The Benefits of Separating the Data Mining of Repositories from the Research Analysis Phase

CS680 Distributed Software Development Position Paper (Spring 2010)

Sammie Stahlback
Drexel University
3141 Chestnut Street
Philadelphia, PA 19104
sws28@drexel.edu

Jordan Osecki
Drexel University
3141 Chestnut Street
Philadelphia, PA 19104
jmo34@drexel.edu

Michael Kim
Drexel University
3141 Chestnut Street
Philadelphia, PA 19104
mk394@drexel.edu

1. ABSTRACT

There are many benefits from collecting history from software projects by mining software repositories. The data can be used to analyze which variables contributed to the success or failures of a particular task or possibly the whole project in a number of different categories. In general, much can be learned by both the developers of the software and the researchers from mining software repositories. Software repositories can give guidance to the developers regarding the current project and help researchers come to conclusions about more general results, benefitting future projects everywhere.

Historical information is typically mined from a wide range of sources. Because there is such a wide breadth of data sources to choose from, there is also a wide range of data formats that need to be handled. Each type of repository is stored in different ways and in different places. In addition, each project may change the way repositories are managed over time.

This results in historical information sources being hard to navigate and parse for any particular software project, and it is almost impossible to scale research results across multiple data sources. Because of the effort required for data mining, the conclusions and types of research that can be performed on software projects becomes very limited and the scalability of any conclusions can be questionable. As a result, research analysis is severely impacted by these constraints.

This position paper proposes that there is value in separating the data mining effort of software repositories from the research analysis process. Currently, data mining is performed alongside the analysis process, but there appears to be a need to develop data extraction techniques outside of the research itself because of the breadth and complexity truly involved in mining software repositories.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data mining.

General Terms

Design, Documentation, Experimentation, Management, Measurement, Reliability, Standardization, Theory.

Keywords

Distributed software systems, co-located software systems, mining software repositories, research analysis, repositories.

2. Introduction

There are many benefits from collecting history from software projects by mining software repositories. Data repositories include code repositories, bug reports, emails, and other communication archives.

Research similar to [14] highlights a common situation where the activities start with the development of tools used to extract data from the various data repositories. These extraction tools handle details such as connecting to the repositoiries, filtering out noise and unwanted data, and producing the relevant information for the research project. A large portion of the extraction process requires knowledge outside of the analysis process. Although the extraction process is not important during the analysis, the extraction tools can affect the results of the analysis (usually only in a negative manner) and may hinder the ability to reproduce the results. By having a common set of extraction tools or a common format to use, researchers can save time and increase the reliability of their results at the same time.

Along with developing tools, the researchers also make decisions regarding certain heuristics in order to make further connections within the data being extracted. The heuristics used within each research effort can make it difficult to reproduce the results or for others to build upon the results in future research efforts. Having data that's been extracted and stored in a common repository for all researchers to access can help expand research. It would expand research by having a common input to reliably compare models and algorithms with each other.

This paper will investigate existing tools and techniques that are currently available for mining software repositories to see if any can be incorporated into this new separate process. It will also analyze what these tools try to accomplish, the effectiveness of them, and what can be learned from them as the need for mining software repository tools continues to grow.

Treating the task of mining software repositories as its own discipline will help expand research in software development projects. Below are the ways in which this will help include the following benefits: it will help to create new "best practices" for extracting data and it will also help to promote the use of different common data formats, tools, schemas, and algorithms for others to easily pickup and also improve upon. As a result, it will allow researchers and even developers on the project being analyzed to more easily duplicate the results of others or expand upon their research results without having to reinvent the data extraction

process. Duplication and expansion on the results of research is important to help to spur innovation and make discoveries.

By allowing others to more easily build upon common extraction techniques, researchers will be less hampered by this process and no longer forced to gloss over pertinent parts of the research. It will also bring additional researchers into the area of both mining software repositories and analysis, improving on these techniques even further.

The rest of the paper will provide examples of common research techniques, ways to move toward separating the mining effort from the analysis effort, and the benefits of separating the two efforts. Section three outlines the common steps used while performing research and how data mining is a large portion of that effort. Next, in section four, the paper covers a case analysis that presents many of the different challenges related to data mining. Sections five and six highlight several tools and techniques used for data mining and the various levels of value they give to researchers. Finally, sections seven, eight, and nine focus on the benefits, conclusions, and future work of separating the mining and analysis portions of the research.

3. Current Method to Conduct Research

The current strategy typically used while performing research on software repositories is a simple model. This model consists of identifying the repositories, extracting the data, performing any additional heuristics on the data, performing the actual analysis on the data, and then presenting the results in a report or some other appropriate format.

First, identifying the repositories with the necessary data required to calculate the metrics is crucial. Since the data comes from several different repositories, the researchers must first become familiar with each of the data repositories. The researchers need to know the standards behind the different data formats and many other details. For example, a common repository used for mining is a project mailing list. Researchers need to learn the protocols used to communicate with mailing lists in order to truly understand the data and to form assumptions when beginning to mine it.

Once it is known how to talk to the repository, then the extraction process is performed. During this step, the format of data being extracted needs to be understood. This may require research into the associated RFC'(s) in order to determine what types of assumptions can be made while parsing the data. Without having the appropriate knowledge regarding the data format, the extraction process may exclude important data or include erroneous data that may affect the final results. The accuracy of the data at the extraction/parsing stage is absolutely important because errors during this stage will be magnified in the analysis stage. Continuing with the mailing list example, it is important to understand the required fields within the header information and the format of those fields.

Even after data is extracted and parsed, it is common for researchers to perform additional heuristics on the data to fit it to their liking. Heuristics are often an important step in order to show additional patterns in the extracted data or perform additional mappings that help generate more metrics in the final product. This is a step that may make it difficult for other researchers to extend upon the results and conclusions of the previous researchers, especially if these heuristics are not well defined for others to follow.

Finally, analysis can be performed on the data using the metrics of interest and the results can be presented or possibly stored in another repository for future use. As can be seen, this stage now seems like an after-thought, even though it is by far the most important stage of this process.

While understanding every detail in this process can be important and help researchers establish potential new avenues for research, the effort will extend the amount of time required to complete the research and can introduce errors into it as well. In addition, each research effort will end up duplicating these same efforts, just to get to a new and unique "analysis" stage.

4. Researcher Example: Data Mining

As a case study, the authors of this paper mined the mailing list archives of an open source database called MySQL. This is an example where the authors had to do both the data extraction and analysis portions of the project. Although it was a single research project, there were two distinct tasks that could have been done individually or even independent of one another. The team extracted the data from the mailing list and imported the output into a relational database.

The extraction process requires one to almost exclusively focus on learning the protocols and semantics of the mailing list. Otherwise, it became difficult to verify that the data was in fact accurate and the extraction process ran cleanly. Without this knowledge, incorrect assumptions can be made which would be fatal to the entire process.

The extraction process presented several challenges related to parsing mail files and understanding what fields could be obtained. Parsing the files was interesting and non-trivial since the data is more or less free-form text. Therefore, there was quite a bit of time dedicated to checking that the data was actually being read and was being outputted correctly.

In the author's research, the team imported the extracted data into a relational database after this parsing stage. Using a database instead of analyzing the data during extraction has a couple of benefits.

First, the speed in which you can query a large volume of data from a database versus a newsgroup or mailing list is much faster. This allowed the team to collect a large amount of data for the analysis step.

Second, the data is one step closer to being in a format that can be used to calculate other metrics outside of the team's own research. Once the schema is published, the next group of researchers can skip the process of learning the details about collecting mailing list archives and begin analysis on the data that is already stored in the database with a common database schema available.

Third, rather than being limited to tools that can read data in mailing lists, the team has expanded the number of tools that can be used to analyze the data. Finding tools that read from a database is much easier than finding tools that read from different mail headers.

Clearly, the team could have saved a lot of time and effort during analysis if the extraction process was already done and the team had started analysis with the database already populated. In addition, even just knowing the complexity of the mining/extraction task could have allowed the team to plan accordingly. Many teams underestimate the size of this step in the process of research that involves mining. Along with saving time,

the team could have possibly developed more models that could have been applied to our data as well.

5. Available Methods of Data Mining

To help reduce the amount of effort required during the extraction process, it will help if there were a group of standard tools that can be used so intimate details of the repositories can be hidden from the researchers. Some of the details that can be hidden are the network protocols being used, API's into the repository, and the exact syntax of the data in the repository.

The following tools and techniques show a few different ways that information can be hidden from researchers and will allow them more flexibility and time while analyzing data from software repositories. In addition, separating the data mining process can help individual research projects and help the software engineering research community grow as a whole.

5.1 CVSgrab

The first step towards building a community around data mining is establishing a set of tools that do not have to be written for each research project. CVSgrab, presented in [8], is an example of such a tool a tool that can be in this set.

In simple terms, CVSgrab is a web-based tool that can be used to pull data from CVS repositories. Data can be entered into a database for further analysis or used as input into another tool to display the results.

CVSgrab is an example of a tool that frees up researchers from having to understand how to talk with a CVS repository by hiding the network details. CVSgrab also hides internal details of the repository; therefore, the researcher can narrow their focus towards just the details of the CVSgrab tool itself. If CVSgrab was not available, then the researcher would have to develop a strong knowledge about how code is stored in CVS repositories and perform all the necessary data parsing each and every time, also taking the time to note nuances that will most definitely exist between different CVS repositories used on different projects.

While a set of tools like CVSgrab (One for SVN repositories as well, etc.) is valuable in saving time during data extraction, there are still limitations in research growth. These limitations are mainly due to the fact that a researcher would have to modify CVSgrab in order to extract any additional information that the tool could not provide.

The next tool presented will illustrate a framework that helps the mining community further and will hide details about the repository even further and allow for easier ability to extend the data model.

5.2 iSPARQL

Research done by [6] presents an interesting framework that extends past interfacing with any details of the software repository data and introduces an example for using exchange formats.

Much of the details presented in [6] are beyond the scope of this document, but it provides a concrete example of a framework that uses several standards developed from the World Wide Web consortium. The standards highlighted are SPARQL, used for retrieving and modifying data in Resource Description Framework (RDF) format, and Web Ontology Language (OWL) to define the semantics of the RDF format.

The initial steps follow the basic steps required for extraction, in that the data mined from the software repository is exported to the OWL format. After the original data is in an OWL formatted file, data is exchanged using the RDF format using iSPARQL queries. It is a powerful concept for a couple reasons. First, by using existing standards to exchange data, there is less additional software that has to be written in order to retrieve data once it has been exported to OWL. This perceivably opens the door for more researchers to duplicate results and perform research without having to write a bunch of additional code to extract the software repository data and convert it into a usable format. Second, now it becomes open and easier to develop heuristics that can be duplicated by others by simply extending OWL semantics.

This framework helps reduce complexities mentioned by [5] that inhibit researchers from participating in software mining.

The next section describes an example where both the tools and framework can be organized into a single community that can be shared and extended.

5.3 RESEARCH COMMUNITY: PROMISE

Research by [9] has used a tool called TracExtractor and has applied the Cox model in order to predict faults in software. While this is interesting, the item of importance in this research was that the researchers used data that was collected through a common site called PROMISE [10] instead of collecting the data on their own in an ad hoc manner.

PROMISE is a community site where researchers can publish tools and data collected from experiments. This site hosts a collection of data, tools, and results that other researchers can use for comparison with their own research. This type of community collaboration helps build experience around what models work best during analysis and whether the tools being used are accurate. It allows for the easy integration and introduction as well of new tools and technologies.

6. Advancements and Techniques for MSR

One of the more important advances in mining software repositories (MSR) is developing standards similar to those presented in the research of paper [11]. First, a standard for an exchange language should be defined in order to integrate data from many different sources.

The exchange languages that have been defined here attempt to cover a wide range of information. They have several common requirements that will help increase adoption within the MSR community. A few of the more interesting requirements are that the exchange language needs to be extensible in order to anticipate new paths of research. In addition, the exchange language should support such flags as quality codes to identify data that is more trust worthy than data that might have known issues or been created from special heuristics. Along this same topic, the exchange language should be able to support what types of algorithms were used for inferred information.

The next advancement in MSR is the formation of common areas or communities where extracted data is collected and shared. There are several collections of data that cover specific areas. PROMISE [10] contains data used in predictive models while FOSSology project [12] maintains data about software dependencies in order to organize the complex license dependencies within the open source software community.

Finally, extraction tools are being developed that hide much of the protocol details needed to interface with repositories but provide no additional research value. As a result, this frees up researchers

to focus on defining data and developing models for research rather than focusing on any kind of extraction processes.

7. The Benefits of Separation

Up to this point this paper has presented several ways that researchers can separate the effort of mining software data from the analysis process. Separating the two gives several benefits that are laid out in the following paragraphs.

Separating mining from analysis will open research to more people by removing the amount of knowledge required for entry into the process. Researchers will be able to focus on the data they are trying to model and not have to spread their focus on learning email, CVS, or bug reports in order to extract the data from them. The team also believes that adopting common standards will also invite more researchers into software research.

Reusing shared mining techniques will help produce reproducible results by storing data sets together and standards around how that data was collected. Along with reproducing results, researchers can also perform benchmark tests. Performing benchmark tests or comparing results is difficult if researchers are using different methods and heuristics while extracting data.

The benefits of an exchange language are that researchers only need to develop tools once in order to import/export data, rather than every researcher having to develop tools for the platform that they are storing data. Tools for extracting things such as RDBMS, xml, etc. can be developed once and then used by everyone.

Finally, a benefit it that it will allow a different set of participants to focus on becoming experts of and can focus on extraction and develop better tools that are accurate and cover a larger spectrum of data sources. As a result, expanding the number of data sources will also attract additional researchers or increase the number of models and algorithms.

8. Closed Source Projects

Much of the research examples this paper has covered have primarily focused on open source projects due to the availability of the data used for analysis. However, separating the mining process can also help include closed source projects into the research arena and allow internal workers ways to analyze as well.

There are many things that a business can do to prepare itself and its projects to take advantage of having the data mining process separate from the analysis work, listed below.

First, an organization may feel that a majority of the data stored in data repositories is too sensitive for public release. By performing the extraction and data mining separately, sensitive data can be made anonymous. For example, employee names can be replaced with numbers or file contents not important for the analysis being performed can be scrubbed or removed. Once the data is extracted, this opens up opportunities for multiple outside researchers to analyze the data instead of one set of researchers performing both the extraction and analysis.

A framework is also presented in [1] which an organization can use to integrate data from different sources within a corporate environment to be used to compare project data over time.

9. Conclusion

There has been a lot of valuable research that has taken advantage of data repositories containing a diverse set of software project data. In general, each research project is performing both the extraction process and the analysis effort. This model is less than optimal for a couple reasons. First, extraction tools are being reinvented for each project and this leads to results that are hard to extend or reproduce. Second, the data extraction process may introduce data that is difficult to use or inaccurate in future research that attempts to build on this previous research.

This paper has shown several areas where separating the mining process from the analysis will benefit the growth of software research. Increased participation occurs by removing the amount of knowledge required in order to perform analysis on data collected from software project. Researchers no longer have to understand details about the software repositories and instead can focus on creating reproducible research that can be extended by using standard data exchanges.

10. Future Work

In recent years there has been a push in advancing the field of mining software repositories (MSR). Outlined in [5] are the challenges of MSR and why it is important to approach this task separately as a stand-alone field. This section presents some of the future work that will help develop MSR as a separate field and help to increase research in software engineering.

One idea for future work is to continue to develop standards by building on those presented with TA-RE [11]. The benefits of establishing standards and increasing a wider adoption have been proven through such examples as the RFC process. A process similar to the one used for RFC'(s) could be applied to exchange languages and involve both mining experts and researchers.

Next, acceptance could be built within the software research industry by keeping access to the data simple and inviting.

A second idea is to continue to develop efficient and accurate tools, but move towards stressing exchange language standards.

The authors of TA-RE identified several ideas that seem to cover a majority of the requirements necessary for sharing data and tools within the software research community. However, it appears that TA-RE has not been that successful in being implemented. The team feels it would be useful to find out why projects like TA-RE haven't been successful and try to repair the errors in their ways.

A third place for future work is to continue to populate shared project data like PROMISE with more project repositories.

A fourth ideas is to increase the number of people performing MSR by expanding interest by finding new data sources. Open Archive Initiative [15] has a good model that separates data harvesters and data set exchange.

As more data becomes available for research, though, MSR needs to be able to scale to larger repositories.

A fifth idea is to continue to develop good relationships between MSR and researchers. The goal is not to automate the whole mining process; otherwise, there could be data not visible to the researchers that could be valuable.

Finally, as the MSR field grows, it will be important to keep transparency by stressing documentation standards.

11. ACKNOWLEDGMENTS

The team gives thanks to ACM SIGCHI for allowing the modification of templates they had developed.

The team would also like to thank Professor Valetto for helping to guide this position paper in the right direction and raise issues for the team to consider.

12. BIBLIOGRAPHY

[1] Alonso, O., Devanbu, P. T., Gertz, M., "Database Techniques for the Analysis and Exploration of Software Repositories", *First MSR Workshop*, ICSE 204, Scotland UK.

[2] Bird, C., Gourley, A., Devanbu, P., Gertz, M., Swaminathin, A., "Mining Email Social Networks", *International Conference on Software Engineering, Proceedings of the 2006 international workshop on Mining software repositories*, 2006, ACM.

[3] Canfora, G., Cerulo, L., "Fine Grained Indexing of Software Repositories to Support Impact Analysis", *METRICS, Proceedings of the 11th IEEE International Software Metrics Symposium*, 2005, IEEE Computer Society.

[4] German, D., Cubranić, D., Storey, MA, "A Framework for Describing and Understanding Mining Tools in Software Development", *International Conference on Software Engineering archive, Proceedings of the 2005 international workshop on Mining software repositories*, 2005, ACM.

[5] Hassan, Ahmed e., "The Road Ahead for Mining Software Repositories". *16th Frontiers of Software Maintenance*, FoSM 2008, Sept 30 – Oct 2 2008, IEEE Computer Society, pp. 48-57.

[6] Kiefer, C., Bernstein, A., Tappolet, J., "Mining Software Repositories with iSPARQL and a Software Evolution Ontology", *Proceedings of the 29th International Conference on Software Engineering Workshop*, 2007, IEEE Computer Society.

[7] Robbes, R., "Mining a Change-Based Software Repository", *ICSEW, Proceedings of the 29th International Conference on Software Engineering Workshops*, 2007, IEEE Computer Society.

[8] Voinea, L., Telea, A., "Mining Software Repositories with CVSgrab", *International Conference on Software Engineering, Proceedings of the 2006 international workshop on Mining software repositories*, 2006, ACM .

[9] Wedel, M., Jensen, U., Göhner, P., "Mining Software Code Repositories and Bug Databases using Survival Analysis Models", *Empirical Software Engineering and Measurement*, 2008, ACM.

[10] "PromiseData.org." *http://promisedata.org/*

[11] Kim, S., Zimmerman, T., Kim, M., Hassan, A., Mockus, A., Girba, T., Pinzger, M., Whitehead, E. J., Zeller, A., "TA-RE: An Exchange Language for Mining Software Repositories", *Proceedings of the 2006 international workshop on Mining software repositories*, May 22-23, 2006, Shanghai, China.

[12] Gobeille, R., "The FOSSology Project", *In Proc. Fith International Workshop on Mining Software Repositories*, 2008, pp. 47-50.

[13] Gonzalez-Barahona, M, J., "Collaboration Plan and Report", http://www.qualoss.org/deliverables/D6_5b_CollaborationReport_submitted.pdf, 2009.

[14] Crowston, K., Kangning, W., Li, Q., Howison, J., "Core and Periphery in Free/Libre and Open Source software team communications", *Proceedings of the 39th Hawaii International Conference on System Sciences*, 2006.

[15] "OpenArchives.org." *http://www.openarchives.org/*