GlueTheos: Automating the Retrieval and Analysis of Data from Publicly Available Software Repositories

Gregorio Robles
Universidad Rey Juan Carlos
grex@gsyc.escet.urjc.es

Jesus M. González-Barahona
Universidad Rey Juan Carlos
jgb@gsyc.escet.urjc.es

Rishab A. Ghosh
MERIT - Univ. Maastricht
rishab@merit.unimaas.nl

Abstract

For efficient, large scale data mining of publicly available information about libre (free, open source) software projects, automating the retrieval and analysis processes is a must. A system implementing such automation must have into account the many kinds of repositories with interesting information (each with its own structure and access methods), and the many kinds of analysis which can be applied to the retrieved data. In addition, such a system should be capable of interfacing and reusing as much existing software for both retrieving and analyzing data as possible.

As a proof of concept of how that system could be, we started sometime ago to implement the GlueTheos system, featuring a modular,flexible architecture which has been already used in several of our studies of libre software projects. In this paper we show its structure, how it can be used, and how it can be extended.

Keywords: *Mining source code repositories, proposals for exchange formats, meta-models, and infrastructure tools, integration of mined data with other project data*

1 Introduction

Libre software projects[1] range from very small ones (with just one developer commited to his own toy) to large-scale global projects whith thousands of collaborating developers [9]. Specially, most of the larger projects follow a way of organization that has been called the 'bazaar'-style development [14], open to everybody willing to participate. Thus, all elements taking part in the software development process are as much open as possible, in the sense that the generated information is publicly available so that it is easier for 'newcomers' to become integrated in the project. Fortunately, this strategy offers to researchers the chance to access large amounts of data about the development process, the participants and, of course, the output product: the software.

[1]In an attempt to avoid any confusion regarding the meaning of free in free software, throughout this article, the term libre software is used instead. It was chosen because of its meaning pointing towards liberation, and not of merely being costless. The term Open Source is refused for its ignorance about the philosophical foundations of what free software meant in the first place. "Libre software" is a term which is more and more usual in some communities, among them many European and Latin American countries.

Previous studies have taken advantage of this situation, and several research groups have focused their attention on the libre software phenomenon in the last years. For instance, [6] offers a software evolution analysis of the Linux kernel versions -without doubt the most known libre software project- following the classical software evolution point of view [11]. Others have paid attention to economic parameters [10] and have investigated how well classical software cost prediction models (as among others COCOMO [1]) can be applied. In [13] it is shown how libre software projects are composed usually of 10 to 15 core developers who lead the software process, a group of around one order of magnitude larger that participate in minor development tasks (bug fixes, etc.) and a final group around another order of magnitude that helps by other means (bug reports, etc.). In any case, the availability of data has proven to be very positive for research in the libre software environment.

But the amount of data and information available for inspection is that big that these analysis are often regarded as being too superficial. An example where this is common case are source code repositories. In such systems, not only the last state of the code is available for download but also all previous states. The amount of information that is ready for being extracted and analyzed is enormous and two factors become key points: automation and data mining.

When analyzing the data available in publicly accessible repositories, the automation of the data retrieval and the quantitative analysis is of great importance[15][3]. In the case of libre (free, open source) software projects, repositories are managed with very similar software (if not the same)[2], and similar access protocols, so automation allows for the access to most of the available projects. Some methodologies have already being described which make strong use of some kind of automated tools to perform this tasks [3, 4, 7, 17, 10, 15] but they usually make use of ad-hoc tools, without proposing a general architecture flexible enough to make several kinds of distinct analysis on different kinds of software repositories.

That is precisely what we have addressed with GlueTheos: to design a system with an architecture which allows in a way as general and flexible as possible the data retrieval and analysis of public software development data repositories. Currently it can access CVS repositories and archives of source packages (both in deb and rpm formats), but others (such as bug tracking systems and mailing lists) are being included soon in the system. It is designed in a highly modularized way, so that adding new retrieval methods (from CVS or other data repositories) and analysis procedures is simple.

2 The GlueTheos system

The structure of the GlueTheos system is simple. Around a core of coordination scripts, there are input modules which download raw data (currently source code) from the repositories where it resides, modules which analyze such code from several points of view (counting lines of code or identifying authorship information), modules which store the information obtained in the previous phase (as a set of XML files or in an SQL database, for instance) and modules which produce the final reports (tables with data, graphs, etc.)

In the rest of this section, all those modules will be discussed in more detail:

- Core scripts. These scripts are the usual interface for users. With the help of a configuration file, they decide which repository is to be used, and which downloading, analyzing, storage and reporting modules will be run for it, according to the characteristics of the repository, the kind of intermediate storage desired and the final reports wanted. The configuration file can also be used to determine, for instance, that periodic snapshots from a CVS are to be retrieved, to study the evolution of a project. Or to perform only some stages of the whole process (like, for instance, downloading, analyzing and storing results, skipping the reporting phase which could be done later).

- Downloading modules. For each kind of repository, a downloading module is available. Currently, there are three: one for accessing CVS repositories, another for storages of RPM source packages (and in particular, those found in Red Hat Linux distributions), and yet another for Debian repositories (with source packages in the deb format). Those modules are capable of downloading the source code, unpackaging it if necessary (for instance, in the case of source packages), and having it ready for the next stage (which usually is the analysis of the code).

[2] The 12 biggest projects in size in Debian 3.0 use a software repository, all of them CVS besides Linux which uses BitKeeper, a proprietary solution

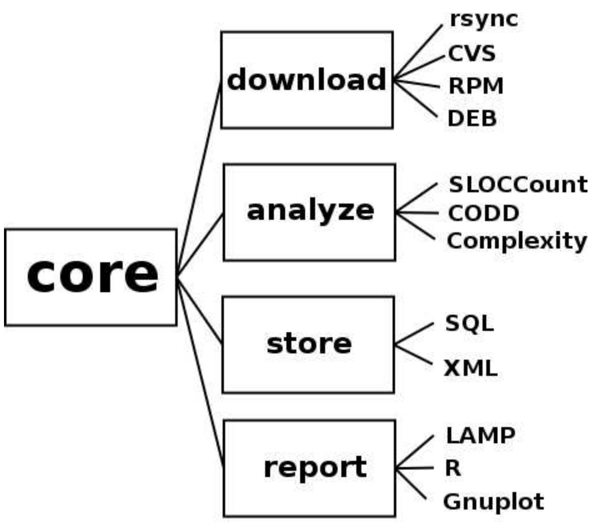

Figure 1. The Gluetheos modules

- Analyzing modules. For this stage, mostly external tools are used, like SLOCCount [17, 16], CODD [2], tools for metrics estimation (which include algorithms to calculate Halstead's[8] and McCabe's[12] complexity measures), raw count of file sizes (using for instance the wc utility), and others. Therefore, these modules are mainly drivers for those tools. Usually, they run the specific external program they drive, and produce results in a given data directory, in the output format used by that program.

- Storage modules. For making it simple the generation of reports, the information has to be in an easy to query storage. In addition, exchange formats have to be defined when information is to be moved or disseminated for study by other groups or at other locations. Currently, for most analyzing modules we have two storage modules, one generating XML files and other using SQL commands to feed a database. The first one is mainly intended for data interchange, while the other is better used for querying in the final stage. Now, we are moving to an architecture where there are only SQL modules for each analyzing module, and an

SQL to XML translator (also dependent on the analyzing module used, but much more simple).

- Reporting modules. These modules are the producers of the outputs of the system. They usually query the database, and massage the obtained information to produce tables, statistical analysis or graphs. For doing their work, in many cases those modules use also external tools, such as Ploticus or Gnuplot for generating graphs, or R for statistical analysis. There are also reporting modules which generate information in a format suitable for being browsed via web, with the help of some PHP code (LAMP = Linux + Apache + MySQL + PHP).

Currently, GlueTheos is written in Python, using Python standard libraries to access SQL databases, to generate XML files or to interface with other tools.

3 Some examples of use

The GlueTheos system has been used in the two following cases (which may serve as examples illustrating its capabilities):

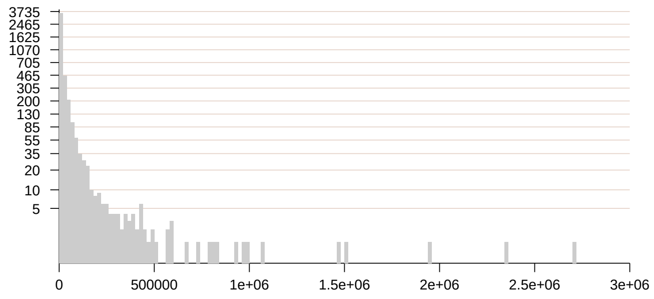

Figure 2. Histogram with the SLOC distribution for packages in Debian 2.0

- The study of the packages in the Debian GNU/Linux[7] and Red Hat Linux[17], distributions (Figure 1 and Figure 2). For this study, GlueTheos downloaded several Debian and Red Hat distributions, and analyzed the resulting source code by using SLOCCount to count its lines in several ways, like SLOC (source lines of code) per package or SLOC per programming language.

- The construction of the website `http://libresoft.dat.escet.urjc.es`. This site includes information and results about several libre software projects, from several points of view gained with the aforementioned analysis and measurement tools GlueTheos makes use of. Most of the information available publicly there has been built with the help of GlueTheos. One of the goals of this website is to offer the libre (free, open source) community the possibility to obtain feedback from our research.

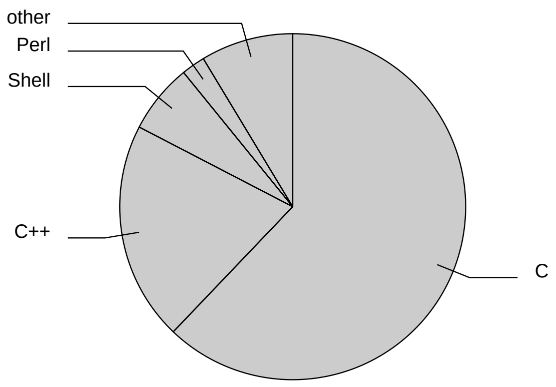

Figure 3. . Pie graph with the SLOC count for main languages in Red Hat 8.0 distribution

GlueTheos also provides an excellent opportunity for economists to measure the demonetized, previously invisible productivity of open source software projects, and also to analyze the organization and production methods of software at a level of detail probably unmatched by any other field of economic activity. This is because almost every single act of production, direct or indirect, is documented and recorded somewhere in the open source development process, much of which is captured and quantified by GlueTheos, which already pays attention to economic measures (such as the Gini[5] coefficient of concentration).

Although this sort of measurement may not, initially, be in monetary terms, it does represent human time and effort spent on productive activity, and can be "remonetized" at least for the purposes of measurement. One use for this may be to improve models for cost estimation of software development, by correlating time spent as reported by individual developers in surveys, with their productivity as determined through the examination of source code and related metadata (such as CVS).

4 Conclusions and further work

The GlueTheos system is an attempt to build a set of tools capable of automating most of the tasks related to the analysis of publicly available information about libre software projects. Currently, it can access CVS repositories and archives of some GNU/Linux distributions. By using external tools it can make several different analysis on the fetched data, and produce several kinds of reports (from tables with organized data to graphs or information suitable for being offered in a website. GlueTheos pretends to fill the gap that exists for in-depth, fully-automated analysis.

Our group is working currently in stabilizing the system, making it more versatile (including more downloading, analyzing and reporting modules), and exploring data formats for the exchange of information about libre software projects. We are planning also to put a big effort in the reporting modules, so that information from different sources can be integrated and correlated giving a wider picture than the one that a unique tool may offer. Special attention is being given in showing the huge amount of data in a way that it is comprehensible avoiding the problem of information overload that is common in these scenarios.

Future plans also include to set up an interactive website where libre software developers can request their projects to be analyzed. Developers would have only to fill out a form where the location of the publicly available data sources should be specified and the system will automatically retrieve and analyze them, putting up a web-sites with the results and finally notifying the developers that they can see results there.

All the GlueTheos system, and the external tools it uses, are libre (free, open source) software.

References

[1] B. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.

[2] Codd website. `http://codd.berlios.de/`.

[3] D. Germán and A. Mockus. Automating the measurement of open source projects. In *Proceedings of the 3rd Workshop on Open Source Software Engineering, 25th International Conference on Software Engineering*, Portland, Oregon, 2003.

[4] R. A. Ghosh. Clustering and dependencies in free/open source software development: Methodology and preliminary analysis. In *Open Source Workshop*, Toulouse, France, June 2002.

[5] C. Gini. *On the Measure of Concentration with Espacial Reference to Income and Wealth*. Cowles Commission, 1936.

[6] M. W. Godfrey and Q. Tu. Evolution in open source software: A case study. Oct. 2000.

[7] J. M. González-Barahona, M. A. Ortuño Pérez, P. de las Heras Quirós, J. Centeno González, and V. Matellán Olivera. Counting potatoes: The size of Debian 2.2. *Upgrade Magazine*, II(6):60–66, Dec. 2001. `http://people.debian.org/~jgb/debian-counting/counting-potatoes/`.

[8] M. H. Halstead. *Elements of Software Science*. Elsevier, New York, USA, 1977.

[9] K. Healy and A. Schussman. The ecology of open-source software development. Technical report, University of Arizona, USA, Jan. 2003. `http://opensource.mit.edu/papers/healyschussman.pdf`.

[10] S. Koch and G. Schneider. Results from software engineering research into open source development projects using public data. *Diskussionspapiere zum Tätigkeitsfeld Informationsverarbeitung und Informationswirtschaft*, (22), 2000. `http://wwwai.wu-wien.ac.at/~koch/forschung/sw-eng/wp22.pdf`.

[11] M. Lehman, J. Ramil, P. Wernick, and D. Perry. Metrics and laws of software evolution - the nineties view. 1997.

[12] T. McCabe. A complexity measure. *IEEE Transactions on Software Engineering*, 1976.

[13] A. Mockus, R. Fielding, and J. Herbsleb. A case study of open source software development: The Apache server. In *Proceedings of the 22nd International Conference on Software Engineering (ICSE 2000)*, pages 263–272, Limerick, Ireland, 2000.

[14] E. S. Raymond. The cathedral and the bazar. *First Monday*, 1997. `http://www.firstmonday.dk/issues/issue3\_3/raymond/`.

[15] G. Robles-Martinez, J. M. Gonzalez-Barahona, J. Centeno-Gonzalez, V. Matellan-Olivera, and L. Rodero-Merino. Studying the evolution of libre software projects using publicly available data. In *Proceedings of the 3rd Workshop on Open Source Software Engineering, 25th International Conference on Software Engineering*, pages 111–115, Portland, Oregon, 2003.

[16] Sloccount. `http://www.dwheeler.com/sloccount/`.

[17] D. A. Wheeler. More than a gigabuck: Estimating gnu/linux's size, June 2001. `http://www.dwheeler.com/sloc/redhat71-v1/redhat71sloc.html`.