Chapter 6

Tool Assisted Analysis of Open Source Projects:
A Multi-Faceted Challenge

M.M. Mahbubul Syeed
Tampere University of Technology, Finland

Imed Hammouda
Tampere University of Technology, Finland

Timo Aaltonen
Nokia Research Center, Finland

Tarja Systä
Tampere University of Technology, Finland

ABSTRACT

Open Source Software (OSS) is currently a widely adopted approach to developing and distributing software. OSS code adoption requires an understanding of the structure of the code base. For a deeper understanding of the maintenance, bug fixing and development activities, the structure of the developer community also needs to be understood, especially the relations between the code and community structures. This, in turn, is essential for the development and maintenance of software containing OSS code. This paper proposes a method and support tool for exploring the relations of the code base and community structures of OSS projects. The method and proposed tool, Binoculars, rely on generic and reusable query operations, formal definitions of which are given in the paper. The authors demonstrate the applicability of Binoculars with two examples. The authors analyze a well-known and active open source project, FFMpeg, and the open source version of the IaaS cloud computing project Eucalyptus.

INTRODUCTION

Open Source Software (OSS) is currently a widely adopted approach to developing and distributing software. Successful open source projects are typically complex, both from the point of view of the code base and with respect to the developer and user community. Such a project may consist of a wide range of components, coming with a large number of versions reflecting their development and evolution history. While it is a challenge to acquire knowledge from developers and users due to their distributed nature, open source development and user communities often produce a rich software repository as a byproduct. In addition to source code and other software artifacts, there

DOI: 10.4018/978-1-4666-2937-0.ch006

are repositories containing other sources of information such as bug reports, mailing lists, and revision history logs.

A variety of tools and techniques have been proposed to study open source projects. However, most of these techniques suffer from fragmentation and lack of synergies. For instance, many tools apply reverse engineering techniques to study the software side of OSS (Knab, Pinzger, & Bernstein, 2006; Zhou & Davis, 2005). Other works have considered social network analysis techniques to study the social model of OSS (Martinez-Romo, Robles, Ortuo-Perez, & Gonzalez-Barahona, 2008; Kamei, Matsumoto, Maeshima, Onishi, Ohira, & Matsumoto, 2008).

We argue that the separation between the two is artificial. The dimensions are complementary, not discrete categories. They can be used together to provide a lot of useful information. To make a decision about using OSS as a part of a software product to be developed, it is essential to be able to understand the role of the developer community in the development and maintenance of different parts of the OSS code base. This would also help in estimating and planning the future development and maintenance activities of the software product. For instance, understanding the relations of the code and developer community structures helps in understanding where the expertise lay within the developer community, which in turn helps in deciding who to contact concerning issues related to a specific part of the OSS code base.

In this paper we address such challenges as follows: first, a compact and extensible metamodel is proposed which captures both the code base and the community dimension of OSS projects; second, a set of reusable formal descriptions of operations are given, which allows the relationships between the code structure and the community structure to be queried and third, a tool, Binoculars, corresponding to both the metamodel and the defined operations is implemented to study the feasibility of our approach. The tool, Binoculars is able to (a) merge a community view with source code views; (b) provide different perspectives to view the data presented in the form of a graph. This feature can be used, for example, to identify and study the groups of developers working with the same (or related) code fragments, or to study whether communication structures of the developing community conform to the architecture of the software, or trace out the relationship between the developer and the user community in the context of the codebase; (c) render the graph information at different levels of abstraction; (d) provide query support for the graphs.

We also demonstrate the applicability of our approach and the tool with two examples. First, we analyze a well known and active open source project FFMpeg (FFmpeg, 2010) and show how to make queries essential from the point of view of development and maintenance of software relying on FFMpeg code. Then we analyze the open source version of the Eucalyptus project (Eucalyptus, 2011) and its community, due to its emerging impact in the field of cloud computing.

The paper is structured as follows. In Section TOWARDS A GENERIC OSS ANALYSIS TOOL we propose our approach. The tool, Binoculars, is described in Section TOOL SUPPORT. The applicability of Binoculars is discussed in Section CASE STUDY. We review known approaches and techniques for analyzing OSS projects in Section RELATED WORK and distinguish our work from that of existing ones. Finally, some concluding remarks and future development of the work are presented in Section DISCUSSION.

TOWARDS A GENERIC OSS ANALYSIS TOOL

Comprehension of a software project can be addressed at different abstraction levels (Rosso, 2006). At the lowest level, the file system (code base) can be analyzed and refactored by employing reverse engineering techniques to extract code dependency and the architecture of the system. At the next highest level of abstraction, the community structure and people's (developers