Universiteit Leiden

ICT in Business

Exploratory Research on the Concept of IT Debt

Name: Ran An

Student-no: S1046853

Date: 28/08/2014

1st supervisor: Dr. Hans Le Fever

2nd supervisor: Dr. Aske Plaat

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)

Leiden University

Niels Bohrweg 1

2333 CA Leiden

The Netherlands

ACKNOWLEDGEMENTS

It would not have been possible to write this thesis without the guidance of my supervisors, help from my friends, and support from my family and my husband. I would like to express my sincere gratitude to all of them.

First and foremost, I want to thank my first supervisor, Dr. Hans Le Fever. With his excellent guidance and insight into the IT industry, I was always inspired and stimulated by his constructive suggestions. Without his support and active participation in every step of the process, this thesis may never have been completed. I also would like to thank Dr. Aske Plaat, my second supervisor, for his time spent to be my committee member and helping me improving the quality of this thesis with his valuable comments.

Next, I highly appreciate the help from Leo Wielstra, my manager and company supervisor, for offering me the graduation internship opportunity in this large bank to conduct the case study. Beside this research, I was also given heavy responsibilities for different company projects which pushed me through the learning curve quickly. I have been extremely lucky to have a manager who trusts me so much. I would also like to thank my colleagues at the bank. All of the staff that I worked with were wonderful people.

My thanks also go to all the lecturers, staff and all my classmates in Leiden University, especially Yu Long, Xi Cui and Fei Liu. It is your direct and indirect support that makes my study at Leiden enjoyable. It is a real pleasure to know all of you.

I would like to thank my mother and my young brother, who are always supporting me and encouraging me with their best wishes. My thanks should also be addressed to my father in heaven for the best of love he gave me.

Finally, I would like to thank my husband, Haiyang Cui. Thank you for being with me through the good times and bad. Thank you for inspiring me with your PHD research experience. Without your help my life would have been much tougher. Thanks for all the happiness you brought into my life.

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Many companies state the challenge of IT debt when they aim to rationalize their applications portfolio or try to reduce the complexity of their enterprise architecture. However, this concept is narrowly understood by both practitioners and researchers. Practitioners refer to IT debt as the cost of clearing the backlog of maintenance to bring the corporate applications portfolio to a fully supported, current release state. This definition only takes into account the existing applications, while the debt incurred by introducing new applications (built or being built) is not included. On the academic side, literature on IT debt can hardly be found. The concept of IT debt is often confused with Technical debt which is solely related to software development. This research aims to provide a clarification of these two concepts by also taking into account debt from e.g. non-compliance with enterprise architecture standards. A conceptual model of IT debt is proposed on the basis of a literature review, and validated and supplemented by a case study conducted in a large Dutch bank. This proposed model incorporates a new perspective to the concept of IT debt and could assist practitioners in their understanding and containment of IT debt.

Keywords: IT debt; Technical debt; Application portfolio; Rationalization; Enterprise Architecture

TABLE OF CONTENTS

CHAPTER 1 – INTRODUCTION

The fact that companies have been continuously adding new functionalities to their IT systems has led to complex enterprise architectures. This stems from both increasing business needs and the eruption of new technologies on the market. As demand grows for more agile and innovative systems, IT leaders are always under pressure to squeeze more performance and value from their existing infrastructure and application portfolio, leaving a backlog of maintenance and enhancements relating to business change and efficiency. The cost of cleaning this backlog to bring the corporate applications portfolio up to date is labeled as "IT Debt" by Gartner [1].

1.1 Definition of IT Debt

The widely accepted definition of IT debt was promoted by Gartner in 2010. Gartner defines IT debt as "the cost of clearing the backlog of maintenance that would be required to bring the corporate applications portfolio to a fully supported current release state [1]". 'IT Debt' has now become a popular term in the IT world to provide a quantifiable measurement of IT backlog of incomplete or yet-to-be-started IT projects.

1.2 Problem Statement and Research Question

Even though the concept of IT debt has been coined a few years ago, it is still not well defined and is often misused and misunderstood by both practitioners and academics. As a newly developed concept, IT debt has only been defined by the practitioners in industry with a comprehensive, prudent approach to manage IT debt obligations. The main problem of the definition given by Gartner is that it only covers half of the real IT debt which is the debt in the existing applications. The other half of the problem, IT debt accruing from newly built applications is not included in this definition. While in reality it is equally important to prevent new IT debt from coming into existence as it is to solve the existing IT debt. Compared to take on debt, "get it right the first time" don't accrue interest to be paid later, which refers to the efforts needed to get rid of IT debt.

On the academic side, IT debt has not drawn much of the attention from the researchers as most of them are still misusing the concepts of technical debt and IT debt. Among the available academic literature and other internet sources, IT debt is often confused with

Technical debt which is solely related to software development. Tom stated in his paper that "the global technical debt bill is estimated by Gartner to be $US500 billion in 2010…" [2]. Thibodeau [3] and Szykarski [4] also made similar citations to Gartner results in their papers respectively. On the other side, practitioners are also using Technical debt while they are referring to IT debt [5]. In some of the articles, the term "IT technical debt" is also used [6]. According to the authors' knowledge, there is no academic research on defining the concept of IT debt. If we use "IT debt" as searching keywords, there are no results returned from Google Scholar. The same keywords have also been used to search through the Leiden University Library Catalogue[1], Scopus[2] and IEEE XPlore Digital Library[3], and still no relevant literature was found.

This research aims to provide a clarification of IT debt and Technical debt by examining the different characteristics of these two concepts. The key research question to be answered in this research is:

- Is there a difference between Technical debt and IT debt?

1.3 Motivation and Legitimization

The management of IT applications with a complex architecture can be challenging. The poor management of application portfolios makes the IT systems vulnerable, hindering their evolution. Significant effort has been devoted to tackle the IT debt problem. Application portfolio management (APM), a framework for managing enterprise IT software applications and software-based services, has emerged in middle to large size IT organizations since the mid-1990s. It applies cost benefit analysis and other business analytics to IT decision-making to justify and measure the business benefits of each company's software applications. It provides managers with an inventory of the company's software applications, the value of each application, and the health of the IT infrastructure. Although application portfolio management provides various tools to rationalize the IT portfolio, these efforts cannot solve the full IT debt problem which is the deviation between the optimal and current IT landscape due to deferred maintenance and update of the IT applications [1]. Given this situation, companies together with academic researchers have been actively seeking for optimal

[1] http://catalogue.leidenuniv.nl/

[2] http://www.scopus.com/

[3] http://ieeexplore.ieee.org/

solutions to address IT debt. Some of the possible solutions proposed are application retirement, cut the backlog, architectural reviews [7] and application inventory [8].

But these methods are not sufficient to solve the IT debt problem. Most of them only cover part of the attributes and part of the dimensions of IT debt. As a matter of fact, the IT debt problem is getting worse. According to a survey conducted by Gartner [1], the global IT debt will rise to $1 trillion by 2015, which is twice as much as the amount in 2010. This number is only the cost of fixing the existing IT debt, not considering the losses from malfunctioning IT [9]. Besides, opportunity cost, which the company could utilize in situations without IT debt, is also not taken into account.

This predicament in tackling the problem is caused by the complex nature of IT debt itself. Both the compounded relationships and interdependencies between the IT applications lead to an insufficient understanding of the whole IT system, which is getting more complex every year. The desperate engineers even declare that they don't know where to start to ease the growing burden of IT [10]. The accelerating pace of change is another reason which made it difficult to solve IT debt [11]. On the business side, the demands for new applications or functionalities are increasing, and this also leads to shrinking timelines for delivery. On the IT side, old application delivery methods are still being used and solution cannot be delivered fast enough and changes cannot be introduced easily and frequently enough.

The confusion and misunderstanding about the two concepts is quite serious at the moment. More than half of the literatures reviewed in this research show the confusion either explicitly or implicitly. Therefore there is an urgent need to address this misunderstanding for the following reasons. First, in developing IT systems, there may be a number of teams involved. As soon as you are in a team, communication becomes the cornerstone of everything you do. In order to facilitate effective communication, it is crucial to have a common language among all involved parties to get consensus in the definition of specific terms.

And secondly, solving the problem of IT debt would take much longer time and efforts without a correct recognition of the problem. Many researchers have explored the nature of problem solving processes and have described these processes in terms of a cycle as shown in Figure 1 [12, 13, and 14]. In this problem solving cycle, identifying the problem to be solved is the first step in the whole problem-solving process followed by other steps. This makes recognition of the problem crucial in the problem solving processes. In case something went wrong in this step or if this step is skipped in the beginning, it would cause delays in the following steps and in the worst case the whole process need to be started all over again. Due to the limited time allowed, this research only focuses on the first two steps of the problem solving cycle in solving IT debt problem, which is to identify the problem and to define the problem.

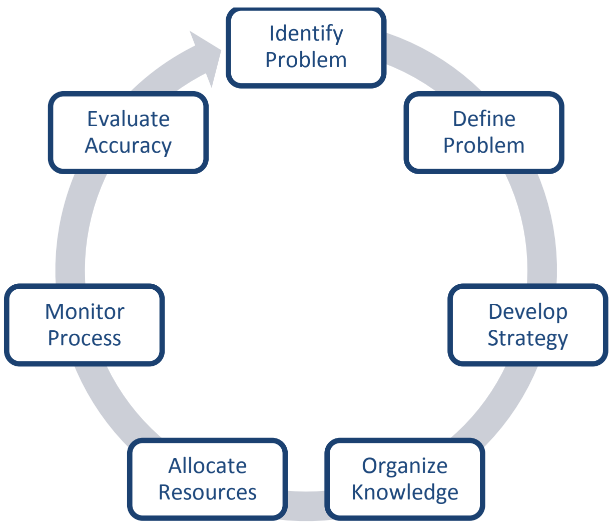

Figure 1 Problem Solving Cycle

1.4 Contributions of This Thesis

My contributions in this thesis include:

- The difference between IT debt and Technical debt is clarified
- A new conceptual model of IT debt is built and validated by a case study
- A new definition of IT debt is proposed.

This research is an exploratory research on the concept of IT debt, so outcomes of IT debt and possible solutions to address the IT debt are left out of scope. A preliminary model of IT debt is proposed on the basis of a multivocal literature review. This proposed model will be further validated and supplemented by a cased study conducted in a large Dutch bank. And research findings will be consolidated into a conceptual model of IT debt and could be used to assist practitioners in their understanding of IT debt. During this research, the current misunderstanding of IT debt and technical debt is also clarified. Although this theoretical distinction between IT debt and technical debt do not directly help practitioners reduce the debt in the field. The correct understanding of the problem is vital to find correct solutions to the problem.

1.5 Reading Guide

The remainder of this thesis is structured as follows: chapter 2 explains the research methods applied in this research; chapter 3 and 4 explain in details the existing technical debt literature and IT debt literature respectively; the output of chapter 4 is a preliminary model of IT debt extracted from the existing literature. A case study is designed and conducted in chapter 5 to validate and improve the preliminary model. The findings from the case study are consolidated into the model in chapter 6 and chapter 7 is the discussion and suggestions for future research.

CHAPTER 2 – RESEARCH METHODS

This research was conducted with various research techniques being applied in three stages as shown in Figure 1. In the first stage, the research gap was identified, and it was presented in the previous chapter. In the second stage, three different exploratory research approaches have been applied. In the third stage, the outputs from stage two were compared with each other to answer the research question raised in the previous chapter.

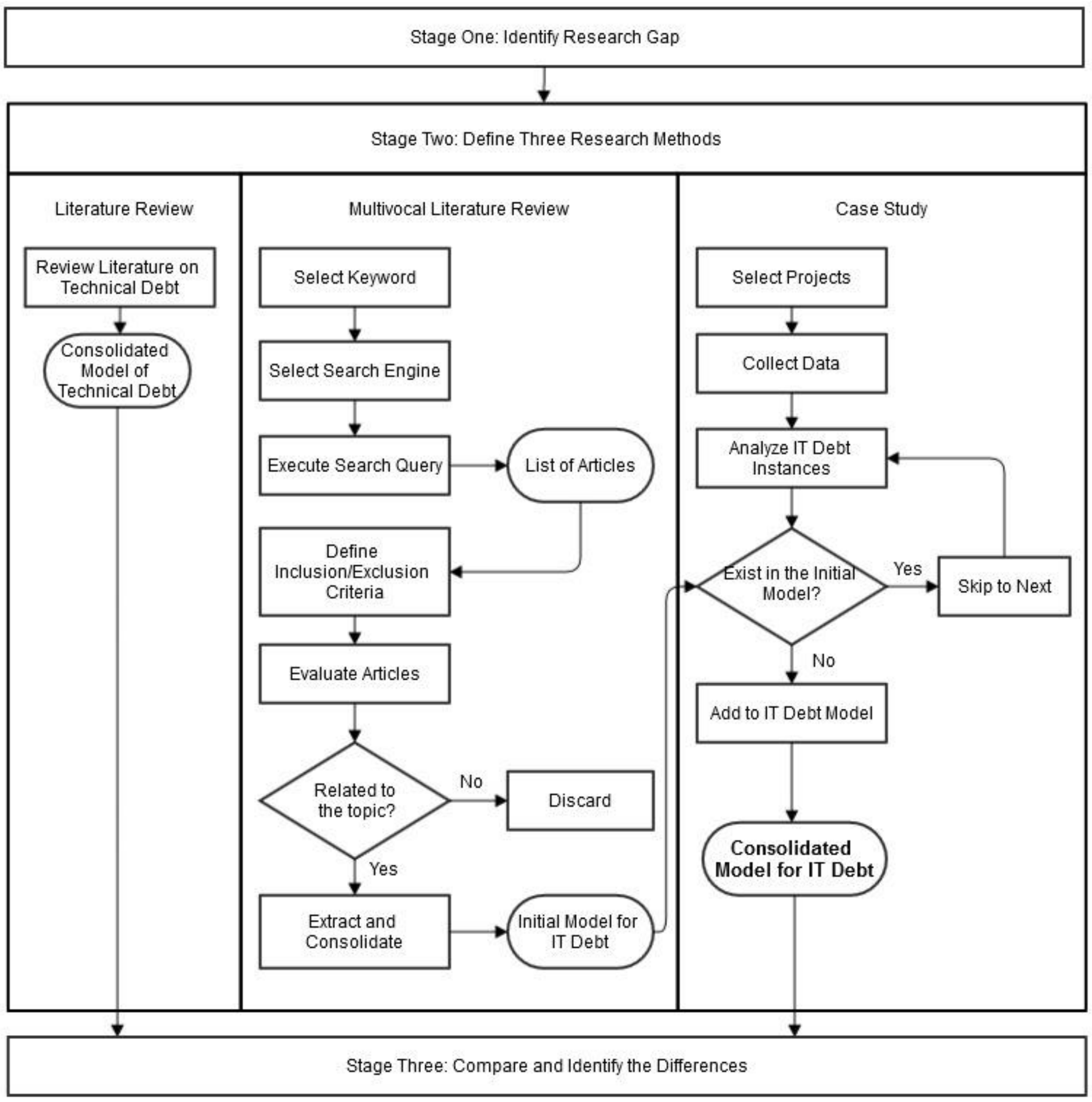
Stage One: Identify Research Gap
Stage Two: Define Three Research Methods
Literature Review
Review Literature on Technical Debt
Consolidated Model of Technical Debt
Multivocal Literature Review
Select Keyword
Select Search Engine
Execute Search Query
List of Articles
Define Inclusion/Exclusion Criteria
Evaluate Articles
Related to the topic?
No
Discard
Yes
Extract and Consolidate
Initial Model for IT Debt
Case Study
Select Projects
Collect Data
Analyze IT Debt Instances
Exist in the Initial Model?
Yes
Skip to Next
No
Add to IT Debt Model
Consolidated Model for IT Debt
Stage Three: Compare and Identify the Differences

Figure 2 Research Methods applied in this thesis

2.1 Literature Review on Technical Debt

Since one of the objectives of this research is to clarify the IT debt and technical debt concepts, it is equally important to examine the concept of technical debt. And because the metaphor to financial debt was firstly applied in technical debt, the study of technical debt will also help understanding the nature of the metaphor itself. In this step, the conceptual Technical debt model developed by Tom [7] was adopted as a reference guide to the concept of technical debt.

2.2 Multivocal Literature Review

As stated in the previous chapter, academic literature is lacking relating to this topic. Multivocal literature review, meaning “all accessible writings on a common, often contemporary topic” can be viewed as a form of original research [15]. This approach has been followed to identify other accessible non-academic writings on the internet.

Google search engine was chosen as the data source for MLR. Since one of the research objectives is set to clarify the IT debt concept (how the term “IT debt” is understood by practitioners), only the term “IT debt” has been used as the keyword for query. Other terms like software debt or design debt are actually referring to technical debt, which will be addressed in details in the next chapter.

After reading the title and short introduction, irrelevant posts that were not about IT debt, such as articles about financial debt, were excluded. Posts about IT debt but identical to each other were also excluded. As determined by the Google ranking algorithm, the first 47 relevant posts were selected as the input for further analysis (see Appendix B). These web posts were carefully read and useful info was extracted and grouped into a preliminary model.

Ogawa and Malen also stated that “reviews of multivocal literatures are suggestive and instructive, not definitive or conclusive” [15]. So the preliminary findings need to be further investigated and consolidated with other research methods.

2.3 Validation by Case Study

According to Yin, a case study is “an empirical inquiry that investigates a contemporary phenomenon within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident [16].”

In this research, the proposed preliminary model is validated by a case study in a large Dutch bank where exploration on IT Debt is ongoing. In consideration of confidentiality issues, this bank is referred to in the following text as “the bank”. The bank started the IT debt analysis in April 2013. Until the time of this research, IT debt data on projects are only available from beginning April 2013 to the end of December 2013. Data on IT debt were collected through relevant document reviews, observations, meetings, and interviews with IT architects and other related stakeholders. In total 50 IT debt instances generated from 31 IT projects were collected and analyzed. Characteristics of these IT debt instances were compared with the proposed model and on the other hand the preliminary model was also tested in this context. New attributes found in the case study were added to the model as a supplement to it. The output of this step is a consolidated conceptual model of IT debt.

In the last stage, the consolidated model on IT debt was compared with the conceptual Technical debt model. Both characteristics that are different and characteristics in common are compared. In this way, we could be able to answer the research question “Is there a difference between Technical debt and IT debt?”, or to be more specific: “what are the main differences between technical debt and IT debt?”

CHAPTER 3 – TECHNICAL DEBT

This chapter aims to provide a clear explanation of the technical debt, its definition and its characteristics. It refers to the first column in Figure 1. To date, technical debt has been widely used as a rhetorical device for both technical communication and for communication between engineers and non-technical stakeholders. The technical debt metaphor was first introduced by Ward Cunningham in his OOPSLA experience report in 1992 as follows [17],

> "*Shipping first time code is like going in to 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 organization can be brought to a stand-still under the debt load of an unconsolidated implementation, object-oriented or otherwise*."

After coined by Ward Cunningham, this metaphor has since been taken on by the industry to describe the consequences of poor software architecture and bad coding. His definition was regarded as a starting point and reference definition, and was developed by various researchers in the software development community. In 2010, Brown et al. pointed out that "technical debt is used to describe a situation in which long-term code quality is traded for short-term gain, creating future pressure to remediate the expedient [18]." Ktata and Levesque made an important extension of the original definition also in 2010 [19]. Their definition is not limited to code quality in software development, but included many other aspects. They stated that technical debt is any side of the current software system that is considered sub-optimal from a technical perspective, which extended the definition to its broadest.

Recent researches are mostly focused on the management of technical debt. As pointed out by Allman, technical debt is inevitable [20]. We should not aim to eliminate it, but rather manage it. Effective management of technical debt is perceived as critical to achieving and maintaining software quality. The SQALE (software quality assessment based on life-cycle expectations) Method is developed to estimate the technical debt and analyze upon technical and business perspective [21]. Guo and Seaman proposed a portfolio approach similar to financial portfolio management to "determine the optimal collection of technical debt instances that should be incurred or held [22]."

3.1 Comparison to Financial debt

Since the term technical debt is a metaphor related to financial debt, there must be something in common between the two. The technical debt is similar to financial debt in the way that it is a burden to the stakeholder, which needs to be paid back in the future in the form of principal and interest. The principle amount of technical debt is the work that engineers need to do to change the system to the desired state. And the interests for taking on technical debt are the extra costs that need to be paid compared with situations without technical debt.

In the article "Managing Technical Debt [20]", Allman made a comparison of the technical debt and financial debt. They are similar in three aspects. Firstly, the person making the loan, either financial debt or technical debt, wants to be repaid eventually. Second, both of the debts need to be paid back with interest. Third, the failure of paying back can result in very bad consequences.

Though technical debt and financial debt are similar in those ways, there are some differences as well. The most significant one is that the person who takes on technical debt is not necessarily the one who has to pay it off. Therefore, the developers are encouraged to focus more on the speed of software delivery than on the maintainability of the software. Another difference is that technical debt almost never has to be paid off entirely. If the cost of paying back exceeds the interest you ultimately accrue, there is no point to pay it back in the first place. In some of the instances when the whole software project failed, there are no needs to pay back technical debt in the project either.

3.2 Categorization of technical debt

The two most well-known and most commonly adopted methods to categorize technical debt were proposed by McConnell [23] and Fowler [24] respectively.

Steve McConnell, CEO and chief software engineer at Construx Software, broke technical debt down to two basic kinds: unintentional debt and intentional debt depending on whether the debt is identified before the acquisition. The later form of debt appears when the organization consciously makes the decision to prioritize the short term benefits than the long term sustainability. It can be further broke down to two categories: short term (tactical) and long term (strategic) debt. Short term debt is taken on tactically and reactively, and is expected to be paid off frequently while long term debt is taken on strategically and proactively, and is expected to be carried on for a few years or longer.

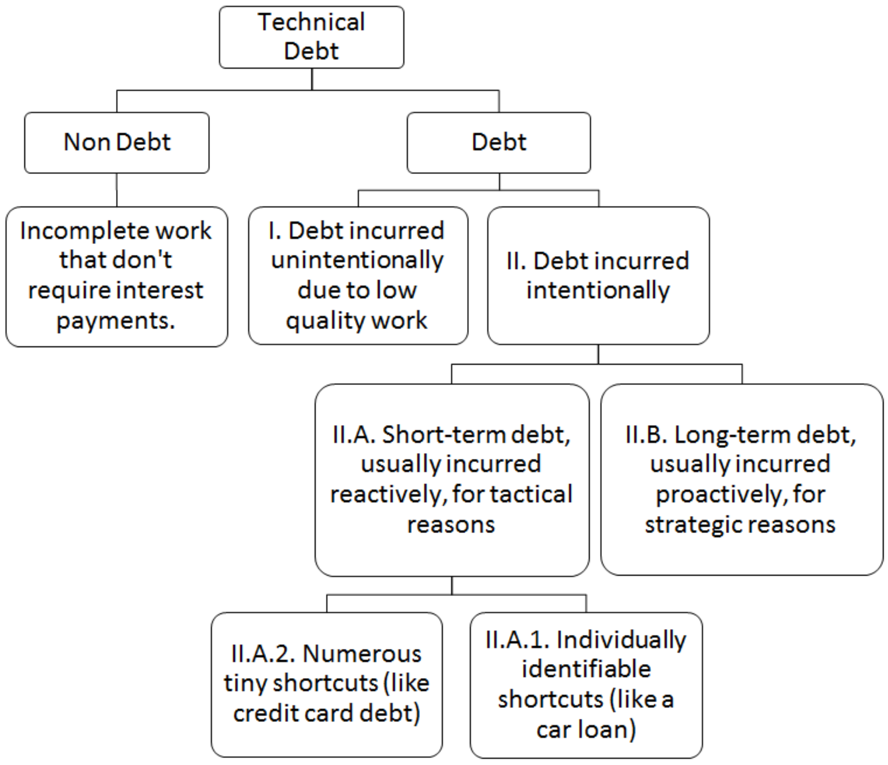
Technical Debt
Non Debt
Debt
Incomplete work that don't require interest payments.
I. Debt incurred unintentionally due to low quality work
II. Debt incurred intentionally
II.A. Short-term debt, usually incurred reactively, for tactical reasons
II.B. Long-term debt, usually incurred proactively, for strategic reasons
II.A.2. Numerous tiny shortcuts (like credit card debt)
II.A.1. Individually identifiable shortcuts (like a car loan)

Figure 3 Technical Debt Taxonomy by McConnell

Martin Fowler breaks down technical debt into four quadrants which are defined by two axes: reckless/prudent and deliberate/inadvertent (see Figure. 4). Prudent/Reckless differences the choices to take on technical debt, i.e. whether the decision is made carefully enough. The difference between Deliberate/Inadvertent debt is whether the team is aware that they are taking on technical debt. Prudent-Deliberate debt refers to debt that is taken on after careful thoughts on the costs, benefits and consequences. Reckless-Deliberate debt is created when the team decided to accept a workaround instead of the right solution because they do not have enough time to implement the right one. The third category Prudent-Inadvertent debt becomes visible at the moment when the team realizes what the right solution should have been despite the fact that the team has followed all principles prudently. And the last quadrant Reckless-Inadvertent debt is made by the team that are ignorant of coding practices and without knowing how much trouble they are getting into.

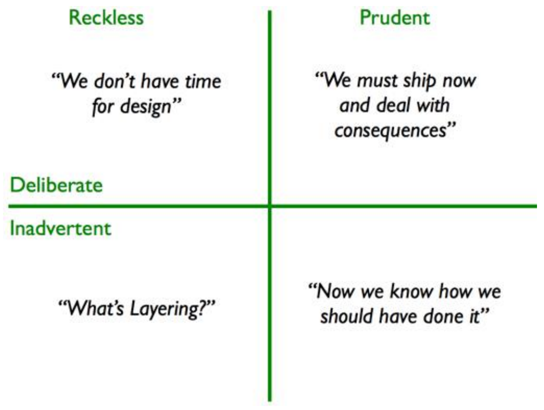
Reckless
Prudent
"We don't have time for design"
"We must ship now and deal with consequences"
Deliberate
Inadvertent
"What's Layering?"
"Now we know how we should have done it"

Figure 4 Technical Debt Quadrant, Fowler, M.[21]

According to Fowler, this metaphor is quite helpful to communicate to non-technical people. However, out of the four categories, the forth one cannot map easily to the financial model: Prudent-Inadvertent. You can't conceive of a parallel with taking on a prudent-inadvertent financial debt. As a result, explaining to managers why this debt appeared is challenging. In view of Martin Fowler, this kind of debt is inevitable and should be expected.

3.3 Existing Conceptual Framework

A comprehensive framework of Technical Debt was proposed by Tom in 2013 aiming to provide a systematic approach to understand the overall phenomenon of technical debt for practical purposes. The summary of their theoretical framework of technical debt is presented in Figure 5 [7]. The framework consists of three basic blocks: Precedents, Technical debt and Outcomes. Precedents of technical debt include pragmatism, prioritisation, process, attitudes, ignorance and oversight. They are the necessary but not sufficient conditions of technical debt. If technical debt is not addressed at the first place because of any of the precedents, it will occur and let you pay back later. In the technical debt block, the dimensions and attributes of technical debt are established.

The dimensions of technical debt are comprised of Code debt (not writing clean code), Design & Architecture debt (which requires refactoring in the future), Environment debt (hardware where the software is deployed is temporary), Knowledge distribution debt (only the programmer himself knows about the code) and Testing debt (not enough coverage of the code). Apart from the dimensions of technical debt, the attributes of technical debt are also recognized to reveal additional facets of technical debt.

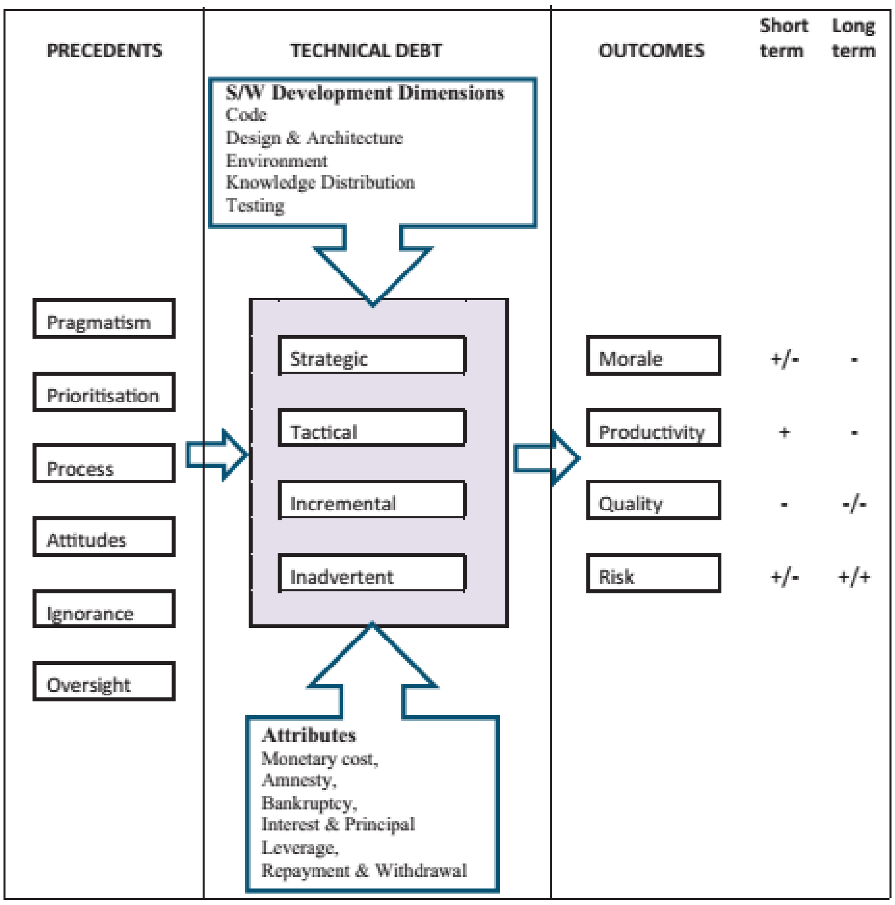

Figure 5 Technical Debt Conceptual Framework, Tom, E. [7]

McConnell's classification of technical debt is adopted in which the technical debt is classified into four groups, strategic, tactical, incremental and inadvertent. As one of the key outcome of Tom's research, new technical debt taxonomy is proposed to describe and encompass different forms of the technical debt phenomenon. The outcomes of technical debt is described in two dimensions short/long term and morale/productivity/quality/risk.

This framework incorporates the different dimensions, attributers, precedents and outcomes of technical debt. According to the author's knowledge, it is the most comprehensive conceptual framework up to date and therefore is adopted in this research and will be used to compare with the model of IT debt.

CHAPTER 4 – MLR ON IT DEBT

Due to the lack of academic literatures about IT debt, it is not possible to conduct an academic literature review. So all accessible writings or publications on the internet were used, which is called Multivocal Literature Review (referred to as MLR in the next paragraphs). The following section provides the analysis on the IT debt MLR results. The findings of MLR will be used to build a preliminary model of IT debt. To make it easier to be compared with Technical debt model, the same wording as Tom used in the Technical debt model was used, such as “precedents” to refer to reasons that enterprises take on IT debt; and “dimensions” to refer to the different forms of IT debt that were found in the multivocal literature review [7].

4.1 Precedents of IT debt

This section presents the different precedents of IT debt that were found and extracted during the MLR process. The entire list of the articles reviewed could be found in Appendix II at the end of this thesis.

4.1.1 Limited budget

Limited budget as one of the causes for IT debt is found almost in all of the literatures reviewed. According to Gartner survey, IT budgets were held tight or even reduced over the last decade. At the meantime, business is requiring more functionality or services to be delivered. The reaction of the IT department is to defer maintenance and upgrade work that should take place, thus creating IT debt. As companies invest in a larger applications portfolio, the increasingly complex IT system requires more maintenance work. If IT debt is not tackled, it will ultimately pose a systemic risk to large companies.

4.1.2 Absence of structured review

Another cause for IT debt is the absence of a structured review process for the application portfolio. At present, most companies are dependent on a large set of applications to support their businesses. These applications are developed at different times and as a result at a different state in their life cycle. Considering the large number of total applications and their interdependencies, the application overhaul required will be less visible without a structured review process. The result of this invisibility is the ignorance of what should have been done. When the IT debt is visible itself, it will be too late for IT departments to catch up with the continuously expanding maintenance backlog.

4.1.3 Short-sighted view

One of the primary causes of IT debt is believed to be that “business managers typically have little real incentive to take a long-term view” [26]. The goals for business managers are to maximize performance or to get a high ROI on IT investments. This unawareness attitude is typically caused by the poor understanding of IT debt both from the business side and the IT people. According to an independent survey of Vanson Bourne in April 2012 commissioned by MicroFocus, 46% of the total 590 IT decision makers participated in the survey admitted that they “do not know the value of their IT debt” [27].

4.1.4 Imposing deadlines

Another reason for IT debt is claimed to be the imposing deadlines. But the root cause is that the rapidly accelerating pace of change is shrinking timelines [28]. On one hand, business demands for new applications and functionality as their responses to change to seek new sources of growth. This has added much pressure and workload on IT. On the other hand, IT departments are still relying on the traditional application delivery methods of “either customizing rigid legacy systems or coding custom applications”. As a result of this, changes cannot be made to the IT system easily and frequently enough. Facing the pressure from business, trade-offs are made in long term maintainability to get short term benefits. One could argue that compliance is a separate reason for IT debt, but in this research it is grouped into the category of rapid changes. Since the economic crisis, people are becoming increasingly aware of the compliance issues to mitigate any potential risks. In one of the MicroFocus weblogs Derek Britton pointed out that “Regulations and legislation on data privacy, banking standards, customer information and international taxation have added to the workload” [29]. Ewan Withers also stated in his weblog that regulatory deadline is a solid business reason why a project has to be implemented quickly. In such situations, shortcuts are taken to meet the deadline, leaving “a legacy of unfinished work” [30]; also compromises are accepted to do more compliance work and the regular maintenance are put aside for later.

4.1.5 Outsourcing

Outsourcing could also bring IT debt to companies as well as numerous advantages. Vinnie Mirchandani said in his weblog that many organizations outsourced their application maintenance to Accenture and other IT service providers. Few of the outsourcing companies offer to refresh portfolio analysis after the first portfolio exercise when they first take over the application inventory [31].

4.1.6 Ignorance

The term "ignorance" could be used to describe one of the reasons for IT debt. To be distinguished from the unawareness of IT debt, ignorance refers to the fact that the organization doesn't know how to deal with their IT debt; or in other words, the organization is not capable of solving the IT debt problem. Do not know where to start and do not have the right tools are two concerns among the difficulties to solve IT debt [29]. The only result of this inability would be an expanding IT debt list.

4.2 Dimensions of IT debt

The most common dimension of IT debt is the maintenance and upgrade debt, as stated in its definition. All papers reviewed talked about this form of IT debt. Deferred maintenance will lead to a system with aging enterprise applications that in reality are "collections of data and business logic encapsulated in programming instructions and myriad platform components" [32].

A second form of IT debt can be found in the form of poor enterprise architectures. It refers to the fact that for example business applications are not integrated or information is manually duplicated [26].

Knowledge is another dimension of IT debt. It consists of knowledge transfer debt between employees and documentation debt. Shull pointed out that "… original knowledge of the application and supporting data structure is no longer in the organization [33]". Also in this weblog, it is claimed that 73% IT decision makers in the interview admitted that their organization's application documentation is incomplete.

4.3 Preliminary Model of IT Debt

The findings of the MLR are summarized in Table 1. Since Gartner's definition on IT debt is the most wildly accepted definition among the literatures, this model will continue to use Gartner's definition. The precedents and dimensions of IT debt, which are explained in details in 4.1 and 4.2 respectively, are listed in the two columns under the definition of IT debt. Due to the fact that this model is derived from limited number of literatures reviewed, the model needs to be validated for the completeness of the precedents and dimensions. In chapter 5, a case study is designed and conducted to validate and supplement this model.

IT debt definition	
The cost of clearing the backlog of maintenance that would be required to bring the corporate applications portfolio to a fully supported current release state.	
Limited budget Absence of structured review Short-sighted view Imposing deadlines Outsourcing Ignorance	Maintenance & Upgrade Enterprise Architecture Knowledge Distribution
Precedents	Dimensions

Table 1. Preliminary Model of IT Debt

CHAPTER 5 – CASE STUDY

This case study is designed to test the IT debt model that we proposed in the previous chapter. The bank where the case study was conducted serves different groups of customers in the Netherlands and across the globe. It has already hundred-year's history of existence. During and after the 2008 economic crisis, the bank has undergone several mergers and acquisitions. These facts have led to a rather complex IT landscape of the bank. The cost of operating such a system is rather high compared to the same financial industry counterparts. According to an internal document, the IT cost of the bank is one third higher than other banks in the Netherlands.

5.1 Initiative within the Bank

Early in 2013, the bank has conducted a comprehensive research on the as-is situation and to-be situation in 2020. After identifying the gaps, a roadmap of business transformation towards year 2020 was developed. IT belongs to one of the themes in this roadmap. The survey revealed that the current IT cost is too high compared to its industry counterparts in the Netherlands, and IT debt is one of the causes to this high IT operation cost. According to an internal analysis, 23% (31 out of 133) projects started during April to December 2013 created IT debt. Therefore, the bank has decided to find a solution to eliminate IT debt. This research is part of this project.

5.2 Case Study Findings

All 50 IT debt instances are listed in Appendix I. Due to confidentiality issues, the project name are not shown in the list, and each IT debt instance is given a unique number instead of the project code as an identifier to it.

Table 2 is the simplest analysis of the total 50 IT debt instances within the bank. A distinction was made between the existing IT debt and New IT debt. Existing IT debt refer to the debt that was caused by outdated application portfolio as in the Gartner definition. New IT debt refers to the debt that didn't exist before and is introduced by the project, which is exactly the other perspective that Gartner's definition is missing. For example IT debt instance nr.18, the outdated data warehouse which doesn't belong to the long term application portfolio was still used due to the fact that it was the only solution in the given context. One example for

New IT debt could be the choice to use MySQL for data storage in IT deb instance nr.16. According to the enterprise application portfolio, Oracle is the target in long term. But switch to Oracle will take enormously more time and effort to rebuild the whole application than switch to MySQL. Out of this reason MySQL was chosen as a temporary solution which will be replaced in the future with Oracle when enough time and budget is available. This would not be labeled as IT debt according to the Gartner definition.

	Number of IT debt instances
Existing IT debt	27
New IT debt	23

Table 2. Statistics on existing IT Debt and New IT Debt

5.2.1 Precedents of IT debt

Table 3 below shows a statistical analysis of the reasons that IT debt was created in the bank. The detailed explanations are presented below.

Precedents	Number	Percentage
Target Solution not available	14	28%
Tight timeline	13	26%
Minimize Cost	11	22%
No Future State Guidelines	5	10%
Limited prior knowledge	4	8%
Out of scope	3	6%
Total	50	100%

Table 3. IT Debt Precedents Identified in Case Study

Target solution not available is the top reason for IT debt creation within the bank. For example in IT debt instance nr.12, the required Linux version was still premature; the bank didn’t want to run any risks. So the backend system was implemented on the Sun Platform which was not in the target portfolio but believed to be safer. And re-platform to Linux will take place when the required version is proved to be safe. Like outsourcing, this reason can be seen as typical external dependency upon the software vendors. On the other side, IT debt accrued due to internal dependency can also be found in this case. For example in IT debt instance nr.29, the bank is aiming to implement bank-wide single sign on mechanism. But this project still used a different set of username and password because at the time that project was implemented, the single sign on environment was not yet available. A transformation project needs to be planned when SSO is ready to use.

The second reason for IT debt creation is because of tight timeline. This reason can be mapped to prioritization. In IT debt instance nr.46, the government tax office required the new functions of the software to be in use before 2014; due to this tight deadline, the bank has decided to continue to work on the non-target application provided by PeopleSoft to avoid the penalties from the local authorities. In this instance, the bank has prioritized the delivery of new functions over the target enterprise architecture.

Minimize cost is the third most obvious reason for IT debt within the bank. As mentioned previously, this decision is made out of pragmatism. For example in IT debt instance nr.32, it was believed that there won’t be many WMB users in the first increment of the project. Balancing the cost and benefit of switching to this standard building block, it was decided not to use this standard building block in the first increment, and will change to WMB in the second increment when there are enough users.

No Future Guidelines is also one of the reasons for IT debt. In IT debt instance nr.19, the requirements as when and where to back up the received/sent fax were unknown. So the project created a temporary solution to back up the fax when the stack is full. It has to switch to the target solution when the requirements are made clear. This is also a typical reason for internal dependency. If the departments responsible for making requirements could have the requirements available earlier, the temporary solution could have been avoided.

The next reason found in the projects is limited prior knowledge. One example is IT debt instance nr.39. In the beginning of the project, the real capability of T24 application was unknown; only after the temporary solution was implemented, it was found out that T24 was able to provide the same functionality. If the project has adopted another approach, which is to investigate the real capability of T24 application before implementing the temporary solution, this IT debt instance could also have been avoided. In IT debt instance nr.18, after investigating all possible solutions, the only one feasible is to accept IT debt and use a datacenter that is to be decommissioned. This incapability of clearing IT debt can be seen as an example of ignorance.

The last reason for IT debt is that the required change is claimed out of scope. In the roadmap towards 2020, the bank has decided to implement service oriented architecture. But in IT debt instance nr.2, cross domain communication was still using the old solution and not via standard middleware because it was declared out of scope for this project. The reason behind this is that there was no requirement from the user to do this. The pragmatic team was not willing to upgrade a system which would bring no improvement in user experience.

Besides, another reason which is not included in the available project documents also contributes to the creation of IT debt. That is employees' attitude. This conclusion is drawn from the author's observations. During the 5 months of stay in the head office, the author has also participated in other projects. Many intangible aspects of the bank, such as the culture and the way of working were under observation both intentionally and unintentionally. One of the findings is that the bank has always had a customer oriented culture. Short term customer needs always take precedence over long term company development, thereby increasing the chance of generating IT debt.

5.2.2 Dimensions of IT debt

Table 4 below provides an analysis of different dimensions of IT debt found in the case study. The numbers on maintenance/upgrade debt and enterprise architecture debt are more or less as expected. But the knowledge distribution debt within the bank is surprisingly high. All projects included in this case study have certain amount of documentation debt due to the fact that not all project related documents were available. Among the available documents, not all required information was carefully written for future reference.

	Nr. of IT debt instances
Maintenance & Upgrade	28
Enterprise Architecture	22
Knowledge Distribution	50

Table 4. IT Debt Dimensions

5.2.3 Categorization of IT debt

McConnell pointed out during one of the interviews to him that technical debt categorization could be used in two contexts [34]: it helps to decide whether the debt is good or bad; and it helps to decide whether pay down the debt. Since the nature of decision for IT debt and Technical debt creation are the same, that is to use debt as leverage to the limited resources, the categorization on Technical debt also applies to IT debt. McConnell provided a widely accepted classification on Technical debt; and his classification was further developed by Tom in his Technical debt framework [7]. In this research, the revised classification is adopted because of its simplicity and clarity on the concept. Table 5 below presents a classification of IT debt instances accrued in the bank.

Classification	Nr. of IT debt instances
Strategic	2
Tactical	24
Incremental	11
Inadvertent	13
Grand Total	50

Tabel 5. IT Debt Classification

Strategic debt is "usually incurred proactively for strategic reasons [23]." The debt is strategic when it is either crucial for the company's continued existence or when there is only a small window of opportunity for new implementation to arrive on the market [7]. For example in IT debt instance nr.41, if the new application was not ready to use before a certain time, the bank would face a huge amount of fine from the local government or even losing the license to operate there. So it is definitely a strategic decision to take on IT debt in the first increment to deliver the required application to the users.

Tactical is described as "a late-stage measure to get a specific release out the door" [23]. The debt is tactical when sub-optimal solutions are adopted in a reactive manner to borrow time [7]. For example in IT debt instance nr.28, the target solution for credit system is not ready to use, changes are made to the non-target system KEM to have the required functionality.

Incremental debt consists of "hundreds or thousands of small shortcuts" [23]. As the system grows older, incremental debt would cause increasingly costly maintenance [7]. For example, it appeared in several projects that a temporary solution was selected without a plan to implement target solution, because the team believed that it would be too simple to build a case for it. Another phenomenon in the case study is that these small shortcuts were not regarded as IT debt by the IT departments, which is quite risky behavior.

Inadvertent debt refers to debt accrued due to ignorance and oversight [7]. An example of this kind of debt is IT debt instance nr.39 which has been already described in the previous paragraphs.

5.2.4 Definition of IT Debt

Another important finding from the case study is on the definition of IT debt. Gartner defines IT debt as “the maintenance backlog to bring an organization's application portfolio up to a fully supported current release state. [1]”. But seen from the case study, this definition only covers half of the problem and thus is not considered to be complete. It emphasizes on upgrading the purchased software at current release. This definition only addresses the problem what is usually called technology refresh or technology gap, and it misses the in-house-built applications. And it also misses the efforts required to change or redesign the enterprise architecture. As new applications are added or old applications decoupled, the complexity of the enterprise architecture of the organization is continually increasing. So work needs to be done to reduce this complexity. If this work is not done timely, it will become the organization’s IT debt. Therefore, we have to incorporate this aspect into the definition of IT debt, which will be as follows: the cost to bring the corporate applications portfolio to a target state. The definition not only includes the efforts needed to upgrade existing application portfolio in the backlog, it also implies the upgrading of newly built applications in the future.

CHAPTER 6 – CONSOLIDATED MODEL OF IT DEBT

In the previous two chapters, a Multivocal Literature Review and a Case Study was conducted to explore the nature of IT debt. This chapter aims to build a consolidated model of IT debt based on the MLR results and Case study results in the previous two chapters. The three aspects of IT debt, namely precedents, dimensions and classifications are compared and used as supplement to each other while constructing the consolidated model.

Table 6 presents the precedents of IT debt identified in MLR (left column) and precedents of IT debt identified in the case study (right column). Due to the fact that the precedents are extracted from different sources, in some of the instances different wording are used to describe the same IT debt precedent. Thus we need to map the Case Study findings to the MLR findings. Each precedent in the right column are selected and compared with the precedents in the left column. If it already exists in the left column, it will be skipped and we will move on to map the next IT debt precedent that is listed in the right column. And in this situation, a decision was made to continue to use the terms on the left, because these terms are used more frequently by practitioners and terms in the right column are only used in the bank.

MLR Findings	Case Study Findings
Limited budget	Target Solution not available
Absence of structured review	Tight timeline
Short-sighted view	Minimize Cost
Imposing deadlines	No Future State Guidelines
Outsourcing	Limited prior knowledge
Ignorance	Out of scope
	Attitude

Table 6. Comparison of Precedents Found in MLR and Case Study

Target Solution not available is caused due to both the external dependency on the software solution vendors and internal dependency on internal stakeholders. The external dependency looks similar to Outsourcing in the left column, while in reality they are still different in nature. For example, if company A purchases software products from company B, then A is dependent on B to provide the latest update of the software. And in this situation, outsourcing is not the right word to be used. Therefore, Dependency is added as a new precedent to the preliminary model. The next two precedents in the right column, Tight timeline and Minimize cost can be obviously mapped to the Imposing deadline and Limited budget in the left column. The next precedent in the right column, No Future State Guidelines, can be mapped into Dependency category which is just added to the left column. In such situations, the organization is dependent on the decision from other organizations or departments, and quite commonly from senior management on the future strategy. Several IT debt instances in the bank were accrued because of Limited Prior Knowledge of the context. It was written in the project document that “we do not know that in the beginning”. This precedent can be related with Ignorance. But the meaning of Ignorance is broader. It also includes the inability to solve the problem. The next precedent Out of Scope is not found either during the MLR findings and is thus added to the IT debt model. Attitude can be seen as personal attitude and group attitude. Personal attitude can be mapped to Short-Sighted View. And group attitude can also be seen as the working culture of a company, both of which are not listed in the MLR findings. So the working culture is added to the IT debt model. Now we have mapped all precedents in the right column to the left ones.

The research findings for both other two aspects of IT debt, Dimensions and Categorization are the same in MLR and Case Study Findings. This also saved the efforts to compare and assembly. Taking all discussions above into account, Table 7 below presents the consolidated model of IT debt based on the MLR and validated by the case study. In the next chapter, this model will be used as an input to answer the research question raised in the beginning of this thesis.

IT debt definition		
The cost to bring the corporate applications portfolio to a target state.		
Dependency	Maintenance & Upgrade	Strategic
Working Culture	Enterprise Architecture	Tactical
Limited budget	Knowledge Distribution	Incremental
Absence of structured review		Inadvertent
Short-sighted view		
Imposing deadlines		
Outsourcing		
Ignorance		
Precedents	Dimensions	Classification

Table 7. Consolidated Model of IT Debt

CHAPTER 7 – DISCUSSIONS

In this chapter we will answer the research question raised in chapter 1 by comparing the two models we got from chapter 3 and chapter 6.

7.1 Answer to Problem Statement and Research Question

Table 8 shows the comparison of IT Debt and Technical Debt from different perspectives. IT debt on the left is the consolidated model we got from chapter 6. The Technical debt on the right side is a part of the technical debt conceptual framework developed by Tom [7] as shown in Figure 5. Irrelevant aspects of the technical debt model are not listed in this thesis. For the readers who would like to know more about this technical debt conceptual model, a detailed description of the TD framework could be found in [7].

	IT debt	Technical debt
Definition	The cost to bring the corporate applications portfolio to a target state.	Any side of the current software system that is considered sub-optimal from a technical perspective [19].
Precedents	Dependency Working Culture Limited budget Absence of structured review Short-sighted view Imposing deadlines Outsourcing Ignorance	Pragmatism Prioritization Process Attitudes Ignorance & Oversight
Dimensions	Maintenance & Upgrade Enterprise Architecture Knowledge Distribution	Code Design & Architecture Environment Knowledge Distribution Testing

Tabel 8. Comparison of IT Debt and Technical Debt

Due to the fact that IT Debt and Technical Debt both utilize the debt metaphor to communicate the idea to non-technical people and they both relates to software applications, it is not surprising that they have some attributes in common. For example, their definitions are quite similar, which is the deviation between the as-is situation and the to-be situation. The difference is that technical debt is about the software system from a technical perspective, and

IT debt is about corporate application portfolios, which not only include software systems but also software-based services.

In comparing the precedents in the two models, some of the precedents of IT debt can be projected to the Technical debt model. Some of them are identical, such as Working culture (attitude) and Ignorance. Imposing deadlines could be seen as prioritization considering the fact that decision makers have prioritized delivery or availability over quality. Short-sighted view and Limited budget can also be regarded as Prioritization. Whereas decision makers in the former situation prioritize short term gains over the long term sustainability, and in the later situation decisions are made to prioritize the budget than the functionality. Limited budget can also be made from a pragmatic decision maker who is more willing to get maximized output from minimum investments. Absence of structured review can be related to the process category in the technical debt model, as structured review can be seen as a part of the project processes. Outsourcing and dependency are two unique attributes of IT debt which cannot be mapped to any of the precedents of Technical debt.

In comparing the dimensions, knowledge distribution debt exists in both IT debt and Technical debt. The other dimensions of Technical debt are all related to software developing processes, such as coding and testing. Design & Architectural debt in Technical debt is related to the design of software architecture and environmental debt is related to the infrastructure on which software is implemented. All of these dimensions of technical debt are all defined on the software level. And architectural debt in IT debt is referring to enterprise architecture, which actually includes both software architectural debt and environmental debt in the technical debt model, as well as the connections between the different software applications. Since IT debt is more related to enterprise architecture, it will not be examined at levels of granularity down to coding and testing. That explains why coding or testing debt is not a dimension of the IT debt.

From the comparison of the dimensions we can see that the main difference of IT debt and technical debt exists in that they are looking at different processes from different perspectives. Technical debt is examined from a software engineering viewpoint on a software construction level. IT debt is accrued in the process of enterprise application rationalization, and it refers to a broader enterprise level than Technical debt. To make the point more explicit, an analogue example of building a house will be given to clarify the two concepts. When building a house, architects and urban planners have different perspectives. Architects are responsible for ensuring the functionality and appearance of a single house. Their roles are similar to software developers who are responsible for the developing of a single software. While urban planners have to formulate a plan to optimize the effectiveness of a community's land use and infrastructure. Their roles are similar to enterprise architects who are responsible for enterprise application rationalizations at higher level.

With all the analysis above, the key research question raised at the beginning of this research paper can be addressed in short as follows:

- Is there a real difference between Technical debt and IT debt?
  Yes, there is indeed a difference between Technical debt and IT Debt. And the significant difference between these two lies in the level of abstraction. Whereas technical debt pertains to the software architecture of a single information system, IT debt is about the debt accrued in the enterprise-wide IT architecture.

7.2 Limitations

Given the limited time allowed and limited resources available, this thesis research has a number of limitations that need to be taken into account.

Although this study tries to incorporate as much available data as possible, only project data from 9 months were available for this analysis at the time when the case study was conducted. More project data inputs will contribute to the reliability of the proposed model. Data inputs from different industries will also help improve the completeness of the model.

During the MLR process, only “IT debt” is used as the keyword to search for articles. There is possibility that other terminology are used to refer to the IT debt phenomenon. But since one of the objectives is to explore how IT debt is perceived, i.e. what are people really referring to when talking about “IT debt”, other terminology won’t help us achieving this goal. Although there might be more attributes found if we searched other terminologies which might be used for IT debt, such as technology gap.

The other limitation is that the consolidated model proposed in chapter 6 is not validated independently. If combined with either surveys or interviews or other research methods with academics and practitioners, this research would be more convincing.

7.3 Future Research

Researchers could use the proposed model in this research as a starting point to focus on more rigorous validation and testing of the proposed model with more data inputs from other companies or industries, and develop this model to a more generic model that includes all aspects of IT debt. Or to be more ambitiously, researchers could try to corporate the IT debt model and technical debt model into one model.

Future research can also start with examining the associations between the precedents, dimensions and classification of IT debt. While in reality it would be worthwhile to associate these aspects with each other, for example, if a certain precedent will always lead to a certain dimension of IT debt or a certain consequence is always accompanied by some certain type of precedents. Besides, the outcomes of IT debt are left out of scope for this research due to limitation of time. It should be added to the model in the future to make the model complete. Researchers could also focus on designing metrics to evaluate the outcome of IT debt.

As stated above, this research covers only the first two steps of the problem solving cycle, which is the problem recognition and definition. It is far not enough to solve the IT debt problem. Future research also needs to focus on finding possible strategies or techniques that could assist practitioners managing their IT debt.

REFERENCES

[1] Gartner. (2010, September 23). *Gartner Estimates Global 'IT Debt' to Be $500 Billion This Year, with Potential to Grow to $1 Trillion by 2015* [Online]. Available: http://www.gartner.com/newsroom/id/1439513

[2] Tom, E., Aurum, A., & Vidgen, R. (2013). An exploration of technical debt. Journal of Systems and Software, 86(6), 1498-1516.

[3] Thibodeau, P. (2011). Counting ‘technical debt’. Information Age, 64.

[4] A. Szykarski. (2012, September 13). *Ted Theodorpoulos on Managing Technical Debt Successfully* [Online]. Available: http://www.ontechnicaldebt.com/blog/ted-theodoropoulos-on-managing-technical-debt-successfully/

[5] T. Theodoropoulos. (2010, November 12). *Technical Debt – Part 1: Definition* [Online]. Available: http://blog.acrowire.com/technical-debt/technical-debt-part-1-definition/

[6] D. P. Kalm. (2013, April). *TECHNICAL DEBT – WHAT IS IT COSTING YOUR COMPANY?* [Online]. Available: http://www.cmfirstgroup.com/wp-content/uploads/2013/04/Tech-Debt-WP.pdf

[7] G. Murphy. (2011, May 20). *How IT Debt is Crippling the Enterprise* [Online]. Available: http://www.enterprisecioforum.com/en/blogs/genefa-murphy/how-it-debt-crippling-enterprise

[8] M. Neuer. *What Is IT Debt And How Is It Being Addressed?* [Online]. Available: http://architectureandgovernance.com/content/what-it-debt-and-how-it-being-addressed

[9] I. Gat. (2010, October 3). *The Real Cost of One Trillion Dollars in IT Debt: Part I – Value Generation and Recapture* [Online]. Available: http://theagileexecutive.com/2010/10/03/the-real-cost-of-one-trillion-dollars-in-it-debt-part-i-value-generation-and-recapture/

[10] D. Britton. (2013, September 27). *Dealing with IT Debt – The Lights are on (Part 2).* [Online]. Available: http://blog.microfocus.com/research/dealing-with-it-debt-the-lights-are-on-part-2/2456/

[11] G. Sehringer. (2013, December 16). *4 Tips for CIOs to Help Solve their Organization's IT Debt Crisis* [Online]. Available: http://sandhill.com/article/4-tips-for-cios-to-help-solve-their-organizations-it-debt-crisis/

[12] Sternberg, R. J. (1986). Critical Thinking: Its Nature, Measurement, and Improvement.

[13] Hayes, J. R. (1989). The Complete Problem Solver (2nd ed.). Hillsdale, N.J.: Lawrence Erlbaum Associates

[14] Bransford, J.D. & Stein, B.S. (1993). The Ideal Problem Solver (2nd Ed). New York: Freeman

[15] Ogawa, R. T., & Malen, B. (1991). Towards rigor in reviews of multivocal literatures: Applying the exploratory case study method. *Review of Educational Research*, *61*(3), 265-286.

[16] Yin, R. K. (1994). Case study research: Design and methods. Sage publications.

[17] Cunningham, W. (1992, December). The WyCash portfolio management system. In *ACM SIGPLAN OOPS Messenger* (Vol. 4, No. 2, pp. 29-30). ACM.

[18] Brown, N., Cai, Y., Guo, Y., Kazman, R., Kim, M., Kruchten, P., ... & Zazworka, N. (2010, November). Managing technical debt in software-reliant systems. In *Proceedings of the FSE/SDP workshop on Future of software engineering research* (pp. 47-52). ACM.

[19] Ktata, O., & Lévesque, G. (2010, May). Designing and Implementing a Measurement Program for Scrum Teams: What do agile developers really need and want?. In *Proceedings of the Third C* Conference on Computer Science and Software Engineering* (pp. 101-107). ACM.

[20] Allman, E. (2012). Managing technical debt. *Communications of the ACM*, *55*(5), 50-55.

[21] SQALE official site. [Online]. Available: http://www.sqale.org/

[22] Guo, Y., & Seaman, C. (2011, May). A portfolio approach to technical debt management. In Proceedings of the 2nd Workshop on Managing Technical Debt (pp. 31-34). ACM.

[23] S. McConnell. (2007, November 1). *Technical debt*. [Online]. Available: http://www.construx.com/10x_Software_Development/Technical_Debt/

[24] M. Fowler. (2009, October 14). *Technical Debt Quadrant.* [Online]. Available: http://martinfowler.com/bliki/TechnicalDebtQuadrant.html

[25] J. Carter. (2013, January). *IT Implementation Debt* [Online]. Available: http://www.iconsolutions.com/wp-content/uploads/2013/01/IT-Implementation-Debt.pdf

[26] Micro Focus, *Mainframe Transformation – The Elephant in the Room* [Online]. Available: http://www.microfocus.com/_ex/External/press/12/Mainframe-Transformation-The-Elephant-in-the-Room.pdf

[27] G. Sehringer (2013, November 14). *The One Idea Radical Enough to Solve the Trillion-Dollar IT Debt Crisis* [Online]. Available: http://www.mendix.com/think-tank/the-trillion-dollar-it-debt-crisis/#more-22363

[28] D. Britton. (2013, September 27). *Dealing with IT Debt – The Lights are on (Part 2)* [Online]. Available: http://blog.microfocus.com/research/dealing-with-it-debt-the-lights-are-on-part-2/2456/

[29] E. Withers. (2013, April 9). I.T. Debt – Ignoring It Won't Make It Go Away [Online]. Available: http://www.iconsolutions.com/technical-debt/

[30] V. Mirchandani. (2010, September 26). *Gartner's "IT Debt" Scare* [Online]. Available: http://dealarchitect.typepad.com/deal_architect/2010/09/gartners-it-debt-scare.html

[31] B. Snyder. *Deferred IT maintenance is a ticking time bomb* [Online]. Available: http://www.infoworld.com/t/it-management/what-you-missed-deferred-it-maintenance-ticking-time-bomb-898

[32] Micro Focus (2014, March 4). *CIOs Estimate 29% Rise in Mainframe Application IT Debt Over Last 18 Months* [Online]. Available: http://www.marketwired.com/press-release/cios-estimate-29-rise-in-mainframe-application-it-debt-over-last-18-months-lse-mcro-1884905.htm

[33] Shull, F. (2011). Perfectionists in a world of finite resources. Software, IEEE, 28(2), 4-6.

[34] A. Szynkarski. (2012, July 30). *Steve McConnell: How to Categorize & Communicate Technical Debt* [Online]. Available: http://www.ontechnicaldebt.com/blog/steve-mcconnell-on-categorizing-managing-technical-debt/

APPENDIX I Full List of IT Debt Instances

IT debt nr.	Description issue	Reason	Type of Debt	Classification	Dimensions
1	The new BAN regulations on connecting the back-end systems are still unpublished at the moment. Use of the existing ACBS screens for the new CPA application might don't comply with new regulations. But it is simpler and cheaper to continue to use the existing solution and adapt to new regulation when it is published.	No Future State Guidelines	Existing	Tactical	Enterprise Architecture
2	A service oriented cross-domain communication is not implemented because it falls in the scope of Finance department.	Out of scope	Existing	Inadvertent	Enterprise Architecture
3	The new CPA application is implemented on a non-target system because it is more economical to do so.	Minimize Cost	Existing	Inadvertent	Enterprise Architecture
4	Communication between two modules are based on JAVA solutions while via Websphere Message Broker is positioned to be strategic. The reason is that the capability of WMB is still under investigation.	Limited prior knowledge	New	Incremental	Enterprise Architecture
5	Fraude-detection, Eurokas, BaNCS en Sealbag are declared out-of-scope on the basis of Bare minimum approach, which targets to minimum amount of budget.	Minimize Cost	Existing	Inadvertent	Maintenance & Upgrade
6	Billing of the Safehuur application is done by the Safehuur application itself and not with the GT application, because GT is not able to bill on external accounts. However, GT should have the ability to charge external accounts in the future state.	Target Solution not available	Existing	Inadvertent	Enterprise Architecture
7	115k was invested in customizing the non-target system Peoplesoft to support the sales of fiscal saving products that came along with new Dutch fiscal rulings.	Tight timeline	New	Strategic	Maintenance & Upgrade
8	Normally SAS authenticates via the SAS Server on Unix. But the ban	Target Solution	Existing	Incremental	Enterprise Architecture

	currently does not have MSEC User Security available on Unix. Therefore in phase 1 it was decided to do authentication via POSA.	not available			
9	Communication between SAS components (Engine and Dataflux tiers) takes place via JDBC protocol. SAS has no solution for securing this connection.	Target Solution not available	New	Incremental	Enterprise Architecture
10	The system is implemented on AIX platform, which is non-target in the long term, because the Linux HVE proposition is not mature yet.	Limited prior knowledge	Existing	Incremental	Enterprise Architecture
11	Because the architecture for back-end were not available, it was not possible to work on the interfaces between Accounts/back-end. The interface has to be changed when the architecture is available.	No Future State Guidelines	Existing	Tactical	Maintenance & Upgrade
12	The SAA filturing remains on the current Sun server, which is not the target in the future. Migration to the required Linux will take place when the required Linux version mature is.	Target Solution not available	Existing	Incremental	Enterprise Architecture
13	Upgrade the filtering solution in Guernsey from Side Safewatch to FircoSoft SAA is out of scope for this project.	Out of scope	Existing	Inadvertent	Maintenance & Upgrade
14	The GFS backend will be hosted on Linux, the WAS server on AIX and the database on Oracle. But AIX is not a target.	Tight timeline	Existing	Inadvertent	Enterprise Architecture
15	Oracle Grid is required but start can be on HVE Oracle, because of not timely availability of Oracle grid and migration priority. Migration step when LCM of SAA toward Linux in 2014.	Target Solution not available	New	Tactical	Enterprise Architecture
16	MySQL is chosen for data storage, because migration to Oracle requires to rebuild the whole application.	Minimize Cost	New	Tactical	Enterprise Architecture
17	The T24 applications in Brazil, Guernsey en Curacao are declared to be out of scope and continue to use the old interfaces.	Out of scope	Existing	Inadvertent	Maintenance & Upgrade
18	The outdated data warehouse (AV2) which doesn't belong to the long term application portfolio was still used due to the fact that it was the only solution and no other solution was available.	Target Solution not available	Existing	Tactical	Enterprise Architecture
19	The requirements as when and where to back up the received/sent fax	No Future State	Existing	Tactical	Maintenance & Upgrade

	were unknown. A temporary solution was adopted to back up the fax when the stack is full. It has to switch to the target solution when the requirements are made clear.	Guidelines			
20	The application requires a separate user ID and password to login instead of using the standard Single Sign On technique, because SSO is not applicable at the time of the project.	Target Solution not available	Existing	Tactical	Maintenance & Upgrade
21	A temporary solution was adopted when replacing the Equens system by a new system. A new project will be launched to solve this.	Tight timeline	New	Tactical	Maintenance & Upgrade
22	Only during the implementation of the project, a conflict between the two involved systems were identified. So a temporary solution was adopted to prevent improper backlog of reports.	Limited prior knowledge	New	Inadvertent	Maintenance & Upgrade
23	Integration of APT-calls in Kosmos Germany will incur redundant data. Increment 3 of this project will move this redundant code to a separate component to remove IT debt.	Minimize Cost	New	Tactical	Maintenance & Upgrade
24	Websphere is not supported on Windows, so Tomcat/Wintel is chosen as Web Hosting	Minimize Cost	Existing	Inadvertent	Enterprise Architecture
25	Data management and data analytics are done on the same SAS environment, which might lead to poor data quality. An MS Access database is created to store data separate from the golden source.	Minimize Cost	Existing	Inadvertent	Maintenance & Upgrade
26	Using DTS to consolidate VAT figures is only a temporary solution. Because the target solution using FRAAI is still in development.	Target Solution not available	Existing	Incremental	Enterprise Architecture
27	The system is implemented on the non-target system AIX.	Minimize Cost	Existing	Tactical	Maintenance & Upgrade
28	Changes are made to the KEM system which is non-target. Because penalty from Authorities involved with poor data quality is severe.	Tight timeline	New	Tactical	Maintenance & Upgrade
29	Single Sign On is not used in this project because the currently available authentication mechanism doesn’t support this; the gap will be closed when an generic integrated SSO environment is available	Target Solution not available	Existing	Tactical	Enterprise Architecture

30	The old version of WAS will be used and the target version of WAS will be implemented later together with other teams who are also using the old version of WAS and have already planned for upgrade.	Target Solution not available	New	Tactical	Enterprise Architecture
31	TFIM was temporarily chosen as Secure Token Service, which might need to replaced when the bank have chosen the target.	Target Solution not available	New	Incremental	Enterprise Architecture
32	The target solution WMB is not used in phase 1. It is more economical to do so because in phase 1quantity of work for WMB is little and the project was already overburdened with new techniques.	Minimize Cost	Existing	Tactical	Maintenance & Upgrade
33	This is a compulsory project required by Dutch authorities. Due to the time pressure, a temporary solution is used which requires manual input to Excel from source systems	Tight timeline	Existing	Tactical	Enterprise Architecture
34	The People Finder application uses Planon platform. But the bank haven't chosen the target platform yet. Re-platforming is required.	No Future State Guidelines	New	Inadvertent	Enterprise Architecture
35	The use of Teradata deviates from the proposed but not yet approved enterprise standards. Adapting to the target will be discussed later.	No Future State Guidelines	Existing	Incremental	Enterprise Architecture
36	A temporary solution of using AIX instead of MSec is adopted due to the urgency to have the solution in place as soon as possible.	Tight timeline	New	Tactical	Enterprise Architecture
37	It was decided not to use SAS Enterprise Minder and Model Management, which might be in use later, due to the project urgency.	Tight timeline	New	Tactical	Enterprise Architecture
38	A temporary intraday reporting solution is adopted in this case due to the fact that otherwise it would not be possible to finish in time.	Tight timeline	New	Tactical	Maintenance & Upgrade
39	The Temenos application is not fully utilized. In the beginning it was not clear what the solution delivery of Temenos was. So a temporary solution was chosen.	Limited prior knowledge	New	Inadvertent	Maintenance & Upgrade
40	Due to the time constraints, the ability of the account administration to deliver a complete account statement is compromised.	Tight timeline	New	Tactical	Maintenance & Upgrade
41	The solution increment 1 creates is just temporary because the ban cannot afford the consequences of not finishing the project in time; It	Tight timeline	New	Strategic	Enterprise Architecture

	will be replaced by target solution in the second phase.				
42	The non-target system KEM was in use and changes were still made to it. Due to the pressure from the Dutch authorities, it is not possible to implement the required solution in time.	Tight timeline	New	Tactical	Maintenance & Upgrade
43	The non-target system Cool/Gen was still in use. Because there is no time to find other solutions.	Tight timeline	New	Tactical	Maintenance & Upgrade
44	AV2 instead of the standard data warehouse was used as the source for reconciling data because no other solution is possible. Decommissioning of AV2 is required later.	Target Solution not available	Existing	Incremental	Enterprise Architecture
45	The application in this project will be implemented on a nonstandard platform since the standard platform on Linux was not available yet. Re-platforming is required to change to target solution..	Target Solution not available	Existing	Incremental	Enterprise Architecture
46	The output file is in Word format, which is not a standard in the bank.	Minimize Cost	New	Inadvertent	Enterprise Architecture
47	In this project, changes were still made to the Essentis application which is to be decommissioned in 2014.	Tight timeline	New	Tactical	Maintenance & Upgrade
48	The TJ application is non-target and a new application will be implemented to replace it. But TJ is still not decommissioned.	Minimize Cost	Existing	Tactical	Maintenance & Upgrade
49	Virtual servers provide insufficient performance to enable for batch processing within the available batch windows, so as a temporary solution physical servers were used. Redesign is required.	Target Solution not available	New	Incremental	Enterprise Architecture
50	Reports are generated with Business Objects which is included in the BWise package (already in use for other processes). But in the target situation standard reporting tools are being used.	Minimize Cost	Existing	Tactical	Maintenance & Upgrade

APPENDIX II Full List of Articles for MLR

Nr.	Title	URL
1	Gartner Estimates Global 'IT Debt' to Be $500 Billion This Year, with Potential to Grow to $1 Trillion by 2015	http://www.gartner.com/newsroom/id/1439513
2	Just What Is IT Debt?	http://redmondmag.com/articles/2010/09/24/just-what-is-it-debt.aspx?admgarea=BDNA
3	What Is IT Debt And How Is It Being Addressed?	http://architectureandgovernance.com/content/what-it-debt-and-how-it-being-addressed
4	How IT Debt is Crippling the Enterprise	http://www.enterprisecioforum.com/en/blogs/genefa-murphy/how-it-debt-crippling-enterprise
5	Dealing with IT Debt – The Lights are on (Part 2)	http://blog.microfocus.com/research/dealing-with-it-debt-the-lights-are-on-part-2/2456/
6	The One Idea Radical Enough to Solve the Trillion-Dollar IT Debt Crisis	http://www.mendix.com/tag/it-debt/
7	Is there really a global 'IT debt' of $500 billion?	http://www.zdnet.com/blog/btl/is-there-really-a-global-it-debt-of-500-billion/39580
8	Gartner's "IT Debt" Scare	http://dealarchitect.typepad.com/deal_architect/2010/09/gartners-it-debt-scare.html
9	TECHNICAL DEBT – WHAT IS IT COSTING YOUR COMPANY?	http://www.cmfirstgroup.com/wp-content/uploads/2013/04/Tech-Debt-WP.pdf
10		http://blog.acrowire.com/technical-debt/technical-debt-part-1-definition/
11	IT Debt and the Case for Application Overhaul	http://thejournal.com/articles/2010/09/23/it-debt-and-the-case-for-application-overhaul.aspx
12	Gartner: Global 'IT debt' hits $500 billion, on the way to $1 trillion	http://www.networkworld.com/news/2010/092310-global-it-debt.html
13	CIOs Estimate 29% Rise in Mainframe Application IT Debt Over Last 18 Months	http://www.marketwired.com/press-release/cios-estimate-29-rise-in-mainframe-application-it-debt-over-last-18-months-lse-mcro-1884905.htm
14	IT debt, Inflated Expectations and Software Engineering	http://agilitator.com/blog/?p=1093
15	use mobile development projects as a means to re-energise	http://www.al.co.za/index.php/ITweb/Business/avoiding-increasing-it-debt-by-preparing-for-mobile-development
16	Walk Away From Your IT Debt With SaaS	http://www.enterpriseirregulars.com/26191/walk-away-from-your-it-debt-with-saas/

17	The Other Debt Crisis	http://www.enterpriseirregulars.com/43755/the-other-debt-crisis/
18	Growing Global IT Debt Hits $500 Billion, Reports Gartner	http://www.techweekeurope.co.uk/news/growing-global-it-debt-hits-500-billion-reports-gartner-10030
19	Takes Two To Tango	http://www.iconsolutions.com/takes-two-to-tango/
20	IT Implementation Debt	http://www.iconsolutions.com/wp-content/uploads/2013/01/IT-Implementation-Debt.pdf
21	The Canary Just Died – Thoughts On Outdated Technology And Another Banking Crisis	http://www.iconsolutions.com/saving-the-canary-thoughts-on-outdated-technology-and-another-banking-crisis/
22	I.T. Debt – Ignoring It Won't Make It Go Away	http://www.iconsolutions.com/technical-debt/
23	Y2K vis-a-vis IT Debt	http://theagileexecutive.com/2010/10/18/y2k-vis-a-vis-it-debt/
24	The Real Cost of One Trillion Dollars in IT Debt: Part I – Value Generation and Recapture	http://theagileexecutive.com/2010/10/03/the-real-cost-of-one-trillion-dollars-in-it-debt-part-i-value-generation-and-recapture/
25	The Real Cost of One Trillion Dollars in IT Debt: Part II – The Performance Paradox	http://theagileexecutive.com/2010/10/06/the-real-cost-of-one-trillion-dollars-in-it-debt-part-ii-the-performance-paradox/
26		http://www.processor.com/editorial/article.asp?article=articles%2Fp3222%2F44p22%2F44p22%2F44p22.asp
27	Global IT Debt: A "New" Buzz Term For IT/ERP Vendor Avarice	http://procureinsights.wordpress.com/2010/12/17/global-it-debt-a-new-buzz-term-for-iterp-vendor-avarice/
28	Avoiding increasing IT debt by preparing for mobile development	http://www.itweb.co.za/index.php?option=com_content&view=article&id=54233
29	GARTNER: GLOBAL IT 'DEBT' $500 BILLION	http://www.ukfast.co.uk/business-news/gartner-global-it-debt-500-billion.html
30	$500 Billion IT Debt For Deferred Maintenance	http://www.galorath.com/wp/500-billion-it-debt-for-deferred-maintenance.php
31	IT leaders unable to manage or measure IT debt	http://networksasia.net/article/it-leaders-unable-manage-or-measure-it-debt-1340068140
32	IT debt costing businesses billions	http://www.siliconrepublic.com/business/item/17896-it-debt-costing-businesses
33	CIOs' 2014 Priority: Reduce 'IT debts'	http://www.idgconnect.com/blog-abstract/5249/cios-2014-priority-reduce-it-debts
34	Global IT Debt Hits $500 Billion on Its Way to $1 Trillion: Gartner	http://www.eweek.com/c/a/IT-Infrastructure/Global-IT-Debt-Hits-500-Billion-On-Its-Way-to-1-Trillion-Gartner-245440/
35	Deferred IT maintenance is a ticking time bomb	http://www.infoworld.com/t/it-management/what-you-missed-deferred-it-maintenance-ticking-time-bomb-898
36	The One Idea Radical Enough to Solve the	http://www.mendix.com/think-tank/the-trillion-dollar-it-debt-crisis/

	Trillion-Dollar IT Debt Crisis	
37	The challenges of using mainframes	http://www.microfocus.com/_ex/External/Mainframe/Racepoint-Micro-Focus-mainframes-commentary-presentation.pdf
38	Mainframe Transformation – The Elephant in the Room	http://www.microfocus.com/_ex/External/press/12/Mainframe-Transformation-The-Elephant-in-the-Room.pdf
39	Global IT debt reaches $500 billion	http://itvoir.com/portal/news/News/Global-IT-debt-reaches-500-billion-17-018.asp
40	IT Debt and the Case for Application Overhaul	http://www.enduserexperience.info/articles/share/57507/
41	Application Overhaul – Top Ten Best Practices for Cutting Your IT Debt	http://blog.microfocus.com/news/application-overhaul-top-ten-best-practices-for-cutting-your-it-debt-2/681/
42	Global 'IT debt' hits $500 billion, on the way to $1 trillion	http://cw.com.hk/news/global-it-debt-hits-500-billion-way-1-trillion
43	Can the cloud bail enterprises out of IT debt?	http://www.zdnet.com/blog/btl/can-the-cloud-bail-enterprises-out-of-it-debt/61122
44	Pressure Is On IBM To Forgive Millions In IT Debt	http://news.slashdot.org/story/07/06/18/206258/pressure-is-on-ibm-to-forgive-millions-in-it-debt
45	IT DEBT INCREASING, FINDS SURVEY	http://www.cfoinnovation.com/content/it-debt-increasing-finds-survey
46	Quantifying Technical Debt: Beware of Your Assumptions	http://blog.castsoftware.com/quantifying-technical-debt-beware-of-your-assumptions/
47	The lights are on, but no-one's home……	http://blog.microfocus.com/compliance/the-lights-are-on-but-no-ones-home/2409/