IT Portfolio management as a framework for managing Technical Debt

Theoretical framework applied on a case study

Mille Edith Nielsen
IT University of Copenhagen, Research Centre for Government IT, miln@itu.dk

Søren Skaarup
IT University of Copenhagen, Research Centre for Government IT
skaa@itu.dk

ABSTRACT

Technical debt (TD) originally names the reoccurring phenomenon of shortcuts and quick fixes within IT development. TD saves time and resources in the short run but may cause problems and require additional resources in the long run. Managing TD is difficult, whether it is seen as a local phenomenon (residing in individual applications) or seen in a portfolio perspective. However, TD is not always caused by the individual developer and it does not always just affect one application. It may affect several parts of the IT portfolio; it may arise in the portfolio and it may be resolved by decisions on an IT-portfolio level. In this paper, we develop a theoretical framework that incorporates TD into IT portfolio management (ITPM). We apply the framework on a case study to explore how TD can be created, reside, resolved and managed in practice in a portfolio-perspective. This paper contributes with a framework integrating TD with ITPM. The paper also provides empirical insights on practice regarding ITPM and TD and highlights implications for research and practice.

CCS CONCEPTS

• **Applied Computing – E-government**; • **IT Portfolio management**; • **Technical debt management**;

KEYWORDS

Technical debt, Technical debt management, IT portfolio management, IT portfolio framework, IT governance

ACM Reference Format:
Mille Edith Nielsen and Søren Skaarup. 2021. IT Portfolio management as a framework for managing Technical Debt: Theoretical framework applied on a case study. In *14th International Conference on Theory and Practice of Electronic Governance (ICEGOV 2021), October 06–08, 2021, Athens, Greece.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3494193.3494296

ICEGOV 2021, October 06–08, 2021, Athens, Greece

ACM ISBN 978-1-4503-9011-8/21/10…$15.00
https://doi.org/10.1145/3494193.3494296

1 INTRODUCTION

IT development often takes place under a strict budget or timeframe, and the scope and the resources available may change during development. These restrictions and changes can lead to short-cuts or suboptimal solutions in order to save resources and deliver on time. Cunningham called this phenomenon "Technical Debt" (TD) [7]. TD has traditionally been understood as a short-term solution which needs to be addressed at later point in time to ensure software stability.

TD can be beneficial in the short run, however, if the debt is not managed it may pose a risk in the long run [7]. TD can have negative effects on IT maintenance, as it can become more time consuming to resolve the debt or develop the IT application further. TD can slow IT applications down and cause breakdowns, and it can bring entire organizations to a stand-still [7, 27]. The costs and constraints introduced by IT can also be an impediment for business-development and growth [7, 23]. TD is often seen as an aspect of individual systems [22] arising from choices made during system-development [7, 22]. However, TD may also be created during IT maintenance [32] and may evolve (increase or decrease) during the application lifecycle [25].

TD has primarily been researched in the context of the private sector and in the software engineering field, thus leaving a gap for TD research in the public sector and from an eGovernment perspective [22]. TD is a serious issue in the public sector. A third of societal or business-critical IT systems in the Danish Government have been found to be in an inadequate condition [8]. Likewise, the Swedish National Audit Office found that 70% of their IT systems were outdated [34]. This shows the importance of understanding the creation and management of TD in the public sector. While the private and the public sector shares a constant drive to increase efficiency and effectiveness, to a large extent through the use of technology, there are also important differences in what drives change in the two sectors [6]. New political initiatives and priorities may require new development or changes in IT-systems within a short timeframe, and this may induce shortcuts and lead to the creation of TD in the systems affected and/or in other systems from where resources may be taken.

The research questions we seek to answer then are: *How can technical debt be conceptualized in an IT- portfolio-management perspective? And how is technical debt created and managed in the IT portfolio perspective in a government organization?* To answer this, we propose a theoretical framework combining IT portfolio management (ITPM) and TD management, and we apply this framework to a public sector case study to explore how TD is created

and governed at a portfolio level in practice. This study makes a contribution to the e-Government literature by offering a tool for understanding TD management in an ITPM context, and by providing empirical insights into TD management practices in a government organization. The framework is also useful for IT-practitioners for increasing awareness of TD challenges in the IT portfolio. The study thus bridges and has implications for both practice and research and advances digital governance research and practice agenda [36].

The paper is structured as follows: First, we present related work on TD management, secondly, we synthesize the literature on IT portfolio management and present a theoretical framework. We then introduce our case study, and our methodological approach, before applying the framework on the case study. Finally, we discuss the application of the framework and the relation between IT portfolio and TD.

2 RELATED WORK: TECHNICAL DEBT MANAGEMENT

The term "technical debt" (TD) was coined by Cunningham in 1992 [7]. Since then, TD research has focused on further defining the concept and develop tools to identify and quantify TD items. This research has generally confirmed the negative effects of TD [22]. Rios et al [27] conducted a tertiary literature study on TD examining 13 secondary TD studies and 185 TD primary studies. Therefore, in our conceptualization of TD we include problems deriving from incomplete documentation, not following the IT-architecture, and from short term solutions which are not sustainable in the long run. To limit the consequences of the negative effects of TD, it is crucial to focus on the management of TD. Griffith, Taffahi, Izurieta, and Claudio's [4, p. 1016] define TD management as comprising "the actions of identification, assessment, and remediation of TD throughout a software system." From this definition TD management can appear to be limited to cover one software system at a time.

Avgeriou, Ernst, Nord and Kutchen [1, p. 40] find that ". . . [what] the software technical-debt community has not discussed is that of technical debt crossing multiple disciplines. In a sense, technical debt is incurred in one discipline, but it is burdened and has to be repaid in another discipline." For example management may make decisions that accumulate TD in applications, which then become the developers' task to solve [1].

Rios et al. [28] conducted a survey of 107 practitioners from the software industry to identify the experienced causes of TD. They identified the most frequent causes as: deadlines, inappropriate planning, lack of knowledge, non-adoption of good practices, ineffective project management, lack of qualified professionals, lack of experience, outdated/incomplete documentation, and lack of commitment. These causes are not restricted to the individual developer or application and includes IT-management and the management of the whole IT-portfolio among the sources of TD. They also indicate that TD may arise from the wider organization, e.g., when deadlines and resource-allocations are decided there.

Klinger et al [14] examines TD from an enterprise perspective, they find that "*enterprise technical debt occurs in the context of a larger portfolio*". For example, different non-technical stakeholders in the portfolio have an impact on TD, however, it is difficult for non-technical stakeholders to understand TD and their effect on it. [14].

TD can be introduced by changes outside the IT-development and -maintenance activities [1]. It may, for example, arise from the relation between IT applications and the infrastructure on which they are built and run. E.g., if the IT application requires a specific infrastructure which is no longer supported, or there is an increased use of the IT application so it exceeds what the infrastructure can deliver (known as infrastructure debt [27]). TD may also arise from changes in the business context of the organization or in the technological environment in which it operates [26]. In the business environment, several factors may affect TD, such as, the business's need for high system-reliability, user/customer satisfaction [26].

With their concept of "technological debt", Magnusson & Bygstad [19] add further dimensions to this growing set of factors that can influence the accumulation of TD. Applying institutional theory, they point to the factors of path-dependency (that previous decisions and established practices and principles may guide future decisions), lock-in (when a technology is embedded in standards, contracts, IT-staff and user-skills and in systems-integrations, making it more difficult to switch), and institutional logics (e.g., a focus on cost savings and standardizations). All these factors, while not strictly rooted in technical considerations, may influence decisions made about and around the systems.

Thus, a wide set of factors may influence TD, as TD occurs in the context of a larger portfolio [2]. There is, as indicated by Klinger [14], Rios et al [27] and by Magnusson and Bygstad [19], a need for a more holistic approach to TD and how to manage it, something which appears to be missing in the literature. Therefore, we find it relevant to view TD in a portfolio perspective, and to view the management of TD as part of the overall IT-portfolio management in order to benefit from collective resources within the organization.

3 METHODOLOGICAL APPROACH

This section presents the methodological approach of the paper. First, we describe how the framework was synthesized, then we describe the selection of the case and the case itself. Lastly, we present the data gathering techniques we applied.

3.1 Framework

We contextualize TD management in an ITPM framework based on a reading of IT Portfolio literature [12]. The result is a framework consisting of the core elements of the IT-portfolio, and the core management activities across the portfolio, including the management of TD. Thus, the framework could be extended with further portfolio elements and further activities. The purpose of applying a framework is to have it as a guidance for exploring, explaining and interpreting the empirical material [12]. A framework can be used as a mirror to the findings and the findings can be used to improve and adjust the framework and show its possible limitations. A theoretical framework draws on theory or concepts from a theory, to shed light on a particular phenomenon or research problem [12]. Therefore, the researcher first chooses the theory they want to guide the analysis and then they synthesize it into a framework.

We choose to review the literature on IT portfolio and ITPM as presented in section 4. We develop a theoretical framework in which we apply a deductive approach in synthesizing the literature on ITPM. Secondly, we apply the framework on the empirical material (see section 5.1).

3.2 Case study

We conducted a case study [37] to explore the relation between IT portfolio and TD and between ITPM and TD management. The case study research approach is recommended when the researcher wishes to explain "how" or "why" a social phenomenon works [37], as it enables a detailed understanding of a phenomenon [9]. We have chosen a deviant case to obtain knowledge on an especially good practice [8], as this allowed us to observe good ITPM and how TD management is performed in the organization.

Agency X is a mature IT organization and considered among the best government organizations in Denmark in the development and operation of IT-systems. The agency has a large IT-portfolio containing internally developed as well as off the shelf applications. The agency has not employed their own developers, instead they have partnerships with software companies. The IT-development is caried out in a collaboration between the agency and software companies' developers placed onsite. Agency X works actively with ITPM; there are monthly meetings between the IT architects and top management, and a quarterly two-day workshop with the IT department, top management, and business managers. The latter is inspired by SAFe's[1] Program Increment (PI) planning, with a focus on planning of IT development and the interdependencies between projects and maintenance [30]. The purpose of these quarterly workshops is to plan the next couple of months of IT development and maintenance and to ensure dependencies between tasks, projects, and systems are taken into account and to mitigate any potential risks. The business managers are present in these workshops to help prioritize tasks, clarify needs and answer funding questions. These workshops primarily focus on internally developed systems. The PI planning format was implemented shortly before to the data collection toke place and the agency was working toward a final form.

Generalizing case studies can be problematic [37], although Flyvbjerg argues it is possible to generalize on the basis on a single case [8]. Our case selection is information-oriented and we have selected an extreme case, more precisely a good case [8]. in Agency X, which is considered among the most mature public sector IT-organizations in Denmark. We would expect that we in this case would be able to find TD managed at portfolio level.

3.3 Techniques for data collection

Data was collected applying several qualitative techniques: participatory observation, semi-structured interviews, and document analysis. The data collection took place from August 2019 to December 2019.

Participant observation [5] was carried out by the first author who was present at Agency X three full days a week during the data collection period. Additionally, the first author attended three of the quarterly PI workshops, in which the following months of IT development were presented and prioritized, and dependencies coordinated. This provided a rich insight into how ITPM took place. Conducting observations allows the researcher to address the say/do problem, where people may say one thing during interviews and do something else in practice [3]. These observations were captured through continuous notetaking.

Next, the first author conducted 15 semi-structured interviews [16] and two in situ interviews [4]. During the in situ interviews the interviewee performed their tasks while the interviewer asked about their practice. The interviews consisted of open-ended questions framed around the employee's collaboration process with the other employees maintaining and managing the IT system. From the interviews, we gained insights into the employees' perception and reflection of the maintenance processes and ITPM.

Together these techniques provide an in depth and detailed empirical insights. Particular, the participatory observation was beneficial in gathering rich insights. The observation notes were coded, together with the transcribed interviews. The empirical material was analyzed by applying the theoretical framework (section 4).

4 FRAMEWORK: IT PORTFOLIO MANAGEMENT

In this section we synthesize the IT-portfolio literature to establish a theoretical framework (figure 1) which can be used to investigate the creation and solution of TD. In this framework we propose to include TD management as a distinct portfolio-management activity. This framework will then be applied to an empirical case (section 5).

A "portfolio" can be defined as "A collection of items grouped together to facilitate their efficient and effective management" [23, p. 37]. Typically, a portfolio is comprised of a group of business-investments with something in common, e.g., contributing to the same overall organizational goal(s). A portfolio is a dynamic entity which has to be continuously managed, monitored and maintained as a set of interrelated assets/activities which address the business needs efficiently [13]. IT-portfolio management (ITPM) includes the management of applications, infrastructure, projects and people and their mutual dependencies [15, 33]. Additionally, Verhoef [35] argues that IT-operations and maintenance are aspects of an IT portfolio.

The main goal of managing the IT-portfolio is to ensure that it supports the activities and goals of the business [18, 20]. Thus, a key activity across the portfolio is to ensure that it is well aligned with business needs and goals. Changes in business-needs may be driven by changes in the business environment where, in the case of public sector organizations the political environment plays an important part [6, 21].

Another key activity across the portfolio is to manage personnel-resources in a global perspective, which may be a challenging task [21]. Closely related to this is deciding what parts of the portfolio to operate, maintain and develop inhouse and what parts of these tasks to buy from outside vendors, including which applications to buy and which to develop internally [31]. These decisions are not only a matter of economics but are closely related with considerations of what resources and skills are, can and should be available in-house

[1]The Scaled Agile Framework®(SAFe®) is a system for implementing agile practices at enterprise scale it includes structured guidance on roles and responsibilities [30]

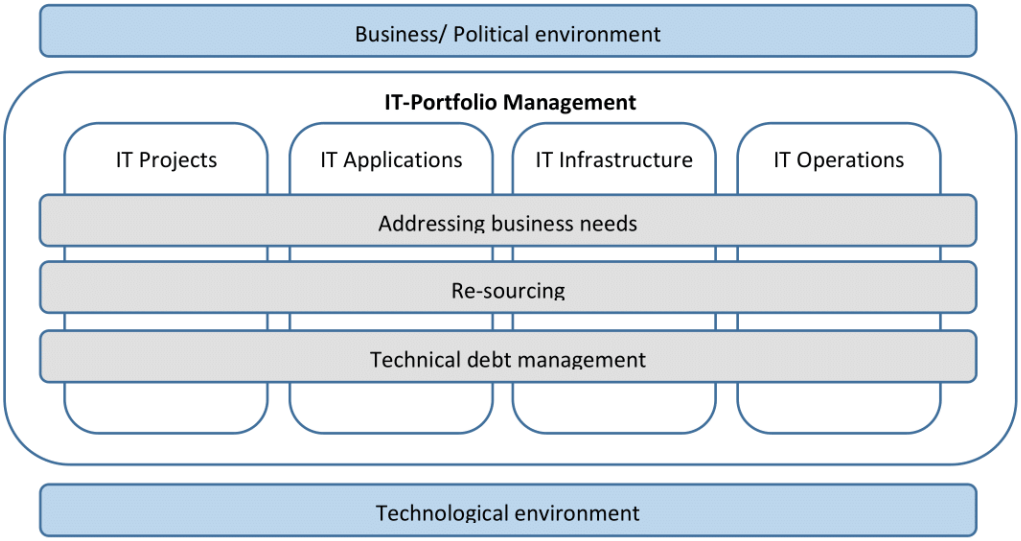

Figure 1: Theoretical framework of the components of an IT-portfolio (white boxes), key management activities across the portfolio (grey boxes) and the most important external factors affecting the portfolio (blue boxes)

[29]. Thus, the management of "people" across the portfolio and the management of sourcing/procurement is closely related [29]. For the purpose of this framework, we have combined the two in what we call the re-sourcing activity.

The purpose of ITPM is to apply a holistic perspective on these sub-portfolios as one interconnected portfolio [10, 15, 24]. And based on our practical experience they are indeed interconnected. IT-applications are either bought or developed inhouse through IT-projects – a re-sourcing decision. The applications run on infrastructure that is either bought or operated in-house (a re-sourcing decision) and applications and infrastructure is maintained by IT-operations (which again can be outsourced or in-house)[2]. Changes in applications and infrastructures are (ideally) primarily driven by business needs which may in turn be driven by changes in the business environment, but changes may also be driven by changes in the external technology-environment [31]. In the technological environment, new technologies may make it possible to discard old systems, including the debt they may have accrued [26] or new technologies may reveal inadequacies in the existing technological setup [2].

As indicated by the discussion in the related work section, TD comes in many forms, and may be created in all parts of this portfolio and in the interplay between its elements [14, 19, 27]. Therefore, we find it relevant to include TD management in the ITPM framework. We combine these elements and present them in a theoretical framework as displayed in figure 1

5 FINDINGS

We apply the framework (figure 1) on the case of agency X to investigate if and how TD is created in the portfolio and to what extent the agency managed TD as part of their ITPM. Through our application of the framework, we find several examples of portfolio-related TD. Here, we present these examples and relate them to the framework. ITPM decisions lead to the creation of TD within the portfolio, however, portfolio related decisions also resolve some of the TD. Also, agency X have implemented tools and procedures to prevent and resolve TD on an IT portfolio level. We present five overarching findings: 1) TD exist in the IT portfolio, 2) TD may not always be created and reside in the application it is handled, 3) re-sourcing decisions impact and reveal TD creation, 4) TD is created by the change of TD needs, 5) successful TD management strategies which are implemented in Agency X.

5.1 TD resides, not in one application but in the application portfolio.

An older internally developed Content Management System (CMS) only partly supported the need for CMS functionalities that had been developed in Agency X. Therefore, several other CMS applications had been added to the application portfolio supporting different needs but with overlapping functionalities. This created unnecessary complexity in the application portfolio which constitutes a type of TD. This TD does not reside in the individual CMS applications but in the application-portfolio. The Agency had to decide whether they should continue with all of these CMS applications or if they should replace them with one off the shelf application. This would eliminate the TD arising from the complexity of having many CMS systems. Resolving the TD in this way would have re-sourcing effects across the portfolio. Operations would have to develop new integrations for the different business-applications to the new system, and to take care of the decommissioning of the old systems. In addition, if the new systems

[2]This inhouse/outsourced distinction is a considerable simplification of a much more nuanced set of options [17]

required different server-resources than the old systems, changes would have to be made in the infrastructure portfolio.

5.2 TD may not always be created and reside in the application where it is handled

TD in an application can be created by the way that application is used by other applications. A case handling component served several business applications. The teams developing the business applications did not always comply with the data formats required by the case-handling component due to lack of knowledge or lack of resources. This created performance problems for the case-handling component. The TD was in this case created and also resided in the business-applications, but the issues were seen as problems in the case-handling component. Therefore, it was left to the developers maintaining that application to fix the problems that arose by swiftly changing their priorities in order to clean up the data before it affected the citizens who relied on that data. TD created in business applications to support business needs created problems for operations because they had to change focus and prevent this TD from affecting citizens (another business need).

5.3 Re-sourcing decisions in one part of the portfolio creates or exacerbates TD in another part

A change was needed in a self-service application to improve the user-experience, however, needs in other parts of the portfolio resulted in the application being deprioritized and striped of its developers on several occasions. Every time a new developer came onboard, they had to be trained to work on the application. The lack of developers with knowledge of the application constituted a type of TD which was exacerbated by re-sourcing decisions, depriving the application of resources and making the change more costly and time-consuming than originally planned.

Another example is the case handling component mentioned in section 5.2. It was stripped of developer resources several times, as they were needed elsewhere. Shortly after these resources where removed errors started occurring in the system. As this affected both internal users and citizens, finding a solution became a priority. Once the component was prioritized and had developers assigned to it, problems where again addressed quickly, and the errors stopped.

These examples of TD were not created in the application or in the application portfolio but in the re-sourcing activities across the whole IT-portfolio. However, this was not a result of any overall prioritization of resources, but of ad-hoc decisions which focused on where the resources where needed, not on the consequences of moving them.

5.4 TD created by re-sourcing decisions and then transferred to operations

The case handling system was belatedly transferred from development to operations. It turned out that this transfer included a backlog of more than six hundred unresolved tasks of varying severity, partly with unknown status. This backlog constituted a significant amount of TD and had accumulated because the resources had been moved from the development-project too soon. It was now up to operations to clean up this backlog, but they were not given resources to do this. This example illustrates a tension in the re-sourcing going from project to operation.

5.5 Re-sourcing decisions reveal existing but unknown technical debt in the infrastructure

Re-sourcing-decisions may also reveal existing but unknown TD. An operations' provider decided to change the hosting of servers from a subcontractor to themselves. This change in hosting did not "create" TD as much as it revealed and actualized TD that existed but was not known; logic that should have been implemented at the application level and thus should not have been affected by the move turned out to be implemented elsewhere and would have to be re-developed for the new setup. Several applications turned out to be running older versions that would have to be updated and there was a lack of documentation which made it difficult and more costly to perform the move. This TD did not reside in individual applications but in the infrastructure setup.

5.6 Technical debt created by changes in business needs

Agency X is a key implementer of high prioritized government policies and also has a long tradition of being very sensitive to the (changing) needs of the citizens they serve. This drives change, sometimes rapid change, in the business needs and goals the IT-portfolio is expected to support.

5.6.1 Technical debt created by assumed business-needs. The old CMS application which we mentioned in section 5.1, was created to support an assumed business need of supporting multiple language versions of the texts it contained as well as a detailed version-history of the texts; however, these functionalities were never actually fully utilized. The consequence was an unnecessarily complicated code-based which was more difficult for developers to navigate and to perform maintenance and further developments on. This TD was not created by shortcuts in the development process but by business needs that never fully materialized. Had the business fully utilized this functionality, the relative complexity of the code would probably not have been considered problematic and there would be no TD here.

5.6.2 Technical debt created by changes in political priorities. Changes in the business environment may create or reveal TD in the IT-portfolio. In a political environment such as the one in which agency X operates, changes in political objectives and the need for fast implementation of new regulations can create shorten deadlines and lead to cutting corners and the creation of new TD. Sudden changes in priorities can lead the organization to postpone or give up on ongoing changes, which would have reduced TD, in order to free up resources to handle the new top priorities.

When a general election was coming up, Agency X had to take a possible change in government into account in their planning as it could change the political priorities in the areas where agency X operate. This in turn could have significant consequences for how resources where prioritized across the IT-portfolio. While some types of new regulations were created through lengthy process

that allowed the agency ample time to plan and prioritize, other types appeared suddenly and unexpectedly e.g., as the result of late-night political compromises. This could mean that the entire IT-planning had to be redone. Not only could this lead to resources being shifted away from ongoing projects or operation, but also to this being done with such haste, that there was no time to document or consider the consequences. Not only could new TD be created but also new unknown TD. In addition, the deadline for the implementation of the new regulations, and consequently for the development and implementation of new IT, could be so tight, that development would have to begin before the details of the new regulation was finalized. Not only could this lead to resources being wasted because things would have to be redone, but this deadline could also lead to the creation of TD. These examples refer to addressing business needs (in form of the politicians), resourcing, IT projects and possibly operations.

5.7 Managing technical debt

The examples above show how TD was created in the IT-portfolio of agency X in a number of ways. However, we also observed how the agency was beginning to handle TD in a more systematic way and in a portfolio-perspective.

5.7.1 Bundling technical debt into projects to be prioritized with other projects. Agency X has a focus on managing TD. The head of the IT department encourage application managers to bundle TD tasks and make a small business case so these tasks can be addressed and prioritized alongside with IT development projects. One example was a project that arose from a TD problem, which affected several parts of the organization. The project solved the problem and was prioritized and funded like other IT projects. Thus, TD management appears to be in the process of becoming an IT portfolio activity as suggested by the framework. In this example, the activity covers IT projects and applications as the TD debt in the applications is reduced in the project.

5.7.2 Integrating technical debt resolution into development projects. We also saw examples of how Agency X resolved old TD during IT development. One example was in the development of new functionality for an existing system. The resolution of a list of older TD items in the system was integrated into the development project as a separate task and prioritized at the same level as the development of new functionality. Thus, the TD management of an application is addressed in the project.

5.7.3 Towards portfolio-focused re-sourcing planning. As we have seen, TD may be created in one part of the portfolio by resourcing decisions made in other parts. A continuous overview and management of resources used in projects and on maintenance across the portfolio can contribute to reduce existing TD as well as the risk of creating new TD. The overview facilitates the allocation of resources to new tasks in a timely manner, and it may to some extent prevent the striping of resources from unfinished tasks, or at least do this in a way where the consequences are considered, and any resulting TD is catalogued for later resolution.

The beginnings of such a holistic portfolio-wide planning process was seen in agency X in the shape of the Program Increment (PI) planning process. The PI planning is a reoccurring event in the SAFe framework, adapted to Agency X's needs [30].

In the PI planning sessions, a "dependency board" displayed the dependencies between the project-, maintenance- and operation-teams over time. Through this visualization of the dependencies, it would for example become apparent that within the same short time period several projects relied on the same application-team to deliver on some tasks. This realization caused the projects to reschedule and postpone these tasks, thus, freeing up the application-team to handle one dependency at the time. The potential consequences of not rescheduling could be not meeting deadlines, having to reschedule much later or creating TD. Rescheduling is not an easy task as a dependency could reach further than one step, it could be multiple steps through various actors. The dependency board illustrated the complexity and final goal of this trail of dependent tasks, also illustrating that scheduling the dependencies is important and they can be complex. This example demonstrates how PI planning involves resource planning (of projects and operations). In effect it also involves TD management because resource planning can prevent TD.

The organizers of the PI planning process stressed the importance of business owners being present in the planning sessions, to facilitate the resolution of any doubts that might arise regarding funding and priorities. The planning process also included the employees' own assessments of their workload and how thinly stretched they were, in order to catch any potential problems before they occurred. One possible shortcoming in the PI planning process could be that while it includes a resource overview of the internal employees it does not include external resources, which means that it does not include any development resources as these are all external in agency X. Prioritization of these resources are made elsewhere.

6 SUMMARY

This case shows that even in a large IT-mature organization as Agency X, TD can be created unwittingly. This TD can result from more than code related decisions made by individual developers in individual applications and it can reside in the IT-portfolio rather than in individual applications. However, resolving this TD may still end up as a task for an application- or operations-team, even though it may be caused by another project, a different application, or decisions regarding, business needs, sourcing or infrastructure in other parts of the portfolio. Unfortunately, even if the application team or operations are aware of this TD, they rarely have the necessary resources to solve this debt on top of their other tasks. The political stakeholders provide another level of complexity in planning IT development, an unavoidable factor which political sensitive organizations such as Agency X has to include in their planning.

Agency X have already implemented several methods to address TD continuously and in an IT portfolio perspective. However, TD will never be truly eliminated nor should it, but it is important to recognize it, and its potential consequences, when it is created. Thus, the methods Agency X already has implemented are a crucial step to prevent and resolve TD. The methods include: 1) PI planning, 2) encouraging application managers to bundle TD tasks so they can

be prioritized along with IT projects, and 3) including TD resolution in IT projects working on existing applications. Additionally, we encourage to adopt a portfolio focused approach to re-sourcing planning.

7 DISCUSSION, CONCLUSION AND SUGGESTIONS FOR FUTURE WORK

In this paper, we explore how TD (management) can be conceptualized in an ITPM perspective, and how TD is created and managed in the IT portfolio perspective in a government organization. We synthesize the IT portfolio literature and propose a framework including the TD management activity, and we examine how TD is created and managed on a portfolio level in Agency X, a Danish government agency.

TD has primarily been researched in the private sector and focusing on the debt created in individual projects or applications [22]. Bygstad & Magnusson [19] and Klinger et al. [14] expand the TD metaphor to include a larger part of IT development.

We contribute with a theoretical framework which incorporates TD management into an IT portfolio and IT portfolio management (ITPM) context. The framework stimulates a different way of thinking of TD and how to manage it at a portfolio level. In addition, the framework can serve as a communication tool for addressing and communicating problem as described by Klinger et al. [14]. The framework only covers the core-elements and activities of an IT-portfolio and of ITPM and we suggest that future studies apply and further develop the framework. We confirm that it can be productive to see TD in a larger and more holistic perspective [14, 19]. TD can be created by and in an IT portfolio and not only in source-code and within IT projects and that a narrow view on resource-planning in the IT-portfolio affects TD creation and management. Thus, we argue that it can be beneficial to integrate TD and TD management into ITPM.

Organizations can benefit from expanding their view of TD from something which is created and dealt with in the respective applications through development projects and primarily resides in the source-code, to something which is also created and live in the IT portfolio. They could also benefit from a holistic portfolio-wide approach to resource planning and from considering the full consequences of ad-hoc changes in resource allocations. This can enhance management's understanding of how TD emerges at an IT portfolio level, for example caused by short deadlines introduced by changes in political priorities which introduces new business-requirements to the portfolio. Among the benefits could be improved planning, more informed decisions when prioritizing resources in the IT portfolio, as well as improved productivity and quality. Agency X exemplify such ITPM activities in their PI planning, as well as when they bundle TD tasks into TD-projects so they can prioritize along with other projects and when they include TD resolution during IT development of existing applications. Better TD management could also be achieved by careful documentation of TD that may be created when withdrawing resources from a project or by extending deadlines when the resources are partly removed. Or by addressing TD during IT-development projects, rather than "exporting" TD-issues to maintenance and operations which rarely has the resources to deal with it.

The analysis of the case shows that the framework developed here can be useful for understanding TD in a portfolio perspective. However, the framework does not, by design, include all possible sub-portfolios of the overall IT-portfolio, nor all possible ITMP activities, only those that were considered the most important in the literature. A broader set of sub-portfolios and activities could possibly help in identifying even more portfolio-related sources of TD and provided a more detailed understanding of the sources and remedies identified in the empirical case. This we leave for future studies.

ACKNOWLEDGMENTS

This study is funded by a Research Centre Government IT, which is a collaboration between the IT University of Copenhagen, the Danish Digitization Agency, and the self-owned institution ATP. The Research Centre's external funding partners were not involved in the research presented herein, or its dissemination. Furthermore, while Agency X has fact checked the article, the Agency has not funded the article nor part taken in analyzing or writing the article.

REFERENCES

[1] Paris Avgeriou, Neil A. Ernst, Robert L. Nord, and Philippe Kruchten. 2016. Technical Debt. ACM SIGSOFT Software Engineering Notes 41, 2: 38–41. https://doi.org/10.1145/2894784.2894800
[2] B. Bahli and S. Rivard. 2003. A validation of measures associated with the risk factors in technology outsourcing. Proceedings of the 36th Annual Hawaii International Conference on System Sciences, HICSS 2003. https://doi.org/10.1109/HICSS.2003.1174792
[3] Jeanette Blomberg and Mark Burrell. 2012. An Ethnographic Approach to Design. In Human–Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications (3rd ed.), J. A. Jacko (ed.). Boca Raton, FL.
[4] Keld Bodker, Finn Kensing, and Jesper Simonsen. 2004. Participatory IT design: designing for business and workplace realities. MIT Press.
[5] Matthew Brannan and Teresa Oultram. 2012. Participant Observation. In Qualitative Organizational Research: Core Methods and Current Challenges, Gillian Symon and Catherine Cassell (eds.). Sage Publications, 296–305.
[6] John Campbell, Craig McDonald, and T Sethibe. 2009. Public and private sector IT Governance: identifying contextual differences. Journal of Information Systems 16, 2: 5–18.
[7] Ward Cunningham. 1992. The WyCash portfolio management system. ACM SIGPLAN OOPS Messenger 4, 2: 29–30. https://doi.org/10.1145/157710.157715
[8] Danish Ministry of Finance. 2017. Regeringens kasseeftersyn på it-området. Copenhagen, Denmark.
[9] Bent Flyvbjerg. 2006. Five misunderstandings about case-study research. Qualitative Inquiry. https://doi.org/10.1177/1077800405284363
[10] Chip Gliedman and Adam Grown. 2004. Defining IT Portfolio Management: Holistic IT investment Planning. Forrester Research.
[11] Isaac Griffith, Hanane Taffahi, Clemente Izurieta, and David Claudio. 2015. A simulation study of practical methods for technical debt management in agile software development. In Proceedings - Winter Simulation Conference. https://doi.org/10.1109/WSC.2014.7019961
[12] Sitwala Imenda. 2014. Is There a Conceptual Difference between Theoretical and Conceptual Frameworks? Is There a Conceptual Difference between Theoretical. 8923, January. https://doi.org/10.1080/09718923.2014.11893249
[13] Jimmi Kellerman and Patrik Löfgren. 2008. Application Portfolio Management - A Framework for Application Destiny Determination. University of Gothenburg.
[14] Tim Klinger, Peri Tarr, Patrick Wagstrom, and Clay Williams. 2011. An enterprise perspective on technical debt. In Proceeding of the 2nd working on Managing technical debt - MTD '11. https://doi.org/10.1145/1985362.1985371
[15] R Kumar, H Ajjan, and Y Niu. 2008. Information Technology Portfolio Management: Literature Review, Framework, and Research Issues. Information Resources Management Journal 21, 3. https://doi.org/10.1039/TF9696500098
[16] Steinar Kvale. 2008. Doing Interviews. Sage Publications.
[17] Mary Lacity, L. Willcocks, and David Feeny. 1996. The Value of Selective IT Sourcing. Sloan Management Review 37, 3: 13–25.
[18] Juha Lemmetti. 2016. Construction of Enterprise Architecture in Discourses Within the Public Sector. Electronic Government: Proceedings of the 15th IFIP WG 8.5 International Conference, EGOV 2016, 287–298.
[19] Johan Magnusson and Bendik Bygstad. 2014. Technology debt: Toward a new theory of technology heritage. ECIS 2014 Proceedings - 22nd European Conference

on Information Systems: 0–15.
[20] James D. McKeen and Heather A. Smith. 2010. Developments in practice XXXIV: Application portfolio management. Communications of the Association for Information Systems 26, 1: 157–170. https://doi.org/10.17705/1cais.02609
[21] Jeppe Agger Nielsen and Keld Pedersen. 2014. IT portfolio decision-making in local governments: Rationality, politics, intuition and coincidences. Government Information Quarterly 21: 467–483. https://doi.org/10.1016/j.giq.2014.04.002
[22] Mille Edith Nielsen, Christian Østergaard Madsen, and Mircea Filip Lungu. 2020. Technical Debt Management: A Systematic Literature Review and Research Agenda for Digital Government. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 12219 LNCS: 121–137. https://doi.org/10.1007/978-3-030-57599-1_10
[23] Ariadi Nugroho, Joost Visser, and Tobias Kuipers. 2011. An empirical model of technical debt and interest. Proceedings - International Conference on Software Engineering: 1–8. https://doi.org/10.1145/1985362.1985364
[24] Joe Peppard. 2003. Managing IT as a portfolio of services. European Management Journal 21, 4: 467–483.
[25] Narayan Ramasubbu and Chris F. Kemerer. 2014. Managing technical debt in enterprise software packages. IEEE Transactions on Software Engineering 40, 8: 758–772. https://doi.org/10.1109/TSE.2014.2327027
[26] Narayan Ramasubbu, Chris F. Kemerer, and C. Jason Woodard. 2015. Managing technical debt: Insights from recent empirical evidence. IEEE Software 32, 2: 22–25. https://doi.org/10.1109/MS.2015.45
[27] Nicolli Rios, Manoel Gomes de Mendonça Neto, and Rodrigo Oliveira Spínola. 2018. A tertiary study on technical debt: Types, management strategies, research trends, and base information for practitioners. Information and Software Technology 102, June: 117–145. https://doi.org/10.1016/j.infsof.2018.05.010
[28] Nicolli Rios, Rodrigo Oliveira Spínola, Manoel Mendonça, and Carolyn Seaman. 2018. The most common causes and effects of technical debt: First results from a global family of industrial surveys. International Symposium on Empirical Software Engineering and Measurement, October: 1–10. https://doi.org/10.1145/3239235.3268917
[29] Vital Roy and Benoit A. Aubert. 2002. A Resource-Based Analysis of IT Sourcing. Data Base for Advances in Information Systems 33, 2: 29–40. https://doi.org/10.1145/513264.513271
[30] Scale Agile Inc. 2021. PI Planning. Retrieved April 19, 2021 from https://www.scaledagileframework.com/
[31] Hans Jochen Scholl. 2006. Electronic government: information management capacity, organizational capabilities, and the sourcing mix. Government Information Quarterly 23, 1: 73–96.
[32] Carolyn Seaman and Yuepu Guo. 2011. Measuring and Monitoring Technical Debt. https://doi.org/10.1016/B978-0-12-385512-1.00002-5
[33] Daniel Simon, Kai Fischbach, and Detlef Schoder. 2010. Application Portfolio Management—An Integrated Framework and a Software Tool Evaluation Approach. Communications of the Association for Information Systems 26. https://doi.org/10.17705/1cais.02603
[34] The Swedish National Audit. 2019. Föråldrade it-system – Hinder för en effektiv digitalisering. Stockholm.
[35] C. Verhoef. 2002. Quantitative IT portfolio management. Science of Computer Programming 45, 1: 1–96.
[36] Charalabidis Yannis, Cunha Alexandra Maria, and Sarantis Demetrios. 2020. Foreword. In Proceedings of the 12 th International Conference on Theory and Practice of Electronic Governance, i–xxi.
[37] Robert K Yin. 2018. Case Study Research and Applications: Design and Methods. Sage Publications.