Chapter 2
The Technical Debt in Cloud Software Engineering: A Prediction-Based and Quantification Approach

Georgios Skourletopoulos
Scientia Consulting S.A., Greece

Rami Bahsoon
University of Birmingham, UK

Constandinos X. Mavromoustakis
University of Nicosia, Cyprus

George Mastorakis
Technological Educational Institute of Crete, Greece

ABSTRACT
Predicting and quantifying promptly the Technical Debt has turned into an issue of significant importance over recent years. In the cloud marketplace, where cloud services can be leased, the difficulty to identify the Technical Debt effectively can have a significant impact. In this chapter, the probability of introducing the Technical Debt due to budget and cloud service selection decisions is investigated. A cost estimation approach for implementing Software as a Service (SaaS) in the cloud is examined, indicating three scenarios for predicting the incurrence of Technical Debt in the future. The Constructive Cost Model (COCOMO) is used in order to estimate the cost of the implementation and define a range of secureness by adopting a tolerance value for prediction. Furthermore, a Technical Debt quantification approach is researched for leasing a cloud Software as a Service (SaaS) in order to provide insights about the most appropriate cloud service to be selected.

INTRODUCTION
The cloud can be considered as a marketplace (Buyya, Yeo, Venugopal, Broberg, & Brandic, 2009), where the web services of the cloud-based system architectures can be leased (Nallur & Bahsoon, 2012), according to their non-functional requirements (i.e. availability, performance, maintainability, etc.) or their maximum capacity in users. One of the advantages of cloud computing is scalability, where a

DOI: 10.4018/978-1-4666-8225-2.ch002
cloud provider gives the opportunity to easily upscale or downscale the Information Technology (IT) requirements as and when required (Buyya, Ranjan, & Calheiros, 2009; Mousicou, Mavromoustakis, Bourdena, Mastorakis, & Pallis, 2013). Additionally, some of the benefits of using cloud are the reduced cost, rapid development, greater productivity and regular integration (Oza, Münch, Garbajosa, Yague, & Ortega, 2013). In this chapter, the authors motivate the need for predicting and quantifying the Technical Debt on cloud service level in order to gain insights, be more technical debt-aware and make the right decisions within a business context. Seaman et al. (2012) and Snipes et al. (2012) discuss that the Technical Debt data can be used as an effective decision making method, such as for incurring or paying off the Technical Debt instances. Additionally, a cost-benefit analysis is proposed, aiming to elicit and forecast the Technical Debt in order to manage it promptly in the cloud service level and create more precise payback strategies.

In this context, this chapter proposes two novel models for predicting and quantifying the Technical Debt for cloud Software as a Service (SaaS). More specifically, the first model is related to implementing SaaS in the cloud. When implementing SaaS in the cloud, the size of the Technical Debt might be affected due to budget constraints, the levels of experience and productivity of the development team, the introduction of new technologies or any kind of cultural issues. On the contrary, the second model is based on leasing cloud SaaS. In this case, the size of the Technical Debt may be affected due to either the capacity of an offered service or the need to abandon an existing one and/or switch to a more upgraded service. Regarding the latter scenario, a sheer increase in the demand is an inevitable cause of abandoning a service and switching to a more upgraded one, resulting in incurrence of Technical Debt and any positive Technical Debt to be further incurred can be hardly managed.

Following this introductory section, related work approaches are presented in the next section, describing the concept of the Technical Debt from different viewpoints. The subsequent sections elaborate on the need for predicting and quantifying the Technical Debt on cloud service level, including a detailed description of the adopted research approach. Finally, a critical evaluation and conclusion of the chapter is provided, by highlighting directions and fields for future research.

RELATED WORK

The Technical Debt constitutes a metaphor coined by Cunningham (1992), which explains the correlation between the software development and the financial debt. The extra work that is necessary to be held in the future is discussed, aiming to improve some of the non-functional requirements, such as the readability or the complexity (Cunningham, 1992), resulting in additional cost and likely interest payments (Allman, 2012; Curtis, Sappidi, & Szynkarski, 2012; Fowler, 2003; Klinger, Tarr, Wagstrom, & Williams, 2011; Lim, Taksande, & Seaman, 2012). Narrowing in, the term refers to the acceleration of the velocity for releasing software products, which might lead to implications, and forms the tradeoff between short and long-term value (Nord, Ozkaya, Kruchten, & Gonzalez-Rojas, 2012). The rework is the effect when the Technical Debt incurs and can range from a trivial one with few amendments to changes that may affect the whole system. Sterling (2010) mentions that the lifecycle can affect the size of the Technical Debt. For instance, an agile software development process would create less Technical Debt than a waterfall model due to the more flexible structure it has.