Managing Technical Debt in Practice: An Industrial Report

Clauirton A. Siebra
Center of Informatics
Federal University of Paraiba
Joao Pessoa – PB – Brazil
+55 83 3216 7093
cas@di.ufpb.br

Graziela S. Tonin, Fabio Q. B. da Silva, Rebeka G. Oliveira, Antonio L. C. Junior, Regina C. G. Miranda, Andre L. M. Santos
CIn/Samsung Project – Federal University of Pernambuco
+55 81 3454 3300
{gst,fabio,rgo2,alocj,rcgm,alms}@cin.ufpe.br

ABSTRACT

The Technical Debt (TD) metaphor has been used as a way to manage and communicate long-term consequences that some decisions may cause. However the state of the art in TD has not culminated yet in rigorous analysis models for large-scale projects. This work analyses an industrial project, from the perspective of its decisions and related events, so that we can better characterize the existence of TD and show the evolution of its parameters. The project in study had a life cycle of six years (2005-2011) and its data for analysis was collected from emails, documents, CVS logs, code files and interviews with developers and project managers. From this analysis, we identified the factors that had influence on the project decisions and their impact on the system along the time. Furthermore, we were able to extract a set of lessons associated with the characterization of TD in projects of this port.

Categories and Subject Descriptors

D.2.9 [**Management**]: Cost estimation

General Terms

Management, Measurement, Documentation.

Keywords

Technical debt, maintenance, software evolution.

1. INTRODUCTION

The development process of large software systems certainly requires several types of project decisions, which affect future development activities. The use of the Technical Debt (TD) metaphor [1] is well suitable to support this decision process. In fact, TD provides both abilities of monitoring the evolution of decisions effects and representing low level development aspects in a more comprehensive and straighter language for stakeholders. Thus, TD could be seen as, or has the potential to be, a powerful tool for management and communication activities. Unfortunately, the state of the art in TD has not culminated yet in rigorous analysis models, mainly for large-scale projects, and we have a substantial lack in formal theories about it [2].

Considering this context, our research group is leading an exploratory investigation to identify sources of technical debt and understand its evolution. The object of study is the SMB (Samsung Mobile Business) system, a commercial mobile software application that runs on different Samsung mobile phone platforms and woks as a client for Microsoft Exchange™ e-mail service (MS Exchange). SMB has a total of 63.218 logic code lines, according to the Unified Code Count tool. The SMB system was developed in J2ME language, using RUP as development methodology, and translated to 17 languages to be released in the European, South American and Asian markets. This application was started at 2005 and was maintained during 6 years. Its first official release was in March 2006 and the system was finished in March 2011. Consequently, there is a considerable amount of documents and code versions. The development was carried out by a consortium stabilized between Samsung Brazil and Informatics Center, at the Federal University of Pernambuco, and involved an average of 20 specialists per month. An important feature of the SMB development was its number of requirement modifications, regarding its adaptation to the very dynamic mobile technology and market. As this application was constantly modified, the process of maintenance and evolution presented several difficulties. We argue that a TD theory could have been used to attenuate such difficulties. This was the main Samsung's motivation to invest in this exploratory study, so that its results could be used in future projects.

2. SCENARIOS

We could identify three main scenarios for our investigation. Such scenarios are summarized as follows.

2.1 Scenario 1: Persistence Layer

The first scenario is associated with the persistence layer. The SMB version 1.0 was based on a client-server architecture and its persistence layer was very simple and defined to only support functions associated with the management of emails, calendar and contacts. This approach was latterly changed to a version without a server component (SMB V1.1), which employed the Webdav protocol due to its simplicity and support to Microsoft Exchange 2003. In this decision moment, the team was aware that Microsoft was going to release a new version of the Microsoft Exchange, called Microsoft Exchange 2007.

The SMB V1.1 presented serious persistence problems, such as low efficiency, adaptability and bad use of memory. Then, a specialist in persistence was allocated in the project and a new architecture to persistence was planned on. Meanwhile, Microsoft informed that the release of *Microsoft Exchange 2007* was going to be on 30th November 2006 and that this new version was not going to support the Webdav protocol. The decision to still using the Webdav protocol was maintained and the persistence layer was almost completely reimplemented. In this new version (V1.2), the communication and protocol layers presented a strong logical coupling. In November 2007, the team decided to change the protocol, once the market was already using the Microsoft Exchange 2007. The ActiveSync protocol was chosen to be used. However, due to the strong logical coupling between the persistence and communication layers, a total refactoring was required in the persistence Layer.

2.2 Scenario 2: GuiFramework

In V1.1, the team decided to develop its own GUI (Graphic Unit Interface) framework, which did not consider the touch functionalities because the team did not believe that the touch technology could be so popular at a short term. However, in November 2008, the team identified a higher probability about the future use of the touch technology, based mainly on trends of the market. A first porting to a touch device was planned and the touch project was started on 24/11/2008. However, such development branch was cancelled three days later due to a mandatory decision of the client. In August 2009, the client sent a new request to porting the system to a touch screen device. According to evidences from emails sent back to the client, three alternatives were evaluated to implement the touch screen support: via fixe virtual command bar (simplest), via implementation of a touch engine in the GUI framework (hardest) and via a hybrid solution. The team decided for the first option as an urgent solution and after the release, they could work in a complete solution to the touch technology. However the client declined this decision and requested the implementation of the hybrid solution. The final version with this feature was released in January 2010 and it was discontinued in the beginning of 2011.

2.3 Scenario 3: New Language Inclusion

In some moment of the development (April 2007), the inclusion of a language became a hard task, so that the team considered to create an automation solution for this task. However, as they believed in a low probability to use other languages in the application, a simple manual process was implemented. In December 2009, the client requested the translation of the application to 7 European languages (Germany, French, Italian, Dutch, Hungarian, Serbian and Slovakian). According to the interviews, at this moment the team had again considered the implementation of a complete automation solution. However this was not carried out due to time constraints, once the team should prepare versions of the application with such languages. In February 2010, the client informed the need of translating the application to Turkish and that new translations were going to be required, once the application was going to be delivered in the Asian Southeast and Medium Orient markets. After that, the team received the request to translate the application to 6 other languages: Farsi, Chinese, Thai, Vietnamese, Malaysian and Indonesian. In February 2010, the automation of new languages inclusion was implemented. Thus, the process of including a new language, which used to spend 2 days with one allocated resource, could be carried in 2 hours.

3. Data Collection and Analysis

The activity of collecting information was based on the parameters presented in the Cunningham metaphor [1]. In this way, concepts such as debt, interests and benefits were mapped to more concrete concepts, such as impact in quality, recoding difficulty and refactoring. The data was collected in a systematic way, considering all the identified changes associated with each scenario. The collected information for each change is: facts of influence, motivation and impact in terms of effort (persons per hour). Regarding decisions, we have collected data associated with possible alternatives, motivations, facts of influence, description and decision identifier. The aim of this data collection was to characterize the moment when a decision was taken. This means, the moment when the debt was created, the impact of this decision in the scenario on investigation (in terms of effort) and the benefits that could be motivated such a decision. The main sources of information were emails, development documents and interviews. The data collection was carried out to each scenario and the results are discussed in the next subsections.

3.1 Analysis of Scenario 1

The analysis of Scenario 1 (Persistence Layer) has stressed three main decisions to be characterized:

- Decision 1: implementation of a simple persistence layer, just to support the current demand;
- Decision 2: complete redefinition of the persistence layer, with a strong logical coupling with the communication layer;
- Decision 3: redefinition of the persistence layer, once the previous version could not be used with the new protocol and such layer was strongly coupled with the communication layer.

After the identification of the motivations to each decision, we could trace some conclusions The Decision 1 could be considered correct because, at that moment, the team was not able to predict, in a long term, the way that the application was going to. The Decision 3 could also be considered correct because the marked was already requiring a new communication solution and the Webdav protocol was not supporting such a solution. We have found strong evidences of technical debt in Decision 2. Thus, such decision was detailed as follows:

- Benefit: to evolve the persistence layer as so as possible to V1.2 of the application;
- Debt: accept the risk of not implementing the persistence layer with an architecture more independent of the communication layer, which could simplify future changes;
- Interest: accumulation of efforts, since the moment of the decision of coupling the persistence layer to the communication protocol, until the moment when the debt was paid;
- Payment: effort spent to carry out reimplementations in the persistence layer.

According to evidences, the debt was created in the moment of Decision 2, when the team decided to implement the new persistence with dependent layers. The total of 12 months was required to plan, implement and test the application with this new architecture. Using the documentation (chronograms, backlogs and register of code lines modifications) along this period, we were able to estimate the total effort, in terms of hours per person, to implement Decision 3 at the moment when Decision 2 was taken. The documentation was also important to identify the effort of the architecture reimplementation, associated with Decision 3. Regarding the values, the effort to implement Decision 3 at the moment of Decision 2 (estimation) was 345 h/person, while the effort to implement Decision 3 (real) was 1416 h/person. Thus we have an interest of 1071 h/person.

We concluded that the scenario where the team consciously decided for implementing a persistence layer strongly coupled to the communication layer (Decision 2), even with the information that the protocol was going to be changed in the future, incurred a debt that needed to be paid some months later. This payment is associated with Decision 3 and it had a high cost of 1071 h/person.

3.2 Analysis of Scenario 2

This scenario also presented three main decisions to be characterized:

- Decision 1: implementation of the GUI framework without touch support;
- Decision 2: non implementation of the full touch support at the moment when the requests for SMB porting to touch devices started to appear;
- Decision 3: implementation of just some few touch functions, considering particular features of the device that was going to receive SMB.

The Decision 1 could be considered correct, once the touch technology was not popular in the market. The risk to consider this technology versus the time required to implement such technology could probably result in a significant delay to deliver the SMB. The Decision 2 could also be considered correct, once the application was finished in March 2011 and few portings to touch devices were performed. The Decision 3 could again be considered correct because the team needed a small effort to its implementation and any new porting was carried out to touch devices. In this context, we can better analyze details about saved effort related to Decisions 2 and 3 and prove that such decisions did not incur TD:

- Benefit: (Decision 2) to prioritize the porting process that was being carried out at that moment, once such porting was strategic to the client; and (Decision 3) carry out a faster implementation, so that the touch device could be released in a short time;
- Debt: (Decision 2) accept the risk of not implementing the touch full support, even with a strong trend of the mobile market in using this technology; and (Decision 3) accept the risk of a future GUI framework adaptation, so that it could support a full touch set of functionalities;
- Interest: (Decision 2) effort accumulation, since the moment of the decision of not implementing touch support, until the moment that this implementation was required; and (Decision 3) accumulation, since the moment of the decision of not implementing a full touch support, until the moment when it was required.;
- Payment: (Decision 2) defined as the effort spent to implement the full touch support; and (Decision 3) the effort to adequate the full touch support to a specific device.

At the first moment, when the team considered to implement the full touch support (November 2008), the effort was estimated to 1567 h/person. However this implementation was not carried out and, later on, the client requested the implementation of some few functions to support part of the touch technology to a specific mobile model. According to emails and chronogram, the effort was estimated to 96 h/person. This application was sent to client in January 2010 and discontinued in early 2011. Regarding the values, the effort for implement a full touch support was estimated to 1576 h/person, while the effort to just adapt a specific mobile model was 96 h/person. Then, we had a saved effort of 1480 h/person (185 work days, considering one person)

Thus, the decision of not implementing the full touch support did not incur technical debt and has saved 1480 h/person. This is a type of decision that avoids unnecessary costs and did not incur technical debt because, in this scenario, the system was discontinued before the need of a touch implementation. We did not simulate the situation where the system is not discontinued and the touch support is implemented in the future.

3.3 Analysis of Scenario 3

We have identified three relevant decisions associated with this scenario:

- Decision 1: implementation of a manual approach to include new languages;
- Decision 2: maintenance of the manual approach for inclusion of new languages at the moment when the translation to 7 new languages was requested;
- Decision 3: implementation of an automatic inclusion process at the moment when the translation to 9 new languages was requested.

The Decision 1 could be considered correct, once the system was initially translated only to three languages and the team could not predict the use of the system in so many countries. We have found evidences of TD in Decision 2, while the Decision 3 could also be considered correct, once a significant amount of effort was saved. Thus, we can detail the features of Decision 2 as follows:

- Benefit: saving of effort;
- Debt: accumulation of effort related to the manual language translation/inclusion;
- Interest: effort accumulation, related to new languages that may be included, from the moment when the automation process was not implemented, until the moment that is was implemented;
- Payment: effort to include new languages.

According to Decision 1, the process of including a new language was carried out in a manual way and the team believed in a low probability of new inclusions. In this scenario, the cost to include two languages (English and Spanish) was 32 h/person. According to Decision 2, the team maintained the manual translations. However, 7 new languages were translated with a total cost of 171 h/person. If the automation process had been implemented, the effort should be 14 h/person. Thus, the saved effort could have been about 157 h/person. However we are not considering here the additional effort associated with the implementation of the automation process, which was 40 h/person. Consequently, if we add this cost to the cost of the automated translations and decrease the resultant value from the manual effort, we could have a final saved effort of 117 h/person.

The value of 117 h/person is also the debt associated with Decision 2. In this case, the debt is fixe (interest is null) and it will only increase if new languages are included. Thus, the cost of payment is also fixe. However, if we extend the TD concepts to other aspects, beyond effort analysis, this cost could increase due to the modifications in the value/hour associated with the work of each collaborator, which tends to increase along the time. Thus, the fact of not implementing the automated approach was not a correct decision (Decision 2), incurring technical debt because the team had an additional cost of including 7 manual translations. In other words, an effort of 117 h/person was wasted.

4. DISCUSSION AND LEARNED LESSONS

According to the analysis of our scenarios, the use of decisions appears as an adequate approach to characterize the historical evolution of a technical debt. The principal aspect of this approach was to identify related decisions, so that we could define the timeline of technical debts and monitor their evolution. A deep study of these timelines raised up important lessons that must be considered, from the start, in any other project.

The first lesson is related to the advantages of incurring a debt. Works about TD [3] comment that intentional debt incurs for strategic reasons and list the time to market and preservation of startup capital as main examples. We have observed, in practice, another important reason. When a system is discontinued, all of the system's technical debt is retired with it. Once a system has been taken out of production, there is no difference between a "clean and correct" solution and a "quick and dirty" solution. The Scenario 2 (GuiFramework) represents this situation in a practical manner, when the decision of not implementing the full touch support did not incur technical debt and has saved 1480 h/person. Thus, this decision, which could normally be considered a TD, brought advantages to the project. This experience shows the importance in predicting the discontinued time of modules and systems, so that the decision of paying a TD could be written as a function that considers its start and discontinued time. In fact, the principal reason to monitor TD is to know the best moment to pay the debt, or part of it [4]. However, this decision makes more sense when we know the remainder life time that the system/module still has before its retirement. Unfortunately, this discontinued time is not generally so easy to predict.

The second lesson is related to what we should consider as a source of TD. In the literature, TD is mostly associated with implementation aspects [5]. However, our experience shows the importance to extend this concept to other aspects. For example, consider the Decision 3 of Scenario 1 (Persistence Layer). According to some interviews, the use of a unique specialist could lead the development team to solutions that he/she wants and believes that are correct. This could be dangerous once some problems may not be identified and covered by the experience of this specialist. In this scenario, the use of a second specialist was essential to take the Decision 3. This was a typical case where modifications in the team had influence on the project decision and, consequently, in the technical debt values. Thus, the conscious use of a team that is not ideal could also be considered a case of technical debt.

A third lesson is related to additional facts that should be considered during the monitoring process of a TD. The Scenario 3, for example, shows the importance of considering the valorization of collaborators to calculate the real interest in terms of financial cost. In order, all our studies are limited just to the development cost analysis (development effort) due to access restriction to financial information of the project. We believe that even if we had access to such information, we probably still have the challenge of estimating what could have been the depreciation of product value due to delays in its release, and how much the product could have been appreciated if it had immediately met the customers demand for mobile solutions for MS Exchange 2007 clients. In both cases, external factors such as the occurrence of competing products can significantly affect this value. Another factor that could be difficult to estimate is the return if the extra effort had been invested in new features such as support for touchscreen technology.

The major contribution of our investigation is its practical perspective. This means, these lessons were extracted from a real project and based on several documents, interviews and coding statistics. Finally, several research directions can be derived from this work. Currently we are working in two main directions. First, the development of empirical studies that could support the development of methodologies to monitor technical debt. This could be important, for example, to identify the ideal moment to pay the debt. Second, applications of financial mathematic techniques in technical debt, so that we can define a methodology to demonstrate the financial viability of changes in a project.

5. ACKNOWLEDGEMENTS

The authors would like to thank all the software engineers from CIn/SIDI-Samsung Development Unit, which have provided the technical details about the SMB application and documentation. The team is also very grateful for the support received from Samsung/SIDI team, in particular from Helder Piho, Miguel Lizarraga, Ildeu Fantini and Vera Bier. Professor Fabio Q. B. da Silva holds a research grant from the Brazilian National Research Council (CNPq), process no. 314523/2009-0. CNPq has also provided valuable support to the project through the Brazilian Federal Law no. 8010/90.

6. REFERENCES

[1] Cunningham, W. 1992. The WyCash Portfolio Management System, *Addendum to the proceedings on Object-oriented programming systems, languages, and applications*, pp.29-30, Vancouver, British Columbia, Canada, doi:10.1145/157709.157715

[2] Seaman, C. and Guo, Y. 2011. Measuring and Monitoring Technical Debt, *Advances in Computers*, vol. 82, pp. 25-46, doi: 10.1016/B978-0-12-385512-1.00002-5

[3] Brown, N. *et al*. 2010. Managing Technical Debt in Software-Reliant Systems", *Proceedings of the FSE/SDP workshop on Future of software engineering research*, pp. 47-52, New York, NY, USA, doi:10.1145/1882362.1882373

[4] Buschmann, F. 2011. To Pay or Not to Pay Technical Debt, *IEEE Software*, vol. 28, no. 6, pp. 29-31, doi: 10.1109/MS.2011.150

[5] Klinger, T., Tarr, P., Wagstrom, P. and Williams, C. 2011. An enterprise Perspective on Technical Debt", Proceedings of the 2nd Workshop on Managing Technical Debt, pp. 35-38, New York, NY, USA, 2011, doi: 10.1145/1985362.1985371