Report on MSR 2004: International Workshop on Mining Software Repositories

Ahmed E. Hassan and Richard C. Holt
Software Architecture Group (SWAG)
School of Computer Science
University of Waterloo
Waterloo, Canada
{aeehassa, holt}@plg.uwaterloo.ca

Audris Mockus
Software Technology Research Department
Avaya Labs Research
NJ, USA
audris@research.avayalabs.com

ABSTRACT
A one-day workshop was held on the topic of mining software repositories at ICSE 2004 in Edinburgh, Scotland. The workshop brought together researchers and practitioners in order to consider methods that use data stored in software repositories (such as source control systems, defect tracking systems, and archived project communications) to further understanding of software development practices. We divided submissions into six sessions, each devoted to a particular topic: 1) Infrastructure and Extraction, 2) Integration and Presentation, 3) System Understanding and Change Patterns, 4) Defect Analysis, 5) Process and Community Analysis, and 6) Software Reuse. To maximize interaction and discussion, we limited each session to a survey of the topic area, followed by the presentation of one or two papers, then an open discussion. We also allocated a demo hour to give interested parties the opportunity to learn more about other accepted papers.

This report includes an overview of the presentations made during these sessions and a summary of the issues raised throughout the workshop.

1 INTRODUCTION

Software repositories contain a wealth of valuable information for empirical studies in software engineering, including:

- *source control systems* store changes to the source code as development progresses,
- *defect tracking systems* follow the resolution of software defects, and
- *archived communications* between project personnel record rationale for decisions throughout the life of a project.

Such data are available for most large software projects and represent a detailed and rich record of the historical development of a software system. Participants in multiple sites, often on multiple continents, develop software projects without ever meeting in person, as is the case in many large commercial and Open Source projects. This fact makes the use of tools to record all aspects of software project more critical.

Until recently, data from these repositories was used primarily for historical record supporting activities such as retrieving old versions of the source code or examining the status of a defect. Several studies have emerged and used this data to study various aspects of software development such as software design/architecture, development process, software reuse, and developer motivation. These studies have highlighted the value of collecting and analyzing this data. Each of these studies has built its own version of methodologies and tools to address the formidable challenge of utilizing such data to perform their empirical research. Several international efforts have identified the development of approaches to extract, share, and study this data as a research priority.

The goal of this one-day workshop is to bring together researchers, practitioners, and source control systems developers in order to consider methods that use the data stored in software repositories to further understanding of software development practices. We hoped that the presentations and discussions during the workshop would facilitate the definition of challenges, ideas and approaches to transform software repositories from static record keeping repositories to active repositories used by researchers to gain empirically based understanding of software development, and by software practitioners to predict and plan various aspects of their project.

2 SCOPE AND TOPICS OF INTEREST

We sought position papers that are no more than 5 pages long and which cover relevant topics such as but not limited to:

- New approaches which analyze the data stored in software repositories to:
  - Assist in program understanding and visualization
  - Predict and gauge the reliability and quality of software systems
  - Study the evolution of software systems
  - Discover patterns of change and refactorings
  - Understand the origins of code cloning and design changes
  - Model software processes for development, defect repair, etc.
  - Assist in project planning and resource allocation
- Case studies on extracting data from these repositories for large long lived projects

- Proposals for common exchange formats, meta-models, and infrastructure tools to ease the sharing of the extracted data and to enable reuse and repeatability of results throughout the research community
- Suggestions for particular large software repositories to be shared among the community for research evaluation and benchmarking purposes
- Approaches to integrate data between repositories and with other software project data such as static or dynamic analysis data
- Requirements and guidelines for users and developers of source control systems to ease the analysis of the stored historical data

3 WORKSHOP FORMAT

We received 38 submissions from 14 countries. Papers were reviewed by the workshop's program committee in terms of their relevance to the aims of the workshop and their technical content. Due to the large number of submissions, submitting authors were asked to assist in the reviewing process as well. We selected 26 papers for publication in the workshop's proceedings. Four papers were invited to a special issue of the IEEE transaction on Software Engineering on mining software repositories. Accepted papers were posted on the workshop's web site prior to the workshop at:

`http://msr.uwaterloo.ca`

The workshop program was broken into six sessions, each devoted to a particular topic:

- Infrastructure and Extraction,
- Integration and Presentation,
- System Understanding and Change Patterns,
- Defect Analysis,
- Process and Community Analysis,
- Software Reuse.

To maximize interaction and discussion, we limited each session to the presentation of one or two papers, followed by a survey of the topic area, then an open discussion. We also allocated a demo hour to give interested parties the opportunity to learn more details about other accepted papers. A wrap-up session at the end of the day established goals for further research in the area of mining software repositories

The organizers chose papers to be presented that could serve as the basis for fruitful discussions. Presenters were asked to keep their presentations short to ensure ample discussion time. A discussion leader was assigned for each topic to provide a (10-15 minutes) critical review of the key concepts in the topic area, notably those concepts given in the accepted papers on the particular topic, with emphasis on what is good or promising and what is weak or doubtful. Presenters were asked to coordinate with their discussion leader ahead of time to avoid repetition.

Authors in the topic area were encouraged to assist their discussion leader by participating actively in the discussion. In particular, they were asked to raise and elucidate key issues. We hoped this would lead to lively open discussions.

4 WORKSHOP SESSIONS

We now give an overview of the presentations and elaborate on the issues raised in each session of the workshop. Papers and presentations for each session are available online at the workshop's web site.

Session 1: Infrastructure and Extraction

The Infrastructure and Extraction session focused on the engineering challenges related to the infrastructure and tools needed for recovering useful data from software repositories. The discussion leader for this session was Daniel German from the University of Victoria, Canada. German described the extraction phase as the "*dirty work*" that must be done with a purpose in mind, otherwise the extracted data will have no value. He highlighted the fact that much of the work in this area is based on repositories from open source projects due to the accessibility of these repositories online.

Papers in this session covered the extraction technique for a variety of software repositories. In addition to papers focusing on source code and bug tracking systems, a paper by Dekhtayar $et\ al.$ talked about preliminary results from mining textual requirement documents. They cautioned of the risks of mining natural text while highlighting the benefits of augmenting software repositories with natural language text used to develop the software. Another paper by German discussed the benefits of mining source control repositories. These repositories reveal a great deal of information about a project. They uncover the process followed, the evolution of the architecture, and hidden dependencies among files in a project. A paper by Howison and Crowston cautioned of the perils and pitfalls of mining project data available online. The paper outlined practical lessons gained from spidering, parsing and analysis of SourceForge data.

Two papers were presented in this session - one by Gasser $et\ al.$ and another by Zimmermann and Weißgerber. Gasser $et\ al.$ drew attention to a new challenge faced by empirical studies; whereas previous studies suffered from lack of data, current studies face challenges dealing with enormous amounts of freely available data from easily accessible repositories online such as forums, code, and bug reports repositories. There exists no simple means for assembling this data under common access points to enable collaborative research. There exists no frameworks for comparative and longitudinal research. The paper identified several common needs and recommendations for the design of a shared research infrastructure in order to encourage comparative and collaborative work in the field. A presentation by Zimmermann discussed an example of a technique used to clean up source control data (stored at the line level) to permit fine grained analysis (at the function level). Similar techniques

have been implemented independently by a number of researchers.

The discussions in this session focused on the risk associated with mining repository data and the unknown quality of the recovered data. Participants felt that the data extraction process plays a central and a critical role in all the topic areas in MSR. The quality of the extraction should be compared possibly through community defined benchmarks. Some participants proposed attaching statistical quality metrics to derived data from software repositories in order to give an indication of the quality of the extracted data to its potential users. Furthermore, participants felt that having standardized extractors to be shared across the community would permit others to join in the analysis and study of data derived from software repositories by reducing the entry barrier and making the stored data more accessible.

Session 2: Integration and Presentation
The Integration and Presentation session focused on techniques for integrating mined data from various historical sources. The session also covered the approaches to visualize and present the results derived from the mined data. The discussion leader for this session was Katsuro Inoue from Osaka University, Japan. Inoue presented the objectives of the research described in the four papers in this session. Inoue covered the approaches used by each paper, while commenting on the strength and weakness of each approach. Approaches by Ohira $et\ al.$ and Alonso $et\ al.$ suggested the integration of recovered data and their storage of in database systems using standardized schemas to permit the usage of traditional database mining infrastructure. Papers by Robles $et\ al.$ and Liu $et\ al.$ showed simple metric plots and summarization tables that are derived from CVS data.

The paper by Liu $et\ al.$ was presented in this session. The paper described the results of a case study that monitored the usage of CVS systems by students in a software engineering course at the University of Alberta, Canada. A system called JReflex is developed to assist instructors in tracking the progress of the students in the development of a term project. The goal of the system is to help instructors understand how students interact, and to find out if there is any correlation between their grades and the nature of their collaboration patterns. A good understanding of these factors will allow instructors to direct their attention to potential problem areas and teams early in the course.

The discussions in this session focused on the scalability of data presentation approaches. Most papers focused on simple visualization and more elaborate visualizations were not investigated. Participants suggested the need for more case studies to determine the most suitable visualizations for particular tasks. Also, participants discussed the feasibility of using databases to store the recovered data in order to permit the adoption of traditional data mining techniques. Furthermore, standardized exchange formats (through databases or XML schemas) would ease the sharing of data between researchers in the community but might cause additional translation overhead during analysis of the data.

Session 3: System Understanding and Change Patterns
The System Understanding and Change Patterns session focused on approaches to use the patterns of usage, change and refactorings to assist in future development. The session also covered approaches to visualize and present the results derived from the mined data. The discussion leader for this session was Annie Ying from IBM Research. Ying described the wisdom and knowledge that exists in software repositories and how such information can be used to support program understanding. A paper by Van Rysselberghe and Demeyer described mining source control repositories for Frequently Applied Changes (FAC). These FACs could be used to guide software developers as they maintain large systems instead of depending on their gut feelings and experience. Papers by Sayyad Shirabad $et\ al.$ and Ying $et\ al.$ reported results on techniques for recovering dependencies between software artifacts. The dependencies are later used to assist developers changing the software system by suggesting other software artifacts that should be changed. A paper by El-Ramly and Stroulia proposed mining software usage data to discover interesting patterns of user activities from system-user interaction traces. These interaction patterns are used to assist in reengineering legacy systems to web based application or to personalize legacy systems.

A paper by Godfrey $et\ al.$ was presented in this session. The paper briefly outlined four types of studies that were conducted by the authors to understand various aspects of a software system's evolutionary history. Each study involved detailed examination of facts extracted from a variety of source artifact repositories. A set of tools were developed to aid in the extraction, abstraction, and comprehension processes. The presentation highlighted the main challenges faced by the authors in these studies. The authors had to deal with a large amount of data using research tools which are usually slow un-optimized prototypes and may not be robust enough therefore requiring handholding.

The discussions in this session focused on the granularity of the data stored in software artifacts. In many cases information about the software system is stored at the file level (for example, CVS tracks changes at the file level). To report results at a higher granularity such as functions, further analysis is required. Furthermore, the higher the level of granularity, the more data that must be processed and studied during the analysis. Automating the analysis is likely to help in the successful adoption of tools that mine software repositories, since it will reduce the required human efforts.

Session 4: Defect Analysis
The Defect Analysis session focused on modeling defects and software reliability through mined data from software repositories The discussion leader for this session was Thomas Ostrand from AT&T labs. Ostrand covered briefly

the papers in this session which focused on a variety of aspects surrounding the reliability of large software systems. A paper by Sandusky $et\ al.$ proposed visualizing the relations between bug reports by creating bug report networks which show the relation between bugs. Such networks give some insight in the strategies used to resolve bugs in large open source projects. A paper by Menzies $et\ al.$ showed that code and defect logs could be combined to build models to assist developers in allocating testing resources. The paper indicated that static code metrics are not sufficient in predicting the reliability of a software system. The idea of combining static metrics and change/defect history to develop better fault predictors was explored in another paper in this session by Ostrand and Weyuker.

Two papers were presented in this session - one by Purushothaman and Perry, and another by Williams and Hollingsworth. Purushothaman and Perry reported on a study investigating the risk associated with one-liner small changes using data from a large long lived telephony system. The study revealed that there is less than 4 percent probability that a one-line change will introduce an error in the code and that nearly 10 percent of all changes made during the maintenance of the software under consideration were one-line changes. Williams and Hollingsworth described a method of creating tools to find bugs in software. Their method is driven by the analysis of previous bugs. By identifying patterns that have resulted in previous bug fixes, the authors could improve the false positive rate of static checkers.

Discussions in this session focused on the potential benefits of building models that make use of characteristics of previous source code change to predict bugs instead of simply using static complexity measures. Results reported in papers in this session along with recent research results are good indicators of the benefits of enriching reliability models with change history information.

Session 5: Process and Community Analysis

The System Process and Community Analysis session focused on mining software repositories to gain insight into software development team organization and structure. The discussion leader for this session was Chris Jensen from the University of California, Irvine, USA. Jensen highlighted the benefits of performing social network analysis to discover how software development takes place in large projects such as open source ones. A paper by Jensen and Scacchi recovering a task-oriented view of the process followed by developers to assist them in maintaining large software systems. Another paper by Schneider $et\ al.$ described a tool called Project Watcher. The tool tracks the local interaction histories between a developer and a development environment. The interaction history could assist in a variety of circumstances,such as the coordination of team activities and identification of code browsing habits.

A paper by Lopez-Fernandez $et\ al.$ was presented in this session. The paper proposed the use of social network analysis for characterizing the interaction between developers in open source projects. Two types of networks were created from mining source control repositories: the committer network and the module network. The committer network has as each vertex a developer (committer). An edge exists between two developers if they both contributed to at least one common module. The module network has as each vertex a module. An edge exists between two modules if they both have a common committer. The paper showed graphs of the module network as it evolved over time for the Apache web server.

The discussions in this session focused on the fact that some interactions cannot be tracked through software repositories as they only contain limited information on the development process (instead approaches such as Schneider $et\ al.$ may be beneficial). Also, it was pointed out that not all projects use source control systems in the same manner. This fact may complicate the comparison of results between software projects. Finally, the ethical issues surrounding mining software data to determine the productivity of developers was raised and it was pointed out that similar issues are faced by other researchers such as medical researchers.

Session 6: Software Reuse

The Software Reuse session focused on the discovery of techniques to facilitate software reuse in large teams. The discussion leader for this session was Pankaj Garg from Zeesource, USA. Garg emphasized the benefit of software reuse for large software projects. A paper by Garg $et\ al.$ presented an approach for developers to benefit from multi-project software knowledge. For example, if a component is replaced with another probably better implementation within a project; this knowledge can be shared with all relevant projects. Papers by Yusof and Rana, and McCarey $et\ al.$ presented information retrieval approaches to assist developers searching for code in large code repositories. These approaches could improve code reuse. Another paper by Amin $et\ al.$ investigated if the names of classes and method names could be used to assist in locating component for reuse; as developers are likely to choose class, method and variable names that are lexically expressive of the goals of the code.

The paper by McCarey $et\ al.$ was presented in this session. The presentation highlighted the potential for reuse to improve software productivity and pointed out the difficulty of locating components to reuse especially as the size of the component repository grows. A technique based on collaborative filtering was used to recommend components for reuse.

The discussions in this session focused on the large amount of code available online nowadays and the need to build Google like tools to assist in locating components to reuse.

5 CONCLUSION

The six sessions of the workshop have helped in defining the

structure of the software repository mining field. The issues raised in the papers and discussed in the sessions are a good starting point to define the important research directions the field should take, the research needs of the field, and possible future benefits of mining such repositories.

We hope that the presentations by the discussion leaders for each session have provided workshop participants with an overview of the important emergent issues in the field, and that the workshop has assisted in identifying potential future research collaborators. The papers and presentation slides are available online at the workshop's web site.

Since MSR 2004 was the first workshop to focus on the field of mining software repositories, it was exploratory in nature with a variety of results in different areas of the field. There will be follow-up workshop on Tuesday May 17 2005 at ICSE 2005 in Saint Louis, Missouri, USA. We expect the MSR 2005 workshop to retain an exploratory feel while promoting collaboration and systematic comparison of approaches and techniques. This mixture of exploratory work with systematic comparisons and collaborations should assist in maturing the field of mining software repository and in directing it to assume a central role in supporting software development practices.