Quantifying Reuse in OSS: A Large-Scale Empirical Study

Eleni Constantinou, Aristotle University of Thessaloniki, Thessaloniki, Greece
Apostolos Ampatzoglou, University of Groningen, Groningen, Netherlands
Ioannis Stamelos, Aristotle University of Thessaloniki, Thessaloniki, Greece

ABSTRACT

Reuse is an established software development practice, whose benefits have attracted the attention of researchers and practitioners. In order for software reuse to advance from an opportunistic activity to a well-defined, systematic state of practice, the reuse phenomenon should be empirically studied in a real-world environment. To this end, OSS projects consist a fitting context for this purpose. In this paper, the authors aim at assessing the: (a) strategy and intensity of reuse activities in OSS development, (b) effect of reuse activities on design quality, (c) modification of reuse decisions from a chronological viewpoint and (d) effect of these modifications on software design quality. In order to achieve these goals, the authors performed a large-scale embedded multi-case study on 1,111 Java projects, extracted from Google Code repository. The results of the case study provide a valuable insight on reuse processes in OSS development, that can be exploited by both researchers and practitioners.

Keywords: Empirical Study, Open Source Software, Project Evolution, Reuse Type, Reuse Decisions

INTRODUCTION

Software reuse refers to the incorporation of existing software artifacts to systems, different than the ones that they have been originally developed for (Krueger, 1992). Nowadays, it is generally accepted that reuse practices are beneficial for systems’ development since they increase productivity and potentially improve system product quality (Lim, 1994; Frakes & Kang, 2005; Ajila & Wu, 2007). In this context, the reuse opportunities that open source software repositories provide, through the enormous amount of available software projects, are considered an interesting research domain (Marri, Thummalapenta, & Xie, 2009). However, in order for the aforementioned OSS reuse benefits to be fully exploited, a systematic process is needed. As a first step towards such a systematic approach we consider the understanding of the underlying reuse mechanisms that exist in OSS projects.

Reuse is established as a common phenomenon during software projects’ development (Heinemann, Deissenboeck, Gleirscher, Hummel, & Irlbeck, 2011). In order to understand the phenomenon of OSS reuse, the different types of reuse must be investigated. Software reuse is discriminated in three types: white-box, black-box and glass-box reuse. The first case, i.e.
white-box reuse, refers to source code reuse, where the external source code is incorporated in the project files. In this case, the internal structure of the reused source code is exposed to the developers, and the source code can potentially be modified (Capiluppi, Stol, & Boldyreff, 2011). The second case, i.e. black-box reuse, refers to the reuse of external libraries in binary form, where the source code is not visible and therefore, not modifiable. The third case, i.e. glass-box reuse, refers to the reuse of the source code, but without changes. Since white-box reuse can entail additional effort for modification and maintenance (Sametinger, 1999), it is expected to occur less frequently compared to the other two types of reuse (Schwittek & Eicker, 2013). In cases of white-box reuse, maintenance effort is allocated both to project elements, i.e. elements internally developed by the project members, and reused elements. An indicator of the required maintenance effort is considered to be the design quality of the system (Bauer, Heinemann, & Deissenboeck, 2012). Although the design quality of OSS systems depends on additional factors, e.g. project maturity, we investigate the effect of reuse practices on systems’ design quality. Thus, we measure projects’ structural quality and investigate the effect of reused elements to the overall product quality.

A software reuse process includes the identification of candidate reuse artefacts and, whenever possible, their incorporation in systems under development. However, as projects evolve over time, their requirements and desired functionality change as well (Koch & Schneider, 2002). Therefore, engineers need to revisit their decisions on the employed reused elements, so as to examine if they need to be substituted, maintained, or removed (Bangerth & Heister, 2013). The modification of reused components can occur for several reasons: removal of reused functionality, upgrade of reused library or substitution of reused library (Bangerth & Heister, 2013). In this work, we investigate how the reuse decisions evolve over time, in order to provide insights on how real-world reuse occurs. More specifically, we argue that reuse modification decisions indicate how systematically reuse practices are applied throughout projects’ lifecycle (Yu, 2006). Nonetheless, the reuse modification decisions entail additional effort during development. However, the aforementioned argument is not evaluated from previous studies, because of their research setup. Until now, studies that measure OSS reuse examine reuse practices in several projects, but only for isolated versions. In this study we measure the frequency and magnitude of such modifications. As a first approach to comprehend the rationale behind such modifications, we investigate if they are quality-driven, i.e. if the developers’ focus on the design quality of the newly incorporated reused elements. The implications of modifying the reused elements during projects’ evolution is studied by Dietrich, Jezek, & Brada, 2014 and Raemaekers, van Deursen, & Visser, 2012, although from a different perspective. More specifically, these studies focus on incompatibility issues that arise from library upgrades.

To sum up, this paper elaborates on the existing empirical knowledge on software reuse by providing the following contributions:

**c1:** We extend empirical knowledge by quantifying software reuse in a large number of software projects. Therefore, we gain further insight on reuse practices (intensity, type of reuse, and effect on design quality) for projects of different scale, maturity and history.

**c2:** We compare the design quality of white-box reused elements to the design quality of project elements. Subsequently we investigate how the incorporation of reused elements affects the overall quality of the system, i.e. the projects’ design quality with respect to the magnitude of reuse.

**c3:** We investigate the modification of reuse decisions over systems’ successive versions for each reuse strategy. Thus, we show empirical evidence on developers’ effort allocated on revisiting reuse decisions during projects’ evolution.
Virtualized Open Source Networking Lab
Lee Chao (2013). Open-Source Technologies for Maximizing the Creation, Deployment, and Use of Digital Resources and Information (pp. 85-100).
www.igi-global.com/chapter/virtualized-open-source-networking-lab/70120?camid=4v1a

Open Source Web Portals
www.igi-global.com/article/open-source-web-portals/101215?camid=4v1a