Repositories with Public Data about Software Development

Jesus M. Gonzalez-Barahona, Universidad Rey Juan Carlos, Spain
Daniel Izquierdo-Cortazar, Universidad Rey Juan Carlos, Spain
Megan Squire, Elon University, USA

ABSTRACT

Empirical research on software development based on data obtained from project repositories and code forges is increasingly gaining attention in the software engineering research community. The studies in this area typically start by retrieving or monitoring some subset of data found in the repository or forge, and this data is later analyzed to find interesting patterns. However, retrieving information from these locations can be a challenging task. Meta-repositories providing public information about software development are useful tools that can simplify and streamline the research process. Public data repositories that collect and clean the data from other project repositories or code forges can help ensure that research studies are based on good quality data. This paper provides some insight as to how these meta-repositories (sometimes called a “repository of repositories”, RoR) of data about open source projects should be used to help researchers. This paper describes in detail two of the most widely used collections of data about software development: FLOSSmole and FLOSSMetrics.

Keywords: Code Forges, Meta-Repositories, Project Repositories, Repository of Repositories, Software Engineering Research

1. INTRODUCTION

Nowadays, most software development projects use a variety of software tools and online systems for coordinating their work. This is true of both proprietary software projects and free, libre, and open source (FLOSS) projects. These tools include centralized web download areas, discussion forums, IRC channels, email mailing lists, source code control systems, bug trackers, wikis, etc (Fogel, 2005). A project team may choose to manage their own tools, perhaps on their own website, or the team may also choose to use a suite of tools provided by one of many online code forges. (An example of this could be SourceForge\(^1\), Savannah\(^2\) or RubyForge\(^3\)).

Regardless of whether the team chooses to host its own repository or to be part of a code forge, the project tools and systems will leave a detailed trace about the activities and products of the project, which can be analyzed by researchers. Because the end product of a FLOSS project is source code that is freely available, the development for the projects is often done in the open as well. When tools like

DOI: 10.4018/jossp.2010040101
issue trackers or email mailing lists are used for software development on a FLOSS project, the data they produce is very often left open to the public. The result is a very large amount of publicly-available data about software development. For example, the archives of an email mailing list will serve as a record of the discussions between developers on a project. This will be searchable by anyone seeking a history of how decisions were made on a project. An online issue tracking system can show the list of features requested or bugs reported in a project, as well as the record of how the feature or bug was handled by the development team. The issue tracking software produces logs that are searchable by anyone wanting to know more about this process.

This public record of activity should make it possible for researchers to develop empirical studies based on data retrieved from these systems or, to be more precise, from the code forges and project repositories that support those systems (Dyba et al., 2005; Kitchenham et al., 2002). However, researchers often discover that even when the data is publicly available (such as it is with most FLOSS projects), the task is made more complicated by the large size and scope of the project repositories or code forges, and the heterogeneity of the projects being studied (Howison & Crowston, 2004; Robles et al., 2008). Making sense of all this data for a research study can be a large challenge, especially for large collections of projects, or for small numbers of projects that have been in existence for a long time.

In this paper, we will outline what kind of data is available from the project repositories or code forges, and what each “repository of repositories” (or RoR) is doing to collect, aggregate, and clean the targeted repository or forge data. We will provide examples from two RoRs, FLOSSMetrics and FLOSSmole, showing what data is available on the original repository or forge, and how that RoR makes this data available to researchers. We also discuss the advantage presented to the researcher for using these RoRs: the researcher does not need to collect data independently. This is an important advantage as it can help the entire research community leverage the work done by others, thus freeing up time and effort to analyze the data rather than to collect it. Finally, we show some example research questions that can be answered using FLOSSMetrics and FLOSSmole data. We conclude our paper with some general observations about the future direction of the RoRs, given the needs of the research community.

2. DATA RETRIEVAL: THE FIRST STEP

In this section we discuss what data is available and how this data can be organized for ease-of-use by the researcher. The large volume of data in public software repositories is usually comprised of:

- Metadata about software projects. Projects, particularly when they are hosted in development forges, are usually annotated with some metadata which is potentially interesting for researchers. Examples of that metadata are: descriptions of the project, list of developers, rankings (according to some metrics maintained by the forge), licensing schemas, programming language(s) used, language(s) into which the software has been translated, etc.
- Source code, including metadata about how it changed. Nowadays software projects commonly use a source code management system such as Subversion, git, or CVS. When this source code management system is public, as it is with FLOSS projects, not only all the versions of all the source code files can be obtained, but also metadata about who performed all the changes to that code, and when they did it (sometimes ‘why’ can be ascertained as well). Metrics can be obtained about all releases of files, and about snapshots of the source code tree. Source code metadata can be used to better understand the actors and artifacts
A New Approach to Knowledge Sharing: The Multifactory Model
Giulio Focardi and Lorenza Salati (2015). *Societal Benefits of Freely Accessible Technologies and Knowledge Resources (pp. 211-236).*

[www.igi-global.com/chapter/a-new-approach-to-knowledge-sharing/130789?camid=4v1a](http://www.igi-global.com/chapter/a-new-approach-to-knowledge-sharing/130789?camid=4v1a)