Mining Challenge 2010: FreeBSD, GNOME Desktop and Debian/Ubuntu

Abram Hindle
David Cheriton School of Computer Science
University of Waterloo
Waterloo, Canada
ahindle@swag.uwaterloo.ca

Israel Herraiz
School of Engineering
University Alfonso X el Sabio
Madrid, Spain
herraiz@uax.es

Emad Shihab and Zhen Ming Jiang
Software Analysis and Intelligence Lab (SAIL)
Queen's University
Kingston, Canada
{*emads, zmjiang*}*@cs.queensu.ca*

***Abstract*—In a young field, such as Mining Software Repositories (MSR), there is always a call for benchmarks so that researchers can compare their results against others. Thus in order to explore and discover the breadth of MSR research, the MSR community has banded together behind the MSR Mining Challenge. The mining challenge allows researchers to demonstrate current working techniques against a common set of repositories or datasets with the express purpose of mining interesting facts from these datasets and then comparing these results against the results from other researchers. This year, 2010, the MSR Mining Challenge has expanded the size of its underlying dataset to include the version control, bug tracker, and mailinglists of the following software distributions and projects: FreeBSD, GNOME Desktop and Debian/Ubuntu. Researchers are asked to look beyond the boundaries of a project and investigate the relationship between the evolution of various programs contained within these software ecosystems.** 9 **general challenge submissions were submitted,** 6 **were accepted with a** 66% **acceptance rate.**

I. Introduction

The MSR Mining Challenge is a track of the Working Conference on Mining Software Repositories, that brings together researchers and practitioners who are interested in applying, comparing, and challenging their mining tools and approaches on the software repositories of open source projects.

This year, the MSR Mining Challenge focused on evolving software ecosystems and the repositories associated with those systems [1]. We introduced a new kind of repository to the challenge: module dependencies from the Ultimate Debian Database. Other repositories were extracted from FreeBSD and the GNOME Desktop. The focus for this year's challenge was on relationships between packages described by these various data-sources.

As with previous years we also have a prediction track related to some of our general track challenge data. Researchers were asked to estimate the bug report growth of Debian by April 30th, 2010, based on data available from February 2010 or earlier.

Thus this year's challenge provided a wide range of data which allows many different kinds of tools to be used, such as mailing list analyzers, bug tracking system analyzers, and source code analyzers. Overall the challenge provided a large dataset, appropriate for a variety of mining tools and methodologies.

[1]Find all of the challenge data here: http://msr.uwaterloo.ca/msr2010/challenge/

II. Previous work

Since 2006, the MSR Mining Challenge has brought together researchers and practitioners who are interested in applying, comparing, and challenging their mining tools and approaches on a shared set of software repositories. The MSR Mining Challenge chairs and program committee have selected a set of open source projects as mining targets, then proposed a kind of emphasis for the challenge. Each year normally consists of two categories of challenges: general and prediction.

Prediction Challenge: The MSR literature is full of papers about predictor models based on the information obtained from software repositories. However, the real challenge is to publish and submit predictions before the events take place. In this category, the current edition of the challenge asked researchers to predict the evolution in the number of bugs reported in Debian. Past editions proposed to forecast the number of changes and bugs for Eclipse (2007, 2008), Firefox (2008), and code growth for the GNOME desktop suite (2009). The winner of the prediction challenge is selected based on the prediction accuracy.

General Challenge: The general challenge is a venue where researchers can submit any kind of short report about the selected target projects and repositories. The idea is to give useful feedback to the authors, so they appreciate the value of MSR research for practitioners. As well it allows current researchers to demonstrate state of the art mining techniques on a public dataset. Researchers can choose any tool and study any kind of public repository, although the MSR Challenge provides researchers with some already extracted datasets and repositories. In this manner MSR supports future research by bootstrapping many researchers with useful data, allowing them to skip the mirroring and extraction steps. The winner of the general challenge is chosen by the attendees of the workshop based on the quality of their presentation. Since 2008, winners of the

978-1-4244-6803-4/10/$26.00 © 2010 IEEE

MSR challenge are awarded a prize, usually a device like an iPod shuffle or a Zune HD, for their efforts and good work.

III. Data

To ease the mining process, raw data for the selected software projects are mirrored each year (2007-2009). The new data for this year's challenge is the FreeBSD distribution, and the Ultimate Debian Database. We also included the GNOME Desktop Suite data from the last challenge.

A. FreeBSD

FreeBSD [1] is an operating system based on Berkley's UNIX, the Berkley Software Distribution (BSD). This means that FreeBSD is a kernel and a userland. Userlands are collections of tools that a user can expect to be installed. FreeBSD acts much like a Linux distribution as it provides FreeBSD Ports [2], a system of maintaining third party packages that work on FreeBSD that are not explicitly part of the FreeBSD userland. Thus FreeBSD is a full and usable UNIX operating system that also provides a myriad of third party software to work with.

FreeBSD was chosen because it is a relatively confined software ecosystem. There are no other FreeBSD distributions (just BSD forks) and FreeBSD is centrally controlled using version control. Since the project operates FreeBSD Ports they also somewhat maintain and fix the third party projects that have been ported to FreeBSD. This is fundamentally interesting because it means a change in FreeBSD can cause changes in the Ports collection if new features or changing behaviour have an effect on Ports programs or vice versa.

Our FreeBSD data include the version control data, the mailing list data, and the bug tracker data. The original FreeBSD's version control system is Concurrent Versioning System (CVS) [3] and now the project uses both Subversion [4] (SVN) and CVS. CVS is popular with FreeBSD because they use CVSup [5] in order to mirror the FreeBSD CVS repository and the rest of the project's data (mailing list archives, bug tracker database, website HTML code). It is easy to set up a complete mirror of FreeBSD for research purposes using the CVSup protocol; the procedure is documented in the FreeBSD Handbook [2].

B. Debian/Ubuntu

Debian and Ubuntu are popular GNU/Linux distributions. They use a package format that is dependency aware. Often as one uses a Debian or Ubuntu system one wants to add more packages or remove packages, the package manager (dpkg) usually handles this and warns you if dependencies will be broken. Packages also evolve and their dependencies evolve over time as well. Since many of the problems that users face with a package are distribution specific they often have to turn to package maintainers or the distribution itself for help. For Debian and Ubuntu the bug tracker is used to allow users and developers to communicate and help resolve issues facing various installations. Luckily for MSR researchers, Debian has provided the Ultimate Debian Database [6].

The Ultimate Debian Database federates both the bug tracker and the historical dependency information into one database. The Ultimate Debian Database was provided by the Debian community, who has provided a front-end to the database that can be dynamically queried, a database schema for the UDD, and a database dump of the UDD. Thus for the mining challenge we simply mirrored the database dump of the UDD. The UDD has Debian and Ubuntu bug reports as well as historical information about the package configurations. We hoped that these package dependencies could be leveraged for some interesting analysis.

C. The GNOME Desktop Suite

The GNOME Desktop Suite [7] is the well known desktop environment for UNIX and UNIX-like systems. The GNOME brand is an umbrella for a myriad of different projects, that are integrated to form the desktop environment. Most of these projects are applications aimed at home computer users and developers of GNOME software. GNOME is also dedicated to usability and accessibility, although the adherence to these goals depends on the project.

We mirrored much of the data from these projects, but to make the challenge accessible to new researchers we had to do more than mirror the source of the data, we had to extract the data itself.

IV. Extraction

Extraction is where a repository is analyzed and facts about the underlying data are abstracted and stored in a database. Often extractors can do more computation and produce more facts about underlying data in the form of metrics or relations between other entities like authors and source code changes. By providing challenge authors with extracted data we effectively bootstrap their research so they do not need to do this resource intensive step themselves.

In the following two subsections, we explain how we proceeded with the extraction procedures for the cases of FreeBSD and GNOME. We did not perform any further processing on the Ultimate Debian Database, which was made available in the same form as it is provided by the Debian community.

A. FreeBSD

The *mailing list archive* datasets were obtained by applying MLStats [3] to the mboxes of all the FreeBSD mailing lists archives. The mailing list archives were obtained via the CVSup protocol and were in the form of compressed mboxes. These archives store all the messages, with rich

[2] See http://freebsd.org/doc/handbook/

[3] See http://tools.libresoft.es

headers, containing information about the authors and recipients of messages, dates and time zones, the user agent used to write and sent the messages, etc. The output of the tool is a relational database, that contains all the above information in a structured manner. We made all the databases available to the public for the MSR Mining Challenge.

We applied the CVSAnalY [8] tool to extract the *FreeBSD SVN and CVS repositories*. CVSAnalY extracts revisions, commits, and authors from source control repositories such CVS and SVN, and stores this extracted data in a database (PostgreSQL or SQLite). CVSAnalY also runs some basic source code metrics as well it rebuild commits from CVS revisions as CVS does not record the group of changes. CVS just stores the revisions to each file.

In addition, we provided an indentation extraction [4] for FreeBSD and FreeBSD ports that used in the indentation metrics papers by Hindle et al. [9]. The indentation extractor looked at revisions to source files, measured textual attributes relating to indentation, line length, code characters, and also provided estimations of the McCabe's Cyclomatic Complexity and the Halstead Complexity metrics of each revision.

Furthermore, we used *C-REX*, to extract the call graph information from FreeBSD source code repositories. C-REX is an evolutionary extractor, which uses lexical techniques (token based) to analyze source control repositories [10]. C-REX extracts facts, such as caller-callee relationships, from different revisions of the source code. Then these snapshot facts from adjacent revisions are compared against each other to identify program entities which have been added, removed, and modified.

To perform the evolutionary extraction, C-REX determines the number of revisions in each file of the FreeBSD project. Then, ctags [11] is invoked to parse the file and identify all of the entities, such as functions, that existed during the lifetime of FreeBSD. The individual entities are further analyzed and their content is categorized into code tokens and comment tokens. The code token, comment tokens and control keywords are compared for each pair of consecutive revisions and their code dependencies are recovered. The changes are annotated with their corresponding revision numbers, the name of the author and the date of the change. The C-REX output is stored in an XML database file. The output in the XML file is grouped by change-list and includes the following information for each change-list:

1) The source code entities added, removed or modified.
2) The location and type of each entity added, removed or modified.
3) The author and time of each change-list.

[4]Indentation extractor available at http://softwareprocess.es/index.cgi/-WhiteSpace

B. GNOME

GNOME projects are quite homogeneous, and share the same structure in regards to how they use their software repositories. This makes it easier to do cross-project empirical studies. However, due to the dimension of the GNOME set of projects, such kind of analysis might be costly and time consuming, not to mention the overload suffered in the server side, particularly in events like the MSR Mining Challenge, where researchers from all over the world try to gather the same data from the same repositories.

Due to this fact, the GNOME datasets were provided by the FLOSSMetrics project [12]. This project is gathering metrics about thousands of FLOSS projects, and made them accessible via both database dumps and even a web interface for easy visualization. This avoids the overloading a project's server, and it is also more convenient for MSR researchers, because it is easier to verify or replicate an empirical study with these common datasets.

There are two different kinds of datasets, both in the form of MySQL database dumps:

- SVN repository dump
- Mailing list archives dump

The SVN repository dump is obtained after applying CVSAnalY [8] to the Subversion repository of GNOME. In this repository, every project has its own module. So a dump is produced for every project. This dump contains the same information as the SVN log history. In particular, this includes information about the development activity (changes to files, developers involved), for the whole history of the project. The documentation of the FLOSSMetrics project contains details about the data schema used in the dumps and the information stored in them [5]. These datasets shared the same format as the FreeBSD SVN and CVS datasets, allowing for an easy cross-analysis between GNOME and FreeBSD.

The mailing list archive dumps are obtained after applying MLStats [6] to the mboxes of all the GNOME mailing lists archives. The archives of the GNOME mailing lists can be retrieved via their websites.

Again, the dumps contain all the mboxes headers information, but in a relational database, which makes it easier to extract and discover relationships between different agents within and across the GNOME projects.

V. SUBMISSIONS

Table I shows the breakdown of the type of project used and data analyzed in each paper, and whether or not they used our provided data. Those papers marked with "*" used our provided data. Among total 9 submitted papers for our general challenge category, 6 papers are accepted. 3 of these

[5]See the FLOSSMetrics wiki available at http://melquiades.flossmetrics.org/wiki/

[6]See http://tools.libresoft.es

Table I
BREAKDOWN OF THIS YEAR'S MINING CHALLENGE PAPERS

Paper	Project	Mined Data	Analysis
Luijten et al. [13]*	GNOME	Bug database	Exploratory
Sasaki et al. [14]	FreeBSD	Source code	Code cloning
Krinke et al. [15]	GNOME	Source code	Code cloning
Bougie et al. [16]*	FreeBSD	Bug database	Predictions
Davis et al. [17]*	Debian	Bug database	Exploratory
Mauczka et al. [18]	FreeBSD	Source code	Exploratory

6 accepted papers (50%) used our provided data. However, these papers used our mirrored raw data (e.g. bug database), rather than the fine-grained analyzed results. In the coming year, we hope more mining analysis will take advantage of our fine-grained processed results (e.g. C-REX data and indentation data).

VI. CONCLUSIONS

The MSR Mining Challenge provides researchers, both within and external to the MSR community, a chance to test their mining techniques and tools on a shared dataset. This year the focus was on FreeBSD source control, bug tracker, and mailing-list, GNOME source control and bug tracker and the Ultimate Debian Database, effectively Debian's configuration management system and a bug-tracker. Participants were asked to consider the task of mining information from a software ecosystem perspective and many did just that.

ACKNOWLEDGEMENTS

We would like to acknowledge the efforts of Israel Herraiz, Emad Shihab, and Christian Bird as they provided both data extraction and experience from previous challenges. We would also like to thank our generous program committee who made this track possible: Adrian Schroeter, Annie Ying, Emad Shihab, Emily Hill, Irwin Kwan, Israel Herraiz, Lile Hattori, Rahul Premraj, Zhen Ming Jiang, and Abram Hindle.

REFERENCES

[1] "FreeBSD," http://freebsd.org/.

[2] "FreeBSD Ports," http://freebsd.org/ports/.

[3] Free Software Foundation, "CVS," http://www.gnu.org/software/cvs/, 2004, accessed July 2005.

[4] CollabNet, "Subversion FAQ," http://subversion.tigris.org/faq.html, 2004, accessed July 2005.

[5] "CVSup Home Page," http://www.cvsup.org.

[6] "Ultimate Debian Database," http://udd.debian.org/.

[7] "GNOME Project Homepage," http://gnome.org/.

[8] G. Robles, S. Koch, and J. M. González-Barahona, "Remote analysis and measurement of libre software systems by means of the cvsanaly tool," in *In Proceedings of the 2nd ICSE Workshop on Remote Analysis and Measurement of Software Systems (RAMSS 2004)*, 2004, pp. 51–55.

[9] A. Hindle, M. W. Godfrey, and R. C. Holt, "Reading beside the lines: Using indentation to rank revisions by complexity," *Science of Computer Programming*, vol. 74, no. 7, pp. 414 – 429, 2009, special Issue on Program Comprehension (ICPC 2008).

[10] A. E. Hassan, "Mining software repositories to assist developers and support managers," Ph.D. dissertation, University of Waterloo, Waterloo, ON, Canada, 2004.

[11] "Exuberant Ctags," http://ctags.sourceforge.net/.

[12] I. Herraiz, D. Izquierdo-Cortazar, F. Rivas-Hernandez, J. M. Gonzalez-Barahona, G. Robles, S. D. nas Dominguez, C. Garcia-Campos, J. F. Gato, and L. Tovar, "FLOSSMetrics: Free / libre / open source software metrics," in *Proceedings of the 13th European Conference on Software Maintenance and Reengineering (CSMR)*. IEEE Computer Society, 2009.

[13] B. Luijten, J. Visser, and A. Zaidman, "Assessment of issue handling efficiency," in *7th IEEE International Working Conference on Mining Software Repositories (MSR 2010)*, May 2010.

[14] Y. Sasaki, T. Yamamoto, Y. Hayase, and K. Inoue, "Finding file clones in freebsd ports collection," in *7th IEEE International Working Conference on Mining Software Repositories (MSR 2010)*, May 2010.

[15] J. Krinke, N. Gold, Y. Jia, and D. Binkley, "Cloning and copying between gnome projects," in *7th IEEE International Working Conference on Mining Software Repositories (MSR 2010)*, May 2010.

[16] G. Bougie, D. German, and M.-A. Storey, "A comparative exploration of freebsd bug lifetimes," in *7th IEEE International Working Conference on Mining Software Repositories (MSR 2010)*, May 2010.

[17] J. Davies, H. Zhang, L. Nussbaum, and D. M. German, "Perspectives on bugs in the debian bug tracking system," in *7th IEEE International Working Conference on Mining Software Repositories (MSR 2010)*, May 2010.

[18] A. Mauczka, C. Schanes, M. Bernhart, and T. Grechenig, "Mining security changes in freebsd," in *7th IEEE International Working Conference on Mining Software Repositories (MSR 2010)*, May 2010.