A Triple Bottom-line Typology of Technical Debt: Supporting Decision-Making in Cross-Functional Teams

Mark Greville
University College Cork
Markgreville@pm.me

Paidi O'Raghallaigh
University College Cork
P.OReilly@ucc.ie

Stephen McCarthy
University College Cork
Stephen.McCarthy@ucc.ie

Abstract

Technical Debt (TD) is a widely discussed metaphor in IT practice focused on increased short-term benefit in exchange for long-term 'debt'. While it is primarily individuals or groups inside IT departments who make the decisions to take on TD, we find that the effects of TD stretch across the entire organisation. Decisions to take on TD should therefore concern a wider group. However, business leaders have traditionally lacked awareness of the effects of what they perceive to be 'technology decisions'. To facilitate TD as group-based decision-making, we review existing literature to develop a typology of the wider impacts of TD. The goal is to help technologists, non-technologists, and academics have a broader and shared understanding of TD and to facilitate more participatory and transparent technology-related decision making. We extend the typology to include a wider 'outside in' perspective and conclude by suggesting areas for further research.

1. Introduction

Organisations increasingly depend on technology as an enabler of heightened business performance. When used appropriately, technology has a multiplier effect in supporting greater operational efficiencies, customer intimacy, and superior offerings. However, compromises are sometimes made in the rollout of technology delivery programs. Technical debt (TD) is a metaphor used to describe compromises made in technology delivery decisions, involving the exchange of short-term benefit for longer term 'debt' to be repaid later. Ward Cunningham coined the metaphor in 1992 as follows: "*Shipping first time code is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite"* [1]. TD reflects the cost of additional future rework caused by delivering a limited solution now rather than a more complete solution that would take more time and resources.

TD can be perceived as debt 'borrowed' by technology staff on behalf of their organisations. It is presented in the literature as emanating from decisions made by technologists where: (i) decisions are made in isolation from the rest of the company [2], [3]; (ii) the impact of decisions may be invisible to those outside IT [4]; (iii) the impacts can cause constrained options for future growth through increased costs [2], [5]–[7], operational risk [8], staff issues [9], [2], [10], [7] and ethical challenges [11], [12]; and (iv) in extreme cases, the impacts can be a threat to the very existence of a company [8], [13].

The aim of this paper is to challenge this narrow perception of TD, explore a broader definition of TD, and examine its impacts across the organisation and beyond. Our aim is to support cross-functional teams in making more participatory and transparent technology-related decisions. Through an extensive thematic literature review and a review of technology-related decisions reported in the media, we create a typology of impacts of TD on organisations. Later, we extend the metaphor of TD and the typology by considering the concept of the 'triple bottom line' developed by Elkington [14], [15] that examines broader '*people, planet, and profit*' concerns. The final typology combines both inside-out and outside-in perspectives across eight categories: Financial, Customer, Growth, Strategic, Internal Process, Economic, Social, and Environmental. We present the typology as a tool for technologists, non-technologists, and academics to expose a broader view of the 'debt' in play when technology-related decisions are being made. It gives those inside and outside the IT department a model and shared language to understand TD and it allows them to get involved in a joint decision-making process. The typology also offers academics a framework for identifying potential areas for future research.

We structure the rest of the paper as follows. We present the background to the topic (Section 2) and an overview of the methodology used in this study (Section 3). Following this, we synthesize the key elements of the typology (Section 4) by including examples of TD derived from academic papers and media reports. Next is a discussion and extension of

URI: https://hdl.handle.net/10125/79408
978-0-9981331-5-7

the typology (Section 5). We finish with the implications and opportunities for further research in academia and practice (Section 6).

2. Background

The concept of debt has existed for at least 5,000 years[16]. Debt is defined as a "*borrower's obligation to the lender from whom he has received funds*" [17]. Once managed correctly, debt can be a valuable tool for companies, allowing them to leverage funds beyond those generated by their revenues [18]. Technical Debt (TD) is a very recent concept. Ward Cunningham originally coined the term in 1992 [1]. His company was designing a software product for the finance industry. Because the technical complexities were beyond the understanding of his non-technical manager, Cunningham used the metaphor of debt to explore trade-offs in delivering the software. Each time they learned something new about the business problem, they could update the software to reflect their new understanding. However, if the updates failed to keep up with the changes in understanding, this caused a misalignment, which resulted in a 'debt' that would need to be paid back in some form in the future. TD if managed well, can be beneficial when 'leveraged' to increase productivity in the short term, and may be useful in growing a business or creating new opportunities [2].

While its original focus was on object-oriented software development, its use has grown to encompass many other areas such as IT infrastructure [2], testing [3], documentation [4], architecture [5], [6], build process [2] and systems integration [7]. Studies have been made across varying types of environments and use cases: Technology start-ups [8]; Open-source systems [9], [10]; Digital platforms [11]; Machine learning [12] [13]; Automated production systems [14]; Telecoms [7]; Social Networks; and Fortune 500 companies [5]. More recently, Rolland et al. widened the scope of TD and introduced the term Digital Debt which they defined as the "*buildup of technical and informational obligations that affect a platform's maintenance and evolvability as part of a user organization's digital infrastructure*" [19].

Tom al. [2], Li et al. [10] and Alves et al. [20] outline directions for future research based on comprehensive reviews of the literature on TD. Tom et al. call out the need to qualify associations of different types of TD to different impacts, and to establish metrics to quantify those impacts. Li et al. suggest the need to look at TD beyond debt generated by software code and to review aspects of TD not currently measured by existing tools to find gaps and directions for future research. Finally, Alves et al. suggest further research into TD identification and management.

Prior research has stressed the importance of cross-functional integration between IT and business groups in decision-making processes [21]. However, there is also a recognition of the inherent difficulties faced by cross-functional teams. Functions require specialization in performing their own tasks successfully but can have different perspectives on work and the organization [22], [23]. Differences in local understandings, expertise, and experience create inconsistent viewpoints. Neither technologists nor business leaders have a complete perspective of the organisational context [23]. Managers seeking to understand the impact of technical debt therefore face the challenge of integrating the perspectives of diverse functions during decision-making.

3. Methodology

We performed a two-phase thematic literature search, first we searched Scopus, and then AIS to return additional IS conference papers. We applied the following inclusion and exclusion criteria - papers should be: (a) from computer science or information systems domains; (b) exclude non-peer reviewed papers (e.g. blogs, trade papers, grey literature); (c) ignore duplicate papers; (d) ignore papers not written in the English language. The authors read the abstract, introduction, and conclusion of each paper to ensure that TD was a primary focus of the paper. We excluded any paper not meeting this criterion. The resulting set of papers was then reviewed in detail by the authors. A list of papers reviewed is available at http://dx.doi.org/10.13140/RG.2.2.10569.67688/1 .

Noted biologist Crowson [24] suggests that "classifying things is perhaps the most fundamental and characteristic activity of the human mind, and underlies all forms of science". Traditionally, typologies have been viewed as classifications rather than theories. However, Doty & Glick [25] argue that such a conclusion, while unsurprising, would be incorrect. This is because of the wide scale misunderstanding of what typologies are and how they should be built. They assert that typology building is a "*unique form of theory building*" and typologies are "*complex theories*" that can be "*subjected to rigorous empirical testing*". Drawing on this literature review, we developed a typology of the types and impacts of technical debt by adopting the typology-building framework advocated by O'Raghallaigh et al. [26] – see Figure 1.

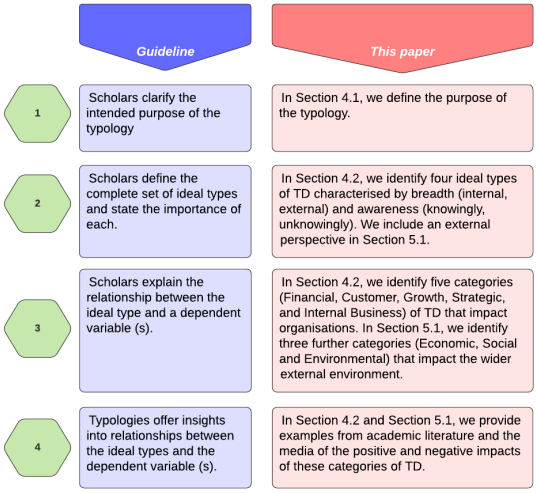

Figure 1: Steps for creating 'good' typologies

In the next section, we present the typology that resulted from following these steps.

4. Development of a Typology of Technical Debt

4.1 Purpose of the Typology

The technical debt metaphor first introduced by Ward Cunningham in 1992 quickly spread as a concept in technology practice [1]. His original quotation was as follows "*Shipping first time code is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite... The danger occurs when the debt is not repaid. Every minute spent on not-quite-right code counts as interest on that debt. Entire engineering organizations can be brought to a stand-still under the debt load of an unconsolidated implementation, object-oriented or otherwise*." [1]. Many papers quote this either partially or in full e.g. [2], [27]–[33]. They imply TD is a conscious choice to implement a shortcut *knowingly*, such as delivering incomplete code that is not optimal. They look at the impacts of these coding choices *internal* to the IT department. Some papers (e.g. [7], [29], [34]–[36]) extend the original metaphor to include aspects beyond code. For example, McConnell continues to focus on the internal impacts of TD but broadens it to include "*a design or construction approach that's expedient in the short term, but that creates a technical context in which the same work will cost more to do later than it would cost to do now*" [37].

Though many studies adopt either Cunningham's or McConnell's definitions, there is some disagreement on which decisions give rise to TD. Li et al. details some of this disagreement on how broad or narrow the scope of TD should be and whether: (i) TD refers to code only, or to the broader project lifecycle; (ii) whether deferred functionality is considered TD or not; (iii) whether defects are TD; and (iv) whether trivial code issues are in scope.

In 2009 Cunningham reflected on how others were using the metaphor and explained that TD went further than code and at its core is about an incomplete understanding of a *system* [38]. He explained TD was sometimes being misused by others to justify producing "*code poorly with the intention of doing a good job later*". He stated that he is not in favour of writing code poorly, but he is in favour of writing code to reflect one's current (but possibly incorrect) understanding [38]. He stated he had not intended offering TD as a way of knowingly taking shortcuts. However, none of the papers reviewed in our study reflected this clarification. Indeed, it can be argued that the definition of TD has now moved beyond the original intentions Cunningham had for it.

Both Cunningham and McConnell assume that TD is taken on *knowingly*. But others challenge this perspective. For example, Fowler introduced the TD quadrant in 2009, which is widely referenced in the literature [2], [8], [10], [27], [29]–[31], [33], [34], [39]. He introduces two dimensions: *reckless/prudent* that looks at the risk appetite of the decision makers, and *deliberate/inadvertent* that looks at whether TD is knowingly taken on.

In Table 1, we build on the seminal and emerging literature to categorise these different perspectives. Accumulated TD is made up of intentional and unintentional debt [40], [10]. We differentiate between knowingly (intentional) and unknowingly (unintentional) accumulating TD. *Intentional debt* is taken on deliberately to achieve some perceived benefit. An example might be the release of a product feature faster than a competitor, but at the cost of making some expedient (but possibly non-optimal) technical decisions. These decisions may contribute to higher future maintenance costs, but this cost may be bearable once it is kept visible and under control. *Unintentional debt* is more destructive because it is initially incurred without the knowledge of the team, often because of a lack of experience or lack of communication. For example, a developer might choose a low level solution which has a detrimental impact on the overall system architecture without realising that the solution will create serious issues for others [39].

In Table 1, we also differentiate between internal and external views of TD. In their literature reviews, Li [10] and Tom [2] look at both internal and external

effects of TD. By internal view, we mean one where the studies concentrate on concerns and effects inside the IT department, while the external view considers effects across the wider business.

Table 1: Different categories of TD

	Knowingly	Unknowingly
Internal	Ex-ante recognisation of impacts from technology-related decisions on IT, e.g. Cunningham, McConnell.	Failed ex-ante recognisation of impacts from technology-related decisions on IT, e.g. Brown and Ernst.
External	Ex-ante recognisation of impacts from technology-related decisions beyond IT, e.g. Sculley, Rios.	Failed ex-ante recognisation of impacts from technology-related decisions beyond IT, e.g. Tom, Woodard.

Several papers take an internal view and consider TD which is taken on knowingly. For example, Potdar et al., Huang et al. and Digkas [3], [41], [42] study TD which is taken on knowingly by the decision makers in IT. Potdar et al. analyse four large open source projects for comments, finding a high positive correlation between the experience level and the amount of TD produced. Huang et al. builds on Potdar et al. and uses machine learning to review comments in source code to identify TD at a more precise rate than Potdar et al. Digkas et al. analyses 57 Apache open-source projects at a code level and look at the rate of TD remediation to understand the lifecycle of TD. None of these papers consider the impacts of TD outside IT.

Sculley et al. also look at TD taken on knowingly but look at the impacts this can have externally [12], [13]. They use TD as a lens through which the impacts of machine learning can be understood. They discuss an example where a machine learning model could incorrectly adjust the price of shares on the stock market because of TD.

Brown et al. and Ernst et al. each look at TD taken on both knowingly and unknowingly and investigate from an internal view of the IT department. Brown et al. [27] offer an invitation to the software engineering community to research how to best manage TD, focusing on further research in seven areas, all of which are primarily code related activities: refactoring, architectural issues, identifying the dominant sources, measurement, process issues, monitoring and non-code artefacts. Ernst builds on this work using a survey of 1831 software practitioners

Some studies investigate TD taken on both knowingly and unknowingly and investigate impacts outside the IT department. Woodard et al. look at the impact of TD as a limiting factor on Design Capital and how it can impact advances in a firm's digital strategy. Tom et al. look at the effects of TD beyond the technology department, and its impact on morale, productivity, quality, and risk. There is an actual monetary cost of TD as quality issues result in potential loss of sales. There is also a discussion of the environmental debt, whereby TD can cause operational and security issues leading to potential brand damage.

The purpose of the typology presented next in this paper is to give a shared language for IT staff, business people, and academics to understand the impacts of TD across the whole of a business.

4.2 Impacts of TD

In taking our cue from extant literature, the initial version of the typology examines the impact that TD has across an entire business. We now discuss the categories of impact identified in the literature. We are influenced by the balanced scorecard in our choice of categories. The balanced scorecard explores the realisation of current and future value across different perceptions of performance, namely Financial, Customer, Internal Process, and Learning and Growth (Kaplan et al.) [43]. This is achieved through an evaluation of metrics related to different functions in an organisation and the working environment more broadly. While the original balanced scorecard was intended as a strategic management tool for assessing the performance of organizations, more recent research has adapted it to the context of project teams [44], [45]. We can readily map the internal impacts of TD identified in the literature to the categories of value identified by Kaplan et al. We illustrate these types of internal TD impacts through examples derived either from the academic literature or from the news media.

4.2.1 Strategic Impacts. TD can be *reputational in nature*. For example, TD can have a positive effect where a firm gets a reputation for putting the customer needs first. Organisations can choose to take on TD to meet customer demands more quickly rather than waiting for the most elegant or robust technology solution [46]. On the other hand, taking on TD can also have a very damaging reputational impact. For instance, redundant code which was no longer needed had a significant impact on American global financial services firm Knight Capital [11]. In 2012, Knight Capital inadvertently caused significant fluctuations in the prices of 148 companies on the New York Stock

Exchange, resulting in a $440 million pre-tax loss [47]. A piece of its legacy code used for routing equity trades had been disabled but never physically removed from the firm's router. A subsequent rollout of updated software inadvertently re-enabled the code. This caused over 4 million orders to be incorrectly sent into the market to fulfil 212 customer orders. This error caused enormous issues for the stock markets and severely damaged the firm's reputation. The firm struggled to recover and was acquired in late 2012 [11].

Some TD can be *security related in nature*. Taking on TD to fix security issues rapidly can be a positive short-term measure. However, if a company does not actively manage it, un-remediated TD can have a longer term impact on a firm's security, causing a direct impact on the firm's wellbeing [10]. Security vulnerabilities can be exposed by the continued use of technology which has already reached the end of life [2]. Spanos and Angelis reviewed 45 studies and showed that for companies affected by an information security event, over 75% of cases result in a statistically significant impact on their stock prices [48].

4.2.2 Internal Process Impacts. TD can be *operational in nature*. Woodard et al. describe a case study that shows both the positive and negative impact that TD can have on operations [8]. The paper describes a three-way merger of cable TV, residential broadband, and mobile businesses into a single company. This company (Infocom) inherited three separate billing and account management systems which were costly and inefficient to run. Initially, taking on TD (in this case from using three separate systems) allowed the company to manage customer accounts and billing immediately. Infocom built a common database schema between the systems, but this required manual interventions to reconcile errors between the systems. The extent of these operational interventions resulted in the firm becoming overwhelmed, forcing abandonment of the common database project (and a significant write-off of the prior investment in it).

TD impacts *staff turnover*. If a company doesn't take on TD, it often indicates a conservative, slow moving approach. This can result in an environment lacking dynamism and can cause dissatisfaction among staff. TD used in the right way can give a short-term productivity boost to a team, but this comes with a longer-term cost. As TD accumulates, staff must contend with diminishing levels of productivity [49]. If it's not managed and continues to grow, staff become increasingly disillusioned and are more likely to leave the organisation [2]. For a manager, carrying TD may be an acceptable risk, but a high level of TD makes a developer's job more difficult [10].

4.2.3 Financial Impacts. Capitalist economies are based on investing capital with the expectation of a positive return on the investment. This view has traditionally led companies to measure themselves by this yardstick, looking at the return to shareholders in a public market, or to a return to private investors in a non-public one. Balance sheet accounting concentrates on metrics which originate inside a company, such as cash flows, costs, risks, size of market, growth rates, availability of resources, and so on. In any company whose business activities require the use of technology as part of its operating model, TD can occur and can have an impact in a variety of ways on inside metrics, often in ways that are initially invisible to business stakeholders and investors.

TD impacts the *cost of change*. This choice is really a 'pay now or pay later' scenario. TD can be a positive choice if incurring TD saves significant upfront investment [8]. However, if TD is incurred, the interest accruing must be paid. If left untreated, this interest continues to grow. Nugroho shows that as debt increases, the cost of maintenance also increases [40]. This means the cost of changes in a system rise as TD rises [2].

When left unmanaged, TD can cause *technical bankruptcy*. The more TD accrues, the closer a system gets to bankruptcy. Technical bankruptcy is the point at which all new work on a project ceases and one of two things happen. Either all new development is halted while the TD is paid down, or a full rewrite becomes necessary [2]. Technical bankruptcy can be positive or negative depending on where in the lifecycle the system is. If a system is past its expected end of life, this implies that the company is extracting unexpected value from the investment, which is a positive. If a system has not reached its expected end of life, the requirement to invest in a rebuild adds financial burden to the organisation. In extreme cases, TD can cause the cancellation of a program of work. A notable example is the case of the HP TouchPad tablet. The product's core software suffered from architectural flaws, and this resulted in the tablet's withdrawal after only seven weeks on the market [8], resulting in a significant loss of investment and reputation for HP.

4.2.4 Customer Impacts. Taking on TD has a direct impact on market *opportunity*. In a digital world, a lack of speed can be detrimental to the future of organisations. Firms delivering digital platforms must have a technology stack which allows fast delivery in the early stages of development. Growing the user

base quickly is vital. Firms with an early advantage here are extremely difficult to catch, even if competitors have better technology. So, firms often take on TD to gain this 'first mover' advantage. However, firms who have incurred a high level of TD can be slowed down by it and therefore begin the race to market at a significant disadvantage [50].

TD can contribute to *customer satisfaction* [51]. TD can allow companies to release additional features more rapidly in an agile fashion [31], and to prioritise customer satisfaction over quality of delivery [30]. But because TD can indicate a lack of quality, it can also cause customer dissatisfaction, which can result in customer turnover and a loss of sales [2]. This can be seen in the case study of Infocom [8]. After the merger of the three companies, Infocom found that TD resulted in an upsurge of customer queries, problems, and general dissatisfaction.

4.2.5 Growth Impacts. TD impacts a company's *ability to innovate*. Firms with a low level of TD can use their position to create options for innovation. Rolland et al. discuss this in depth, with new technologies becoming facilitators of innovation, as long as TD in the guise of 'digital debt' is manageable [8], [19]. Firms with high TD, those who delay repayment of the debt inherit a reduced capacity for innovation. These firms cannot react to the needs of the market as fast as competitors, or at all in some cases [12]. In the mobile device arena, both RIM (makers of the Blackberry device) and Nokia found themselves unable to react to new entrants, such as Apple and Google Android, because of architectural TD [8].

5. Discussion and Expansion of the Typology

As we can see from Section 4, TD has the capacity to influence the fate of a business, from its impact on its customers, its operations, its financials, its employees, and its future success. This impact can be positive or negative depending on the nature of the TD and how it is managed.

Many companies have moved from inward-only metrics of success, such as when using the Balanced Scorecard, towards a more holistic view, where long-term environmental and social concerns are now also incorporated into business planning and corporate sustainability [52]. This approach is not always as selfless as it may seem, it may be an attempt by a business to safeguard future financials. If society unravels, the economy degrades and customers become impoverished, it then will become increasingly difficult to do business.

In 1997, Elkington introduced the concept of a triple bottom line (TBL) [14]. In looking at sustainability of businesses, he discussed the need to ensure that actions taken by companies today should not limit the range of options open to future generations. He advocated looking beyond the traditional model of measuring a company by financial or 'inside' metrics only, and instead looking at a TBL that adds social and environmental imperatives to traditional economic ones. The TBL advocates that companies should commit to focusing as much on social and environmental concerns as they do on profits. TBL argues that instead of one bottom line, there should be three: profit, people, and the planet. This was further advanced by Dyllick and Muff [15] who contributed a business sustainability model. This model suggests that a sustainable company should move from an inside-out perspective towards an outside-in perspective - beginning by asking how a business can solve global challenges, and then developing strategies and business models to overcome them.

In 2020 Kaplan and McMillan discussed the importance of incorporating the triple bottom line of financial, environmental and societal factors into a balanced strategic perspective needed to run a company [53].

When we re-examine the typology derived from current literature and presented in Section 4, we see a strong focus on a narrow inside-out perspective. Further to this, the external impacts of TD identified in the literature can be mapped to the TBL of profit, people, and the planet. We therefore advocate an important extension to the original typology presented in Section 4 to include an outside in perspective.

The outermost ring in Figure 2 presents an expansion of the typology of TD based on this outside-in view of business. This adds environmental, economic and social measures to the existing middle ring, which primarily focuses on measures inside a company.

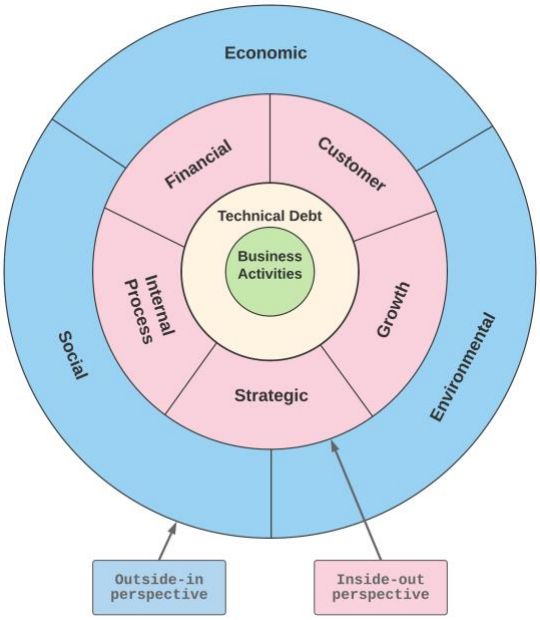

Figure 2: Typology of TD

5.1 Environmental Impact of TD [Planet]

The Environmental Bottom Line pertains to the environmental sustainability of a company's practices. The goal is to minimise any impacts on the environment, and to benefit the natural order where possible. A Triple Bottom Line (TBL) perspective advocates the management of a company's energy and raw material consumption in a way that poses minimal disruption to the ecology of the planet, for example through reducing waste and disposing of hazardous materials in a safe and legal manner. Otherwise, the company might be viewed as complicit in increasing long term costs which must be re-paid by society.

TD can generate environmental costs through its *energy consumption practices*. For instance, Bitcoin mining has become a technological phenomenon since 2010. Constantindes explains how the design of the Bitcoin Core allows between 5 and 7 transactions per second, compared with 25,000 per second for Visa. This 'design debt 'has a serious consequence for the environment, with one report suggesting that energy consumption for all households in Iceland was less than used for bitcoin mining [50].

TD can lead to a high risk of *environmental hazards*. IT systems have a part to play in many vital industries such as water, power generation and distribution and gas. The existence of TD can increase the chance of issues arising. In a case study, which looked at the process automation industry, Sandberg, Holstrom and Lyytinen showed how the company accumulated TD in its move to an 'Industrial IT' strategic initiative to integrate thousands of IT systems [54]. These process automation systems run in dynamic and demanding environments with a "*risk of significant environmental hazards*". We have already witnessed examples of disasters caused by technology failures, such as the Maroochy Shire Sewage spill where up to one million gallons of sewage was released into rivers and coastal waters in Australia [55], the Stuxnet attacks which destroyed approximately 20% of Iran's nuclear reactors [56] or the Deepwater Horizon oil spill disaster which released 779 million litres of crude oil in the Gulf of Mexico [57].

5.2 Social Impact of TD [People]

The Social Bottom Line pertains to how a company treats people in its employment, in communities, and in broader society (particularly if it effects the company's fortunes). A TBL perspective advocates that the company sees an interdependence between the interests of corporate, labour, and other stakeholders. A company could set goals to avoid exploitation of vulnerable people (e.g. use of FairTrade, avoiding child labour, paying a living wage).

TD can raise *ethical costs*. For example, the use of feedback loops in systems using Machine Learning can result in costs. A feedback loop happens when a model consumes some of its own output as an input. The most insidious type of feedback loop is known as the Hidden Feedback Loop [11], [12]. Here loops may develop between two otherwise disconnected systems. As one model changes, it influences the output of another model. There is the hypothetical scenario of two stock market prediction models, where improvements/bugs in one influence the buying behaviour of the other.

With the increasing prevalence of machine learning models in the real world, we can extend this TD scenario to ethical concerns in systems. Ntoutsi et al. discuss the COMPAS system which predicts higher rates of re-offending for black inmates than the actual risk (and higher rates than for white inmates). They also discuss the Google Ads tool, which showed significantly fewer higher paid job ads to women than to men. Other issues have been noted in areas such as credit scoring, automated screening of job applicants and profiling of civilians by police departments [58]. These biases can potentially undermine the sense of fairness and justice that is required for society to prosper.

Social media is firmly entrenched as part of modern life. Social media companies are fast moving with a '*release quickly*' mindset which comes hand in hand with some level of TD. However social media

has many issues yet unanswered for its role in society. There are issues with addiction to the medium itself [59], the spreading of violent extremism and terrorism [60], along with the fake news phenomenon which has caused a hardening of stances politically in the US since 2016 [61].

5.3 Economic Impact of TD [Profit]

The Economic Bottom Line is a more holistic view of economic performance. A company's bottom line often refers to profit and earnings per share and is made up of the value of total assets minus total liabilities. Traditionally, capital in production is made up of physical capital (machinery, buildings) and financial capital. In the digital economy, human capital and intellectual capital must be added. In the world of sustainability, social and environmental capital must be added, reflecting the cost/benefit the organisation has on the overall environment and society. As companies move to different calculations of their economic impact, the potential impact of taking on TD needs to be considered on intellectual and human capital, as well as social and ecological capital.

6. Conclusion

Overall, the literature presents Technical Debt as a useful metaphor, defining the gap between the technical implementation of a system at a point in time and the ideal implementation based on complete knowledge. TD is predominantly seen as an issue for technologists, however our typology shows how TD has an impact well beyond the confines of the IT department and indeed beyond the organisation. The decision to take on TD can have consequences beyond internal measures (such as profit); it can also impact people and the planet. Further research into how the decisions to take on TD are made by individuals and groups is needed. Research looking at TD through the lens of an outside-in business perspective is missing but we suggest that it is required to offer a more holistic perspective on technology related decision making.

TD provides the mechanism to assist in exploring the link between technology-related decisions and the future wellbeing of organisations. However, this mechanism remains underdeveloped. This paper finds that TD is not well understood and on its own does not contribute to better decisions. More effective use of TD demands that technologists, non-technologists, and executives embrace and understand the impact of technology-related decisions. In this paper we sought to extend the use of TD to support more collaborative discussions include both inside-out and outside-in perspectives.

While taking on financial debt is for the most part a deliberate action in that one takes it on by choice, this is not always the case with TD, which is often inadvertent [27]. In fact, TD is often invisible to management and executives in a business [34]. Therefore, every time they fund a new project or feature, they may be inadvertently paying interest on it, with no way of addressing the interest payment as an overall concern. A primary contribution of this study is to help make hidden TD more accessible. The typology presented in Section 5 provides a discursive framework for practitioners and academics to use as they collectively grapple with how TD can affect organisations and beyond. It details the eight categories of impact TD has on a business, namely Financial, Customer, Growth, Strategic, Internal Process, Economic, Social and Environmental.

While this paper is based on an extensive thematic review of over 100 relevant papers from the Computer Science and Information Systems domains, there may be justification for undertaking a more extensive review of literature, particularly given the paucity of material in IS. The narrower scope of this paper could limit the validity of the conclusion. However, at the same time we believe the typology provides a foundation for further research. We recommend the need for more studies looking at TD from a non-technical perspective, potentially interviewing non-technology staff across different firms to empirically document their understanding of the impact of technology-related decisions on their operations, tactics and strategies. Research providing case studies of issues resulting from TD would be an interesting addition to the existing cannon of literature. We also suggest the need for scholars to develop a taxonomy that gives businesses a '*TD scorecard*', which might allow them to measure their business health against potential technology-related risks. This could offer a practical application of our typology for practitioners.

8. References

[1] W. Cunningham, "Experience Report - The WyCash Portfolio Management System Report," in *Addendum to the proceedings on Object-oriented programming systems, languages, and applications (Addendum) (OOPSLA '92)*, 1992.

[2] E. Tom, A. Aurum, and R. Vidgen, "An exploration of technical debt," *J. Syst. Softw.*, vol. 86, no. 6, pp. 1498–1516, 2013, doi: 10.1016/j.jss.2012.12.052.

[3] Q. Huang, E. Shihab, X. Xia, D. Lo, and S. Li, "Identifying self-admitted technical debt in open source projects using text mining," pp. 418–451,

2018, doi: 10.1007/s10664-017-9522-4.
[4] E. Lim, N. Taksande, and C. Seaman, “A balancing act: What software practitioners have to say about technical debt,” *IEEE Software*. 2012, doi: 10.1109/MS.2012.130.
[5] T. Besker, A. Martini, and J. Bosch, “Managing architectural technical debt: A unified model and systematic literature review,” *J. Syst. Softw.*, vol. 135, pp. 1–16, Jan. 2018, doi: 10.1016/j.jss.2017.09.025.
[6] T. Sharma and D. Spinellis, “The Journal of Systems and Software A survey on software smells,” *J. Syst. Softw.*, vol. 138, pp. 158–173, 2018, doi: 10.1016/j.jss.2017.12.034.
[7] N. Rios, R. O. Spínola, M. Mendonça, and C. Seaman, “The most common causes and effects of technical debt: First results from a global family of industrial surveys,” in *International Symposium on Empirical Software Engineering and Measurement*, 2018, no. October, pp. 1–10, doi: 10.1145/3239235.3268917.
[8] C. J. Woodard, N. Ramasubbu, F. T. Tschang, and V. Sambamurthy, “Design capital and design moves: The logic of digital business strategy,” *MIS Q. Manag. Inf. Syst.*, vol. 37, no. 2, pp. 537–564, 2013, doi: 10.25300/MISQ/2013/37.2.10.
[9] C. Seaman *et al.*, “Using technical debt data in decision making: Potential decision approaches,” in *MTD 2012 - Proceedings*, 2012, doi: 10.1109/MTD.2012.6225999.
[10] Z. Li, P. Avgeriou, and P. Liang, “A systematic mapping study on technical debt and its management,” *J. Syst. Softw.*, vol. 101, pp. 193–220, Mar. 2015, doi: 10.1016/j.jss.2014.12.027.
[11] D. Sculley *et al.*, “Machine Learning : The High-Interest Credit Card of Technical Debt,” *NIPS 2014 Work. Softw. Eng. Mach. Learn.*, 2014, doi: 10.1007/s13398-014-0173-7.2.
[12] D. Sculley *et al.*, “Hidden technical debt in machine learning systems,” in *Advances in Neural Information Processing Systems*, 2015.
[13] F. Gillette, “The rise and inglorious fall of Myspace,” *Bloomberg Businessweek*, 2011.
[14] J. Elkington, “Cannibals with Forks: The triple bottom line of 21st century,” *Altern. Manag. Obs.*, no. April, 1997.
[15] T. Dyllick and K. Muff, “Clarifying the Meaning of Sustainable Business: Introducing a Typology From Business-as-Usual to True Business Sustainability,” *Organ. Environ.*, vol. 29, no. 2, pp. 156–174, 2016, doi: 10.1177/1086026615575176.
[16] D. Graeber, “Debt: The first five thousand years,” *Mute*, 2009.
[17] P. Vernimmen, Y. Le Fur, M. Dallochio, A. Salvi, and P. Quiry, “Bonds,” in *Corporate Finance*, 2017.
[18] S. Cecchetti, M. Mohanty, and F. Zampolli, “The Real Effects of Debt,” *BIS Work. Pap.*, 2011.
[19] K. H. Rolland, L. Mathiassen, and A. Rai, “Managing digital platforms in user organizations: The interactions between digital options and digital debt,” *Inf. Syst. Res.*, vol. 29, no. 2, pp. 419–443, 2018, doi: 10.1287/isre.2018.0788.
[20] N. S. R. Alves, T. S. Mendes, M. G. De Mendonça, R. O. Spínola, F. Shull, and C. Seaman, “Identification and management of technical debt : A systematic mapping study,” vol. 70, pp. 100–121, 2016, doi: 10.1016/j.infsof.2015.10.008.
[21] S. De Haes and W. Van Grembergen, “An exploratory study into IT governance implementations and its impact on business/IT alignment,” *Inf. Syst. Manag.*, vol. 26, no. 2, pp. 123–137, 2009.
[22] D. Dougherty, “Interpretive barriers to successful product innovation in large firms,” *Organ. Sci.*, vol. 3, no. 2, pp. 179–202, 1992.
[23] B. A. Bechky, “Sharing meaning across occupational communities: The transformation of understanding on a production floor,” *Organ. Sci.*, vol. 14, no. 3, pp. 312–330, 2003.
[24] R. A. Crowson, *Classification and Biology*. Routledge, 2017.
[25] D. H. Doty and W. H. Glick, “Typologies As a Unique Form Of Theory Building: Toward Improved Understanding and Modeling,” *Acad. Manag. Rev.*, vol. 19, no. 2, pp. 230–251, Apr. 1994, doi: 10.5465/amr.1994.9410210748.
[26] P. O’Raghallaigh, D. Sammon, and C. Murphy, “Theory-building using Typologies - A Worked Example of Building a Typology of Knowledge Activities for Innovation,” in *DSS*, 2010.
[27] N. Brown *et al.*, “Managing technical debt in software-reliant systems,” in *Proceedings of the FSE/SDP Workshop on the Future of Software Engineering Research, FoSER 2010*, 2010, doi: 10.1145/1882362.1882373.
[28] P. Kruchten, R. L. Nord, and I. Ozkaya, “Technical debt: From metaphor to theory and practice,” *IEEE Software*. 2012, doi: 10.1109/MS.2012.167.
[29] R. L. Nord, I. Ozkaya, P. Kruchten, and M. Gonzalez-Rojas, “In Search of a Metric for Managing Architectural Technical Debt,” in *WICSA/ECSA 2012*, Aug. 2012, pp. 91–100, doi: 10.1109/WICSA-ECSA.212.17.
[30] J. Yli-Huumo, A. Maglyas, and K. Smolander, “How do software development teams manage technical debt? – An empirical study,” *J. Syst. Softw.*, vol. 120, pp. 195–218, 2016, doi: 10.1016/j.jss.2016.05.018.
[31] W. N. Behutiye, P. Rodríguez, M. Oivo, and A. Tosun, “Analyzing the concept of technical debt in the context of agile software development: A systematic literature review,” *Inf. Softw. Technol.*, vol. 82, pp. 139–158, 2017, doi: 10.1016/j.infsof.2016.10.004.
[32] A. Martini, T. Besker, and J. Bosch, “Technical Debt tracking: Current state of practice,” *Sci. Comput. Program.*, vol. 163, pp. 42–61, Oct. 2018, doi: 10.1016/j.scico.2018.03.007.

[33] T. Besker, A. Martini, and J. Bosch, “Managing architectural technical debt: A unified model and systematic literature review,” *J. Syst. Softw.*, vol. 135, 2018, doi: 10.1016/j.jss.2017.09.025.
[34] N. A. Ernst, S. Bellomo, I. Ozkaya, R. L. Nord, and I. Gorton, “Measure it? Manage it? Ignore it? software practitioners and technical debt,” in *ESEC/FSE 2015*, Aug. 2015, pp. 50–60, doi: 10.1145/2786805.2786848.
[35] V. Lenarduzzi, T. Besker, D. Taibi, A. Martini, and F. Arcelli Fontana, “A systematic literature review on Technical Debt prioritization: Strategies, processes, factors, and tools,” *J. Syst. Softw.*, vol. 171, p. 110827, Jan. 2021, doi: 10.1016/j.jss.2020.110827.
[36] J. Holvitie *et al.*, “Technical debt and agile software development practices and processes: An industry practitioner survey,” *Inf. Softw. Technol.*, vol. 96, no. November 2017, pp. 141–160, Apr. 2018, doi: 10.1016/j.infsof.2017.11.015.
[37] S. McConnell, “Managing Technical Debt.” https://www.youtube.com/watch?v=lEKvzEyNtbk.
[38] W. Cunningham, “Debt Metaphor,” *YouTube*, 2009. https://www.youtube.com/watch?v=pqeJFYwnkjE (accessed Jun. 14, 2021).
[39] A. Martini and J. Bosch, “The Danger of Architectural Technical Debt: Contagious Debt and Vicious Circles,” *Proc. - 12th Work. IEEE/IFIP Conf. Softw. Archit. WICSA 2015*, pp. 1–10, 2015, doi: 10.1109/WICSA.2015.31.
[40] A. Nugroho, J. Visser, and T. Kuipers, “An empirical model of technical debt and interest,” in *Proceedings - International Conference on Software Engineering*, 2011, doi: 10.1145/1985362.1985364.
[41] A. Potdar, “An Exploratory Study on Self-Admitted Technical Debt,” 2014, doi: 10.1109/ICSME.2014.31.
[42] G. Digkas, M. Lungu, P. Avgeriou, A. Chatzigeorgiou, and A. Ampatzoglou, “How Do Developers Fix Issues and Pay Back Technical Debt in the Apache Ecosystem ?,” pp. 153–163, 2018.
[43] R. S. Kaplan and D. P. Norton, “Using the Balanced Scorecard as a Strategic Nanagement System,” *Harv. Bus. Rev.*, 1996.
[44] W. E. Stewart, “Balanced scorecard for projects,” *Proj. Manag. J.*, vol. 32(1), pp. 38–53, 2001.
[45] M. Martinsons, R. Davison, and D. Tse, “The balanced scorecard: a foundation for the strategic management of information systems,” *Decis. Support Syst.*, vol. 25, no. 1, pp. 71–88, Feb. 1999, doi: 10.1016/S0167-9236(98)00086-4.
[46] C. Gralha, N. Lincs, D. Damian, A. I. T. Wasserman, and M. Goul, “The Evolution of Requirements Practices in Software Startups,” pp. 823–833, 2018, doi: 10.1145/3180155.3180158.
[47] US Securities and Exchange Commission, “SEC Charges Knight Capital With Violations of Market Access Rule,” 2013.
[48] G. Spanos and L. Angelis, “The impact of information security events to the stock market: A systematic literature review,” *Computers and Security*. 2016, doi: 10.1016/j.cose.2015.12.006.
[49] C. Seaman and Y. Guo, *Measuring and Monitoring Technical Debt*. 2011.
[50] P. Constantinides, O. Henfridsson, and G. G. Parker, *Platforms and infrastructures in the digital age*, vol. 29, no. 2. pubsonline.informs.org, 2018.
[51] J. Yli-Huumo, A. Maglyas, and K. Smolander, “The Sources and Approaches to Management of Technical Debt: A Case Study of Two Product Lines in a Middle-Size Finnish Software Company,” in *Lecture Notes in Computer Science*, vol. 8892, no. June 2016, 2014, pp. 93–107.
[52] M. Dutt, “Sustainable investment: A new landscape,” *OECD Obs.*, May 2020, doi: 10.1787/7cd0e90e-en.
[53] R. S. Kaplan and D. McMillan, “Updating the Balanced Scorecard for Triple Bottom Line Strategies,” *SSRN Electron. J.*, 2020, doi: 10.2139/ssrn.3682788.
[54] J. Sandebrg, J. Holmstrom, and K. Lyytinen, “Digitization and Phase Transitions in Platform Organizing Logics: Evidence from the Process Automation Industry,” *MIS Q.*, vol. 44, no. 1, pp. 129–153, Jan. 2020, doi: 10.25300/MISQ/2020/14520.
[55] J. Slay and M. Miller, “Lessons Learned from the Maroochy Water Breach,” in *Critical Infrastructure Protection*, 2008, pp. 73–82.
[56] B. Mo, Yilin and Weerakkody, Sean and Sinopoli, “Physical Authentication of Control Systems: Designing Watermarked Control Inputs to Detect Counterfeit Sensor Outputs,” *IEEE Control Syst.*, vol. 35, no. 1, pp. 93–109, Feb. 2015, doi: 10.1109/MCS.2014.2364724.
[57] R. M. Atlas and T. C. Hazen, “Oil Biodegradation and Bioremediation: A Tale of the Two Worst Spills in U.S. History,” *Environ. Sci. Technol.*, vol. 45, no. 16, pp. 6709–6715, Aug. 2011, doi: 10.1021/es2013227.
[58] I. Žliobaitė, “Measuring discrimination in algorithmic decision making,” *Data Min. Knowl. Discov.*, 2017, doi: 10.1007/s10618-017-0506-1.
[59] D. Kuss and M. Griffiths, “Social Networking Sites and Addiction: Ten Lessons Learned,” *Int. J. Environ. Res. Public Health*, vol. 14, no. 3, p. 311, Mar. 2017, doi: 10.3390/ijerph14030311.
[60] M. Conway, “Determining the Role of the Internet in Violent Extremism and Terrorism: Six Suggestions for Progressing Research,” *Stud. Confl. Terror.*, vol. 40, no. 1, pp. 77–98, Jan. 2017, doi: 10.1080/1057610X.2016.1157408.
[61] N. Grinberg, K. Joseph, L. Friedland, B. Swire-Thompson, and D. Lazer, “Fake news on Twitter during the 2016 U.S. presidential election,” *Science (80-.).*, vol. 363, no. 6425, pp. 374–378, Jan. 2019, doi: 10.1126/science.aau2706.