Science of Computer Programming 75 (2010) 264–275

Contents lists available at ScienceDirect

Science of Computer Programming

journal homepage: www.elsevier.com/locate/scico

The Small Project Observatory: Visualizing software ecosystems

Mircea Lungu [a,*], Michele Lanza [a], Tudor Gîrba [b], Romain Robbes [a]

[a] REVEAL @ Faculty of Informatics, University of Lugano, Switzerland

[b] Software Composition Group, University of Bern, Switzerland

ARTICLE INFO

Article history:
Received 1 December 2008
Received in revised form 17 August 2009
Accepted 3 September 2009
Available online 11 September 2009

Keywords:
Software evolution
Software visualization
Software ecosystems
Reverse engineering
Maintenance

ABSTRACT

Software evolution research has focused mostly on analyzing the evolution of single software systems. However, it is rarely the case that a project exists as standalone, independent of others. Rather, projects exist in parallel within larger contexts in companies, research groups or even the open-source communities. We call these contexts software ecosystems. In this paper, we present the Small Project Observatory, a prototype tool which aims to support the analysis of software ecosystems through interactive visualization and exploration. We present a case study of exploring an ecosystem using our tool, we describe the architecture of the tool, and we distill lessons learned during the tool-building experience.

1. Introduction

Software visualization tools span a wide range of abstraction levels. One of the first visualization tools was Eick's SeeSoft [1] which summarizes changes in the software at the level of lines of code. Increasing the level of abstraction, other tools present UML diagrams or other representations that work at the class level [2,3]. Others focus on visualizing higher-level abstractions such as modules and the interdependencies between them [4,5]. At the next higher abstraction level, Holt visualized architectural differences between two versions of a system [6].

An underlying assumption of these approaches is that a project represents one single, complex entity that can be analyzed in isolation. However, we argue that this is often not the case. Instead, projects exist in larger contexts such as companies, research groups or open-source communities and corresponding versioning repositories exist in parallel. We call these context ecosystems.

In this article we present the Small Project Observatory (SPO), a tool that embodies our approach to analyzing software ecosystems. Our first goal when building SPO was to provide an interactive interface for the visualization and exploration of software ecosystems. A few other research projects analyze such groups of projects and treat them as whole, but none of them provides an interactive tool to support the analysis. One such project is FlossMole which provides a database compilation of open-source projects from Source-Forge and several other repositories [7]. Weiss analyzed Source-Forge from a statistical point of view [8].

Groups of software projects have been also analyzed in various instances by the members of the Libresoft research group from Spain whose main interest is analyzing open-source software. In this context they looked at various collections of systems to study their properties and their evolution. In one instance, they analyzed the Debian Linux distribution and estimated the cost of implementing it from scratch [9]. The analysis of the Debian distribution was not focused on the source code but remained at a higher level. In another article, they proposed a quantitative methodology for analyzing how

* Corresponding address: REVEAL @ Faculty of Informatics, University of Lugano, Via Giuseppe Buffi 13, 6900 Lugano, Switzerland.
E-mail addresses: mircea.lungu@usi.ch (M. Lungu), michele.lanza@usi.ch (M. Lanza), girba@iam.unibe.ch (T. Gîrba), romain.robbes@usi.ch (R. Robbes).

0167-6423/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.scico.2009.09.004

the developer turnover affects open-source software projects [10]. In this work they took a few representative open-source projects and studied the information in the versioning system repositories without looking at entire groups of projects that belong together. A case where they indeed took entire collections of projects was when they studied the behavior of the developers from a social networking analysis point of view [11]. In this work they looked at the social networks that are built around the Gnome and Apache projects. All of the previous work does however consider group of projects as simple collections of projects, not ecosystems.

The Small Project Observatory is implemented as an online application. There are several other examples of using the Web for software engineering. Holt et al. have developed the Software Bookshelf, which is a web-based paradigm for the presentation and navigation of information representing large software systems [12]. Mancoridis et al. presented REportal which was aimed to be an online reverse engineering portal [13]. Recently D'Ambros et al. proposed an online framework for analysis and visualization in a software maintenance context. Their tool, Churrasco emphasizes flexibility by allowing easy model extension as well as collaboration [14]. Although they support loading multiple models at the same time in Churrasco, they do not consider an ecosystem as a first-class entity in their analysis.

The remainder of this article is organized as follows. In Section 2 we introduce the concept of a software ecosystem. In Section 3 we present the Small Project Observatory, its user interface, and an example of analysis performed using it. In Section 4 we talk about the architecture of the tool. In Section 5 we discuss about validation, interaction, developing for the web and other lessons learned during our tool-building experience. We conclude in Section 6 with a discussion and we outline the directions of future work.

2. No project is an Island

Software systems are seldom developed in isolation [15,16]. On the contrary, many companies, research institutions and open-source communities deal with software projects developed in parallel and depending on one another. Such collections of projects represent assets and analyzing them as a whole can provide useful insights into the structure of the organization and its projects.

In this context we define a software ecosystem as follows:

A software ecosystem is a collection of software projects which are developed and evolve together in the same environment.

The environment is usually a large company, an open-source community, or a research group. It is even possible for multiple organizations to collaborate and develop software in a common ecosystem.[1]

One particular type of organization which owns ecosystems are telecommunications companies. They are supported by highly integrated IT systems developed decades ago and extended over time. Telecommunications companies usually develop software products for variations of the phone hardware and in these cases they usually have to manage the various versions which are slightly different by introducing product line families. A product family [17,18] is a set of software intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way. A product family is indeed a special type of ecosystem, but the ecosystem concept is much broader.

The large amounts of code that is developed in an ecosystem guarantees that it is very hard, if not impossible for a single person to keep track of the complete picture. Many times, even if there exists documentation to describe the interdependencies between the projects and the way the developers and teams are supposed to collaborate, it is out of date or inaccurate. Thus, the only reliable source of information about the ecosystem is the data present in the versioning repositories of the projects. In this context we define a super-repository as follows:

A super-repository represents a collection of version control repositories of the projects of an ecosystem.

Beside the information about the evolution of the projects in an ecosystem, super-repositories also contain information about developers. Depending on the organization, developers might work on multiple projects simultaneously, move from one project to another and collaborate on multiple projects at the same time. All this information can be recovered from the information in the super-repository.

Some repositories are dedicated to a particular language such as RubyForge[2] and SqueakSource,[3] while others are language agnostic such as SourceForge[4] and GoogleCode.[5] Although most of the discourse can be generalized to any of these repository types, this article is derived from our experience of analyzing three open-source and one industrial super-repositories, each containing the history of several dozens to hundreds of applications versioned in Store (Smalltalk Open Repository Environment).

1 For example, the Gnome family of systems is developed by an open-source community to which various companies contribute, too.
2 http://rubyforge.org.
3 http://squeaksource.org.
4 http://sourceforge.org.
5 http://code.google.com.