SECONDA: Software Ecosystem Analysis Dashboard

Javier Pérez, Romuald Deshayes, Mathieu Goeminne, Tom Mens
Software Engineering Lab, University of Mons, Belgium
{javier.perez, romuald.deshayes, mathieu.goeminne, tom.mens}@umons.ac.be

Abstract—Software ecosystems are coherent collections of software projects that evolve together and are maintained by the same developer community. They exhibit some particular evolution features because of the dependencies between the software projects and the interactions between the community members. Tools for analysing and visualising the evolution of software ecosystems must take these aspects into account. SECONDA is a software ecosystem visualization and analysis dashboard that offers both individual and grouped analysis of the evolution of projects and developers belonging to the software ecosystem, at coarse-grained and fine-grained level. Using GNOME as a case study, we use SECONDA to study these ecosystem and community aspects.

Index Terms—open source, software ecosystem, developer community, software evolution, visualisation, empirical analysis

I. INTRODUCTION

In this article we present SECONDA, a tool under active development at our research lab. It provides a visual dashboard for analysing and understanding the evolution of software ecosystems, that can be seen as “a collection of software projects which are developed and evolve together in the same environment” [3]. Related approaches [4] demonstrate the need for analysing the evolution of such software ecosystems.

The quality and health of a software project is not only related to its product properties, but also to the characteristics of the software ecosystem and developer community that surrounds it. In addition to the fine-grained analysis that other software evolution analysis tools support, the study of the evolution of a single project or product, SECONDA offers a coarse-grained analysis that takes into account the role of a software project within the evolution of a larger software ecosystem. It also takes into account the “people” factor, by analysing the evolution of characteristics of persons belonging to the developer community. This will enable researchers to study how project evolution is influenced by other projects and by human factors, and how the quality of a collection of projects relates to the quality of each individual project.

Other software evolution studies, such as [1], [5], started analysing software ecosystems as a whole, but lack visualisation tools to present the results of the analysis in a convenient way. More recent projects, such as the open source software directory Ohloh [2], offer insightful details about the evolution of several projects in terms of size and number of authors/committers. It also displays developer profiles for tracking the activity of each community member. Nevertheless, the service lacks support for software ecosystem analysis and offers a limited comparison functionality that allows to compare the evolution of up to three software projects.

II. ARCHITECTURE

The SECONDA visualisation dashboard integrates a set of third-party and custom-built components. It is comprised of five modules: data extraction, metrics computation, visualisation, statistical analysis and reporting. An overview of the these modules is given in Figure 1. A screencast demonstrating the usage and functionalities of SECONDA is available at the following address: http://www.youtube.com/watch?v=7p9BfJ4HmDA

The data extraction module relies on Git repositories that are replicated in a local cached copy to be used by SECONDA during software ecosystem analysis. The extraction module also obtains the commit log history of each project belonging to the ecosystem. A post-processing phase, takes care of identity matching, to identify and match different names (usernames, mail addresses, logins, etc.) used by the same developer.

The metrics computation module uses shell tools (sed, awk, etc.) on the commit logs and source code from the locally stored Git clone, and metrics tools (CMETRICS and SLOCCOUNT) for computing size (lines of code per programming language used) and complexity metrics (e.g. McCabe and Halstead) for C code\(^1\). The results are stored in a MySQL database and in CSV files.

The visualisation module, implemented in JAVA, is based on JFREECHART, a library for producing a variety

\(^1\)Metrics tools for other languages can be integrated easily.
of charts. It is used to visualise the evolution of the software ecosystem, project, community and developer information at a coarse-grained and fine-grained level. The statistical analysis module, implemented in R, provides a portfolio of statistical techniques for hypothesis testing, regression, correlation, distribution fitting and so on. Finally, reporting modules can be added to create document files that report on the analysis carried out.

At the time of writing, the identity matching and statistical analysis are not fully integrated yet, and the reporting modules still need to be developed. They are nevertheless described here in order to provide a complete overview of the SECONDA tool.

III. CASE STUDY: GNOME

We have studied the GNOME ecosystem for testing our approach. We have selected GNOME because it has the following requirements: it has a long development history (at least several years) in order to obtain meaningful results about its evolutionary history; it possesses a large developer community, i.e., in which many different developers are involved; it contains a large and active software ecosystem, i.e., a large number of projects, many of which still being actively maintained today; it is open source, and therefore the code is available for downloading and experiment with it; GNOME is well-known to researchers and developers alike.

We refer to the GNOME ecosystem as the set of projects that evolve within the GNOME environment and more precisely, the projects that are stored at the GNOME Git source code repository. Currently, the repository stores over 1330 projects (September 2011), whose life time spans from a couple of months –e.g., gnome-contacts– to 14 years –e.g., gnome-disk-utility–. In addition, the majority of GNOME projects (over 900 of them) are no longer being actively maintained today.

Using the Git terminology, project developers are explicitly distinguished between committers and authors. The committer is the person that has the right to commit files to the repository. The author is the person that actually made the changes to the committed files. Table I illustrates how some project characteristics vary across GNOME projects.

<table>
<thead>
<tr>
<th></th>
<th>authors</th>
<th>committers</th>
<th>commits</th>
<th>files</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimum</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>25</td>
</tr>
<tr>
<td>Q1</td>
<td>3</td>
<td>2</td>
<td>23</td>
<td>61</td>
</tr>
<tr>
<td>median</td>
<td>12</td>
<td>9</td>
<td>131</td>
<td>112</td>
</tr>
<tr>
<td>Q3</td>
<td>59</td>
<td>46</td>
<td>517</td>
<td>237</td>
</tr>
<tr>
<td>maximum</td>
<td>1142</td>
<td>692</td>
<td>35191</td>
<td>7097</td>
</tr>
<tr>
<td>mean</td>
<td>62.07</td>
<td>45.78</td>
<td>760.2</td>
<td>252.3</td>
</tr>
</tbody>
</table>

TABLE I

VARIATION ACROSS 1325 GNOME PROJECTS OF NUMBER OF COMMITS, COMMITTERS, AUTHORS AND FILES. THE FIRST THREE VALUES ARE COMPUTED FOR THE FULL PROJECT HISTORY, THE NUMBER OF FILES IS SHOWN HERE FOR ONLY THE LAST CONSIDERED COMMIT OF EACH PROJECT.

http://git.gnome.org
IV. DATA EXTRACTION AND METRICS COMPUTATION

SECONDA provides two types of manipulation of the data extracted from Git repositories: global analysis, which clones the Git repository and performs a global coarse-grained analysis; and local analysis, which analyses the software projects in the local repository clone at a fine-grained level. Although both analyses can be used independently, it is advisable to first carry out the global analysis, and then request a local analysis for those projects that deserve more attention.

A. Global analysis

The data extraction module downloads and maintains a local cached copy of the GNOME repository on which the metrics module runs a coarse-grained analysis, using SLOCCOUNT for obtaining the projects’ size metrics, for the latest revision of each project, and with Git for obtaining the commit history. For each project we extract and store the list of authors and committers, the Git commit log and the results of running SLOCCOUNT for the whole project. The latter counts the lines of code for a variety of programming languages (including Ada, C, C++, Cobol, Fortran, Haskell, Java, Pascal, LISP, XML, Perl, PHP and many more). The raw repository data is also summarized into a CSV data file.

The analysis is run over the local repository cache, unless a project hasn’t been downloaded yet or an update of the local copy is needed. In such cases, the latest revision of the project is pulled and stored in the local repository. The first time the cache is created the extraction process can take several hours. This is reduced to minutes for the successive executions of the tool.

B. Local analysis

Given that the large majority of code files for GNOME are written in C, local analysis relies on CMETRICS, an open source metrics tool for C code to compute, among others, size metrics (SLOC), Halstead metrics (HLEN, HVOL, HLEVEL, etc.) and McCabe’s cyclomatic complexity (CYCLO).

The data extraction module creates a MySQL database for each project. This database is filled by the metrics module with the information computed by CMETRICS for each revision of the project stored in the local repository. CMETRICS collects data related to C files and to the functions contained in them as well. All this information, together with the links between revisions, files and functions is stored in each database. This makes it possible, for example, to know in which file is contained a certain function and by extension, what revision this file came from.

Once all the data of all the projects’ commits is extracted and stored, the database can be queried to perform detailed analyses. It is also possible to compare data across projects by performing searches over the databases of each project.

V. VISUALISATION

The visualisation module allows to display global and local analyses and therefore, to gain understanding of individual projects or to compare metrics across different projects and developers.

A. Ecosystem and project analysis

GNOME projects can be jointly analysed by combining their separate metrics. The visualisation module uses the main metrics previously computed and displays them using four different types of charts: scatter plots that allow to confront two metrics in order to visualise and find out their possible correlation (see Figure 2); programming language boxplots that display the usage distribution of different programming languages, including main descriptive statistics such as mean, median, and quartiles; ecosystem boxplots that display the distribution of number of commits, committers, authors, and files over all projects. spider web metrics that display and compare a set of metrics for a set of different projects selected by the user (see Figure 3).

The fine-grained analyses for single projects allow to visualise and understand the evolution of each project over time. It comprises two different types of charts:
Fig. 3. Radar chart visualising the comparison of 5 coarse-grained metrics for a user-defined selection of 3 projects.

**B. Community and developer analysis**

From the point of view of SECONDA, there is a duality between projects and developers on the one hand, and between the developer community and the software ecosystem on the other hand.

We already explained before that SECONDA can visualise and analyse data at the project level, by considering all commit activities carried out by all developers for this particular project. The dual of this is the visualisation of data at developer level: SECONDA can visualise all activities carried out by a given developer for all projects this developer has been involved in.

Similarly, SECONDA can either be used to compare a selection of projects against one another (see, e.g., Figure 3) or, alternatively, compare the work carried out by a selection of developers against one another. Finally, SECONDA can perform global analyses of the entire ecosystem (see, e.g., Figure 2 where each point represents a project) or, alternatively, perform a global analysis of the entire developer community (each point in the visualisation represents an individual developer).

**VI. Work in progress**

The current version of the tool is under active development and changing every week. Currently, we are working on the following issues that will be integrated in future releases of SECONDA: integrate the identity matching algorithm (already implemented) as a post-processing phase of the data extraction module; implement an incremental version of the data extraction and metrics computation to accommodate new projects’ revisions without needing to recompute everything everytime, thereby saving bandwidth, time and memory; integrate the community and developer analysis; integrating the statistical analysis module; implement a reporting module; analyse other ecosystems than GNOME; support other types of version repositories than GNOME; support other ecosystems than GNOME; support other types of version repositories than GNOME; support other types of version repositories than GNOME; support other types of version repositories than GNOME; support other types of version repositories than GNOME; support other types of version repositories than GNOME; support other types of version repositories than GNOME; support other types of version repositories than GNOME.

**VII. Conclusion**

In this article we presented the essential characteristics of SECONDA, an extensible modular framework for the analysis and visualisation of open source software ecosystems. The tool is under active development at our lab, and is useful for researchers and practitioners that wish to study how the evolution of open source projects is influenced by their surrounding ecosystem and developer community. It offers a dashboard for rapid visualisation of global (ecosystem-level) and local (project-level) metrics that can be extracted from information stored in the version repositories. Currently, the tool is used for analysing the GNOME ecosystem, but other ecosystems will be analysed in future releases. We will also continue to extend the dashboard with new functionalities, such as person and community visualisation, statistical analysis and reporting.

**Acknowledgments**

This research has been co-funded by the European Regional Development Fund (ERDF) and Wallonia, Belgium. The research is also partially supported by the F.R.S.-FNRS FRFC project 2.4515.09.

**References**


