Context Analysis of Historical Process Data with the Project Replayer

Kimiharu Ohkura[1], Paul S Grisham[2], Hajimu Iida[3], and Dewayne E. Perry[4]

[1,3]*Software Design Laboratory*
Graduate School of Information Science,
Nara Institute of Science Technology, Japan
kimiha-o@is.naist.jp, iida@itc.naist.jp

[2,4]*Empirical Software Engineering Laboratory*
Electrical and Computer Engineering
The University of Texas at Austin
{grisham,perry}@ece.utexas.edu

Abstract

In many software projects, mistakes are often repeated due to knowledge that has been forgotten from past practice. In order to capture such knowledge, recording, visualizing and identifying underlying contexts and process phenomena in project data is very important. However, directly inferring such contexts from software documents or formal reports and extracting relevant process phenomena from archives of raw process data is very difficult.

To assist in understanding development projects, we use the Project Replayer, a tool for visualizing many types of historical process data using natural language processing techniques and cluster analysis on project communication, e.g., email records. To validate the utility of Project Replayer, we attempted to visualize data from the Bell Labs 5ESS project. This paper presents our early experiences in using Project Replayer to understand a large, historical development project.

1. Introduction

Project postmortem analysis is recognized as critical to improving a software development process. Unfortunately, sufficient analysis is rarely performed because of time constraints and the general difficulty of analyzing massive amounts of data and identifying the underlying contexts for the project. Too often, development teams refocus quickly on the next project without adequate introspection on the successes and failures of the previous project.

The Project Replayer [1] is a tool to visualize project data collected by the Empirical Project Monitor [2] in order to assist in understanding a project, either incrementally, through the ongoing accumulation of process data, or retrospectively, by visualizing historical data. Developers can use the Project Replayer to revisit their past projects for postmortem evaluations, while researchers can use the Project Replayer to understand and analyze dynamic behavior. Figure 1 presents a screenshot of the various views and analysis tools available in the Project Replayer.

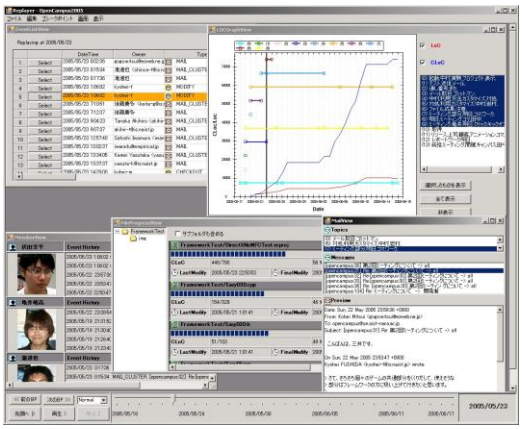

Figure.1 Screenshot of the Project Replayer

In previous studies, we applied the tool to small and medium scale projects to examine characteristic process phenomena from email archives. This paper presents early results from an ongoing study that applies the tool to the Bell Labs 5ESS project to validate the use of Project Replayer on historical data.

2. Analysis Method

We propose a method to capture project development context from email archives and trouble reports collected automatically by development tools such as Mailman or GNATS. Email is widely used by developers to communicate with each other during a project. Email archives contain many contexts about the development, such as notifications of program code modification, negotiations for product specification change, and other interactive communications. Trouble report archives also contain various contexts for specific problems and change requests occurred during software development.

Generally, it is very difficult to extract relevant process phenomena from archives of raw process data. Since both emails and trouble reports are produced in large number during development projects, our method classifies the archives into some featured topics based on natural language processing and clustering algorithms.

In order to classify the archives, each document is represented by a document vector which consisted feature term frequencies. After vector conversion, similarities among each vector are calculated based on the Vector Space Model. Also, based on the result of the calculations, each archive which represented as a vector is classified using clustering algorithm. As a result, clusters containing similar documents are obtained. These can be assumed to be the set of conversation topics in the project. To visualize the topics, the clusters are plotted into time-series charts, and offer a quick overview of the project contexts.

By overlapping clusters with other time-series charts such as growth of LoC, characteristic process phenomena can be visualized, and underlying contexts can be unveiled. More details of these analysis and graphic visualization features are described in [3].

3. The 5ESS Project

The Bell Labs 5ESS Project is a very large scale project that has previously been used as the basis of many research studies [4]. The subsystem of 5ESS used in this project represents 15 years of development on one module of 5ESS. The raw project data contains 38,500 issue records and over 160,000 source code changes. We extracted this data and converted it into the native XML format used by Project Replayer.

One interesting characteristic of the 5ESS project is that email communication was largely rejected by the developers [5]. Developer communication was handled by informal meetings, and email was used only as a broadcast medium for group coordination. As a result, we were forced to use Project Replayer's visualization modules without the benefit of contextual analysis from email communication in the project.

4. Results and Conclusion

By applying the Project Replayer to the 5ESS project data, we were able to visualize a summary of issues of the project. However, because there is so much historical data from the 5ESS project, it was necessary to pre-filter the data by module and date range. Project Replayer cannot currently manage the entire 5ESS data archive. A 3 month plot of the 5ESS project data is in Figure 2. Issues are represented the scattered rectangles on code-growth graph.

The lack of email communications for the 5ESS project proved to be a difficult obstacle. Much of the email communication used in previous studies of the Project Replayer was automated messages generated by the tooling, and this type of communication is easily reproducible from the 5ESS data. Issue negotiation and resolution used a formal process with the Extended Change Management System (ECMS) to track and audit changes.

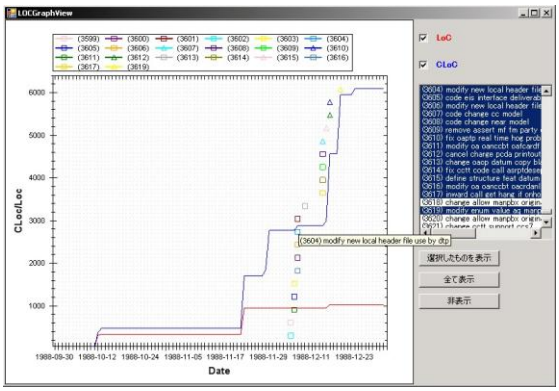

Figure.2 An example of visualizing the issues

Unfortunately, negotiations conducted through informal meetings are lost forever. We are using this opportunity to consider how other forms of communication can be included in EPM and Project Replayer, such as design history logs and other, new informal communications, such as instant messaging.

Acknowledgements

We would like to thank Prof. Noriko Hanakawa of Hannan University and Assistant Prof. Shinji Kawaguchi of NAIST for their valuable advice.

This research was supported in part by the Japan Ministry of Education, Culture, Sports, Science and Technology, Grant-in-Aid for Scientific Research (C) 17500024, the EASE project in Comprehensive Development of e-Society Foundation Software program of the Japan Ministry of Education, Culture, Sports, Science and Technology, and also by the JSPS 2007 Summer Program, the NSF 2007 East Asia and Pacific Summer Institute (EAPSI) Program.

References

[1] K. Goto, N. Hanakawa, and H. Iida, "Project Replayer An investigation tool to revisit processes of past project", In Proceedings of SPW/Prosim2006, LNCS, May 2006.

[2] M. Ohira, R. Yokomori, M. Sakai, K. Matsumoto, K. Inoue, M. Barker, and K. Torii, "Empirical Project Monitor: A System for Managing Software Development Projects in Real Time", In Proceedings of ISESE2004, Aug 2004.

[3] K. Ohkura, K. Goto, N. Hanakawa, S. Kawaguchi, and H. Iida, "Project Replayer with Email Analysis - Revealing Contexts in Software Development", In Proceedings of APSEC2006, Dec 2006.

[4] R. Purushothaman and D.E. Perry, "Towards Understanding the Rhetoric of Small Source Code Changes", IEEE Trans. on Software Engineering, 31(6), June 2005.

[5] D.E. Perry, N. Staudenmayer, and L. Votta, "Understanding and Improving Time Usage in Software Development", In *Trends in Software: Software Process*, Volume 5, 1996.