondents, who answered between 30 and 65.

Category	Count
Managers	
Service/Product Owner	14
Team Manager	14
Other Management	12
Technical	
Developer	55
Architect	23
Other Technical Roles	13

Table 2: Role Frequency by Category

The participants' experience varied, however, it is clear how the majority of participants had between 10 and more than 20 years of experience, while a minor part were distributed between 1 and 10 years of experience (Fig 5).

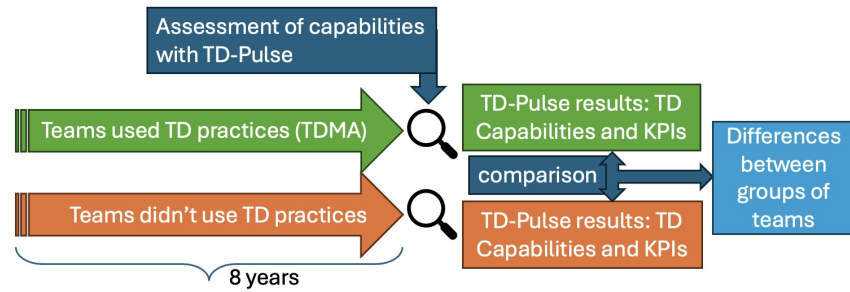


Figure 3: Study Design: after 8 years where some teams in the organization used the recommended TDMA practices, we studied if the teams using it reported better capabilities, KPIs and improvements. We could then study if there was a notable difference.

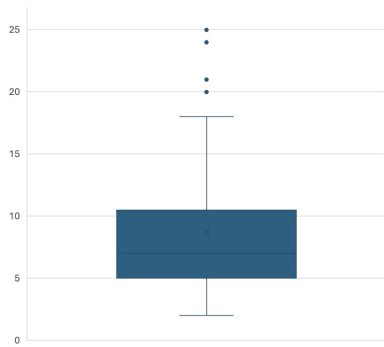


Figure 4: The distribution of team size related to the participants

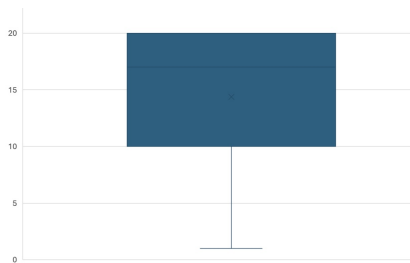


Figure 5: Distribution of experience for the participants in the study

We report boxplots showing the difference in the distribution of answers between those practitioners using TDMA and those who didn't. We differentiate between managerial roles and technical roles. We show the picture only when we found a significative difference between the two groups.

5.2 Association between TDMA and KPIs

First, we present the results related to TDMA and those variables in the survey identified as KPIs: the wasted time and the missed business opportunities because of TD.

As for the wasted time, there is quite a difference in the distribution of the answers between those who haven't taken the TDMA

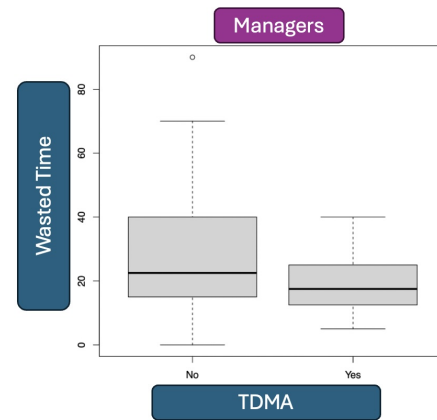


Figure 6: Difference in Managers' perception of wasted time because of TD between those using TDMA and those who don't

and those who have. The most striking result is that both the Managers and Technical roles who have used the TDMA perceive less wasted time because of TD. In particular, for the technical roles (Fig. 7), the average is 10% lower, while for managers (Fig. 6), although the average is not much lower, the distributions of answers is much less varied, especially lacking respondents reporting really high waste (over 40%) and very little above the 20%.

On the contrary, for missed business opportunities, the presence of TDMA does not show a large difference, as shown by Fig. 8 and Fig. 9.

5.3 TDMA and capabilities

We can see from Fig. 10 and Fig. 11. how having TDMA is associated with having **better guidelines providing guidance**, both for managers and technical roles.

The technical respondents using TDMA also clearly report to use more helpful code reviews to find TD (Fig. 12), they feel more encouraged to avoid and assess TD (Fig. 14, 13), they feel more confident using tools to identify, prioritize (Fig. 15) and visualize TD (Fig. 16).

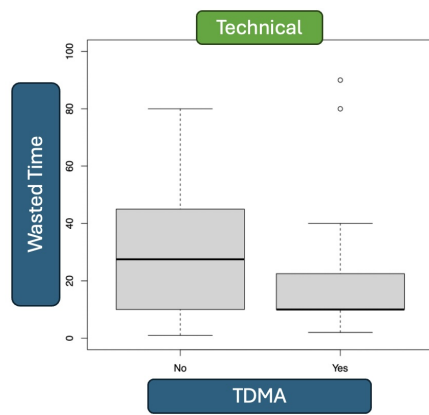


Figure 7: Difference in Technical roles' perception of wasted time because of TD between those using TDMA and those who don't

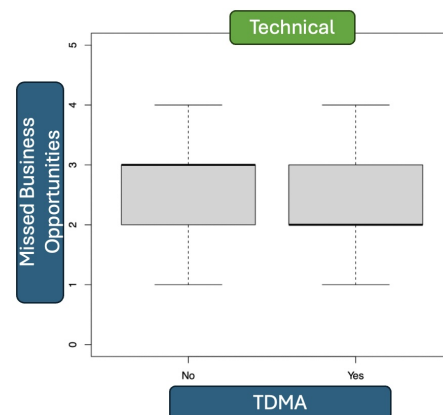


Figure 9: Difference in Technical roles' perception of missed business opportunities because of TD between those using TDMA and those who don't

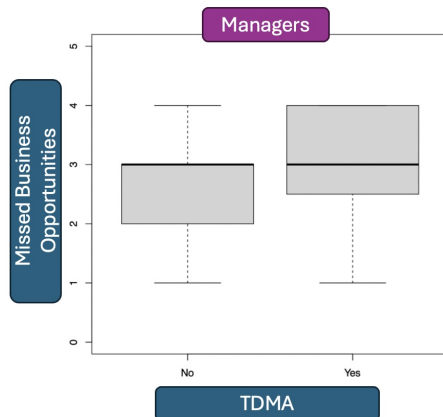


Figure 8: Difference in Technical roles' perception of missed business opportunities because of TD between those using TDMA and those who don't

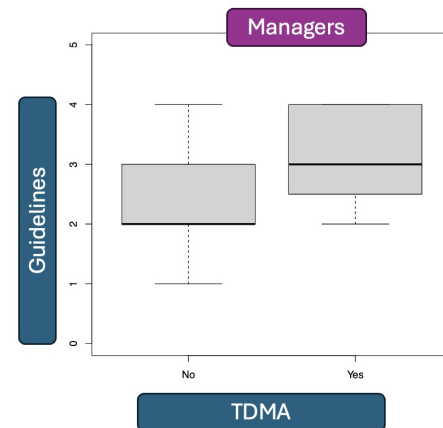


Figure 10: Difference in Managers' perception of having guidelines for guidance between those using TDMA and those who don't

6 Discussion

In this experience report, we assessed 91 teams in a large organization to understand if the teams using systematic TD management practices, introduced over 8 years, did also achieve better TD management capabilities and better outcome of managing TD.

6.1 Lessons Learned

This experience report leveraged a lightweight, holistic assessment (TD Pulse) to compare teams and roles that reported using a technical debt management suite of practices (TDMA), against those that did not. Several notable findings emerge:

Technical Debt is a useful skill: All in all, it is clear that using a suite of TD management practices is associated with the capabilities of the organization in managing TD. This is a crucial result for the TD community and for the software industry, because it's an actionable insight supporting the need for an organization **to develop a skill**: First, adopting TD practices (e.g. the TDMA suite

of practices), then assessing the current capability (e.g. with TD Pulse) and then assessing the major TD KPIs (such as both wasted time and missed business opportunities) to understand if the skill is being fruitful. This paper provides an example of the full chain of tools that can be used to make organizations better at managing TD: **practices, assessment and KPIs**.

Reduced perceived waste: Both managers and technical roles that used TDMA reported less wasted time due to technical debt, with technical roles indicating an average reduction of about 10%. Among managers, the average reduction was smaller, but the dispersion notably shrank, as few managers in TDMA contexts reported very high waste (40%), and responses clustered below 20%, suggesting more predictable, bounded waste.

No clear effect on missed opportunities: The presence of TDMA did not correspond to a strong difference in perceived missed business opportunities due to TD, indicating that either this KPI is

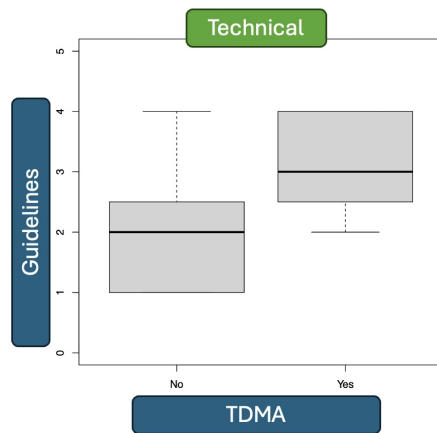


Figure 11: Difference in Technical roles' perception of having guidelines for guidance between those using TDMA and those who don't

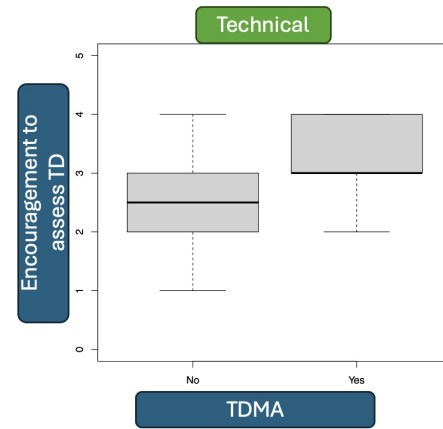


Figure 13: Difference in Technical roles' perception of being encouraged to assess TD between those using TDMA and those who don't

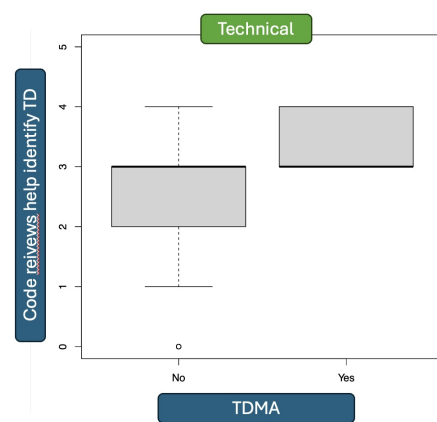


Figure 12: Difference in Technical roles' perception of having code reviews that help finding TD between those using TDMA and those who don't

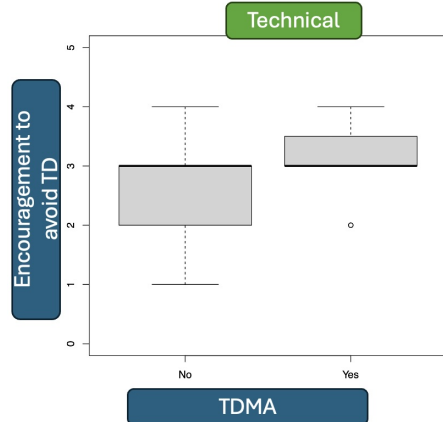


Figure 14: Difference in Technical roles' perception of being encouraged to avoid TD between those using TDMA and those who don't

harder to influence or that effects are lagging and not yet visible in perceptions. A possible explanation is that developers might have less understanding of the business implications of the TD, while they can feel the pain of the wasted time. Another explanation is that pre-existing TD in the system might have caused business opportunities despite good practices being in place.

Stronger capabilities where TDMA is used: Teams using TDMA (TD practices) reported having clearer guidelines (both managerial and technical perspectives), more helpful code reviews for finding TD (technical), stronger encouragement to assess and avoid TD (technical), and greater use of tools to identify, prioritize, and visualize TD. This clearly shows that **practicing TD management is crucial to become better at it**. This suggests that investing in organizational scaffolding (guidelines, review practices, and encouragement) is as important as introducing tools. This aligns with

[9], framing that TD management is an organizational capability requiring coordination across various aspects and not just tools.

Motivation behind the success: The studied company's motivation has been key to the successful implementation of the practices and their assessment: investing in Technical Debt (TD) management stems from the commitment to developing "mission-critical" software that plays an integral role in society. The whole company, starting from the top management, recognizes the importance of creating sustainable and long-lasting products that are capable of evolving over time. Therefore, it was crucial to consider and invest in managing Technical Debt at the appropriate moments in the company's development process.

Lightweight, holistic assessment is actionable: TD Pulse proved practical and informative. By simultaneously capturing capabilities and perceived outcomes (KPIs), it helps organizations

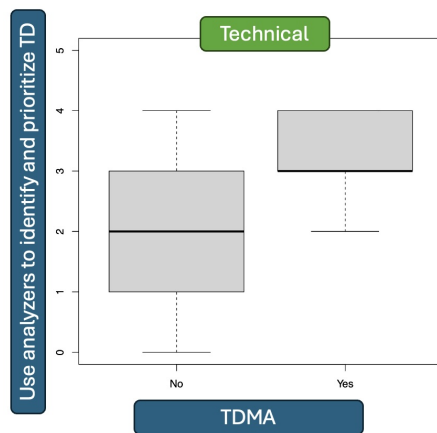


Figure 15: Difference in Technical roles' perception of using tools to identify and prioritize TD between those using TDMA and those who don't

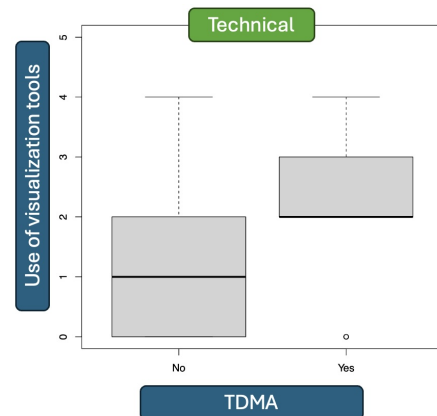


Figure 16: Difference in Technical roles' perception of using tools to visualize TD between those using TDMA and those who don't

identify meaningful gaps as opposed to merely cataloging practices in isolation.

Roles: Technical roles appeared to experience clearer benefits in day-to-day practices (code review effectiveness, tooling, encouragement), while managerial roles reported improvements in governance artifacts (guidelines). Investigating which interventions are most effective for different roles and how cross-role alignment influences outcomes. It is likely that different sub-roles (for example, architects and developers) would have reported different results and that they would benefit from different interventions. Tailoring interventions by role may increase effectiveness. In future work we will study more in depth the role aspect of TD management.

Is it worth the investment? Although a clear cost-benefit analysis was not possible, as the effort invested in setting up the practices was not tracked formally, several clues point to a clear benefit from investing in introducing practices, assessing them and analyzing KPIs. First, although the practices can require time in

the beginning to be set up, such investment will dilute over a few years, where the teams become more mature and automatize the mechanisms, reducing the overhead to just an assessment over 18 months or even less.

Using TD Pulse requires only 12 minutes per participant to get a large snapshot of an organization. In this study, one or two persons per team (a technical role and a manager) did take the survey, which would mean max 24 minutes per team. The assessment gives clear indications both in terms of needs for improvement and current impact of TD.

As the company itself declares: *"We received extensive positive feedback on the TDMA from both technical teams and stakeholders, highlighting its impact and effectiveness."*

These insights strongly support the benefits for teams to practice and self-assess their TD management.

6.2 Implications for research

Causality: The cross-sectional snapshot report only associations and not causal inferences. Future work should examine mediators (e.g., guidelines, code review quality, encouragement, tooling) to understand how TDMA reduces waste and whether these mediators differentially affect KPIs. However, it is unlikely that less wasted time would lead to better TD practices, and so we favor a potential causal relationship from practices leading to improved capabilities and KPIs.

First of its kind: This full-scale study is the first in literature where the effect of a suite of TD practices (TDMA) was assessed after long being used in practice by several teams (at least 23) in contrast with several other teams that didn't formally use it. This study provides quantitative evidence that applying TD practices is beneficial, which can be considered a milestone towards establishing **proven best-practices** in the software industry. Although more studies of this kind are needed, we consider this as a crucial contribution.

Future work: More analysis is needed to report the full results of both using the practices, reaching the capabilities and their connection to the KPIs. In our future work, we also plan to assess the company's TD capabilities again to understand changes in the practices and their effect in a longitudinal study, for example, studying the introduction of AI both to generate code (and additional TD), but also as a help to manage it.

6.3 Threats to validity

Internal validity: Teams adopting TDMA may differ systematically (e.g., maturity, leadership support, product criticality). Without randomization or controls, observed differences may reflect pre-existing advantages. Over the eight-year window, other organizational changes (toolchain, maturity, leadership) could influence perceptions independently of TDMA.

Self-report bias: Using TDMA, capabilities and KPIs are self-reported in a single instrument, increasing risks of halo effects and social desirability. Participants may misattribute causes of waste/opportunities to TD or to TDMA usage due to retrospection limitations.

Construct validity: “Wasted time” and especially “missed business opportunities” are perceptual and may be interpreted inconsistently across roles/teams. Lack of objective anchors can blur meaning. Self-reported “used TDMA” lacks granularity on depth, duration, and fidelity of practice adoption, potentially masking dose effects. Technical debt itself is a construct that is hard to pinpoint and can be interpreted differently by different practitioners.

To mitigate this threat, all the participants have participated in a webinar before taking the survey, where TD was introduced, reporting the definition from Dagstuhl and various examples.

External validity: Findings arise from one multi-team organization with its own culture, domain, and history. Generalizability to other contexts is uncertain. However, the company itself is a collection of various companies under the same umbrella. This mitigates this threat.

7 Conclusion

Using TD Pulse with 131 participants from 91 teams to evaluate an eight-year rollout of a suite of Technical Debt practices (TDMA) at Visma, we found that teams reporting using TDMA exhibit stronger TD management capabilities: Clearer guidelines, more helpful code reviews, greater encouragement, and broader tool use, and perceive less wasted time due to TD, with about 10% lower waste among technical roles and markedly reduced variance among managers. To our knowledge, this is the first empirical study that combines a long-running organizational TD management program with a full-scale holistic capability snapshot and explicit outcome KPIs, integrating both in one assessment. Practically, institutionalizing guidelines, review practices, and tooling appears to yield near-term operational benefits, and would likely justify the investment in setting up the practices, their assessment and the KPIs in the long term, also considering that, as the teams mature in managing TD, the investment is reduced drastically over time.

Acknowledgments

We would like to thank all the employees at Visma who participated voluntarily in this study.

This project has received funding from the Research Council of Norway Innovation Project for the Industrial Sector (IPN) Programme under grant agreement for project No. 340991

References

- [1] N. S. R. Alves, T. S. Mendes, M. G. de Mendonça, R. O. Spínola, F. Shull, and C. Seaman. 2016. Identification and management of technical debt: a systematic mapping study. *Information and Software Technology*.
- [2] Terese Besker, Antonio Martini, and Jan Bosch. 2019. Software developer productivity loss due to technical debt - a replication and extension study examining developers' development work. *Journal of Systems and Software*.
- [3] Ward Cunningham. 1992. The wycash portfolio management system. In *OOP-SLA '92 Experience Report*.
- [4] Neil A. Ernst, Stephany Bellomo, Ipek Ozkaya, Robert L. Nord, and Ian Gorton. 2015. Measure it? manage it? ignore it? software practitioners and technical debt. In *Proceedings of the 10th Foundation of Software Engineering (FSE)*. IEEE.
- [5] Philippe Kruchten, Robert L. Nord, and Ipek Ozkaya. 2012. Technical debt: from metaphor to theory and practice. *IEEE Software*, 29, 6, 18–21.
- [6] Jean-Louis Letouzey. 2012. The sqale method for evaluating technical debt. In *Proceedings of the 3rd International Workshop on Managing Technical Debt (MTD)*.
- [7] Zengyang Li, Paris Avgeriou, and Peng Liang. 2015. A systematic mapping study on technical debt and its management. *Journal of System and Software*.
- [8] Antonio Martini, Terese Besker, and Jan Bosch. 2018. Technical debt tracking: current state of practice: a survey and multiple case study in 15 large organizations. *Science of Computer Programming*.
- [9] Antonio Martini, Terese Besker, Torsten Posch, and Jan Bosch. 2023. Td pulse: assessing the systematic management of technical debt. *IEEE Software*. doi:10.1109/MS.2022.3226035.
- [10] Antonio Martini, Jan Bosch, and Michel R. V. Chaudron. 2015. Investigating architectural technical debt accumulation and refactoring over time: a multiple-case study. *Information and Software Technology*.
- [11] N. Rios, T. S. Mendes, and R. O. Spínola. 2018. A tertiary study on technical debt: types, management strategies, research trends, and base information for practitioners. *Information and Software Technology*.
- [12] Edith Tom, Aybuke Aurum, and Richard Vidgen. 2013. An exploration of technical debt. *Journal of Systems and Software*.
- [13] Jesse Yli-Huomo, Andrey Maglyas, and Kari Smolander. [n. d.] How do software development teams manage technical debt? – an empirical study. *Journal of Systems and Software*.