Empirical Project Monitor: A System for Managing Software Development Projects in Real Time

Masao Ohira[1], Reishi Yokomori[2], Makoto Sakai[3],
Ken-ichi Matsumoto[1], Katsuro Inoue[2], Michael Barker[1], Koji Torii[1]
[1] *Nara Institute of Science and Technology*
ohira@empirical.jp, {matumoto, mbarker, torii}@is.naist.jp
[2] *Osaka University* *{yokomori, inoue}@ist.osaka-u.ac.jp*
[3] *SRA Key Technology Laboratory, Inc.* *sakai@sra.co.jp*

Abstract

The poster illustrates our ongoing research project (EASE: Empirical Approach to Software Engineering) and a prototype system called the Empirical Project Monitor (EPM). EPM automatically collects and measures history data accumulated during everyday software development activities. By providing graphical views of measurement results, EPM helps developers and managers keep their projects under control in real time without additional work.

1. Introduction

In software development in recent years, the improvement of software process has increasingly gained attention. Project management for achieving effective software process improvement should be based on quantitative data. The Goal-Question-Metric (GQM) paradigm [1] provides a sophisticated measurement framework for collecting software metrics. Although software measurement based on the GQM paradigm is reasonable, to practice it stakeholders need to develop measurement processes. In general, data collection for software measurement requires high costs and additional effort from developers and managers.

Our EASE (Empirical Approach to Software Engineering) project [2], in collaborative research with software companies, has developed a prototype software engineering environment for supporting measurement, analysis, evaluation, and feedback during the last year. In the poster, we introduce the Empirical Project Monitor (EPM) [3] which currently automatically collects and measures quantitative data from three kinds of repositories widely used in software development support systems such as configuration management systems, mailing list managers and issue tracking systems. By providing integrated measurement results graphically, EPM helps developers and managers keep their projects under control in real time.

2. Empirical Project Monitor (EPM)

Figure 1 shows an overview of EPM. EPM consists of four components: data collection, format translation, data store, and data analysis/visualization.

First, EPM automatically collects versioning histories from configuration management systems, mail archives from mailing list managers, and issue tracking records from issue tracking systems. Because these data are accumulated through everyday development activities, developers and managers do not need to do additional work for data collection. Second, EPM converts the collected data into the XML format, so that EPM can deal with not only the above three kinds of history data but also other kinds of data according to the purposes for measurement. Data from other systems can be included by small adjustments of data collector and format translator. Third, the data is stored in a PostgreSQL database. Finally, EPM analyzes the data stored in the database and then visualizes various measurement results.

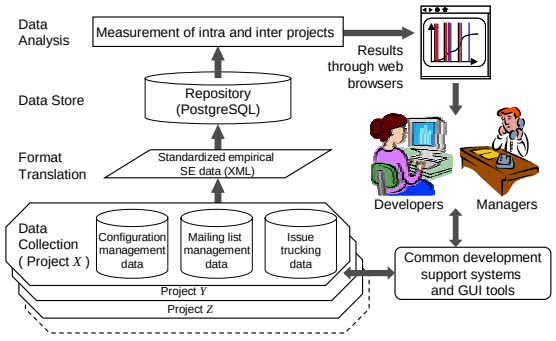

Figure 1. Overview of EPM

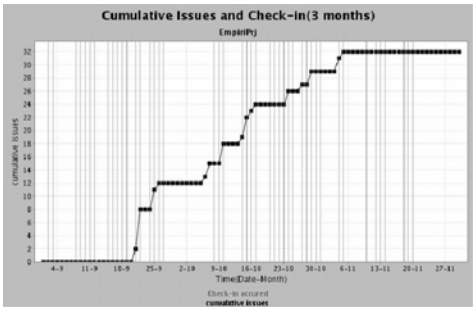

Figure 2. Issues and checkins

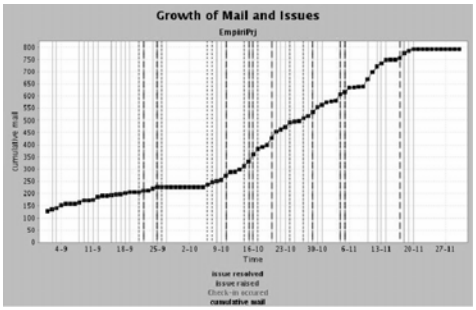

Figure 3. Issues and e-mails

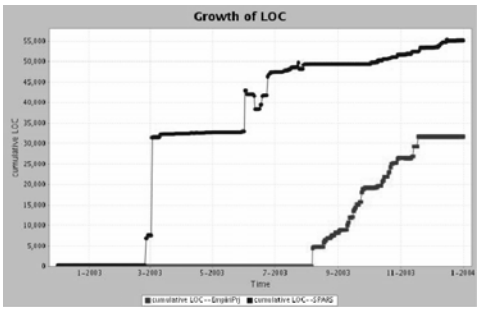

Figure 4. Comparison of two projects

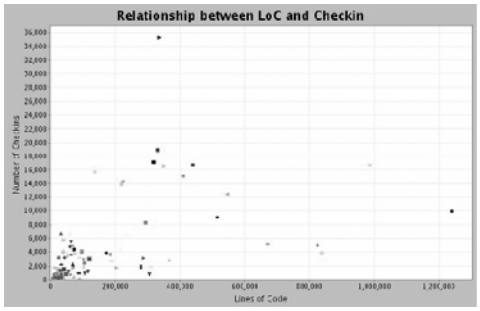

Figure 5. Distribution map

3. Combinations of measurement results

EPM provides users with a variety of visualizations of measurement results based on the three kinds of history data currently available. The features of EPM are to visualize combinations of measurement results and to deal with data from multiple projects simultaneously.

For instance, Figure 2 represents the relationship between changes in the cumulative total number of issues (the line graph) and the time of checkins (the grayed vertical lines on X-axis). The number of issues is measured using data from an issue tracking system. The number of checkins is from a configuration management system. This allows developers to confirm which file versions include critical bugs. Figure 3 illustrates a communication history among developers. The black line graph is the transition of the cumulative total of e-mails exchanged through a mailing list manager. The vertical shorter/longer dashed lines represents when bug issues were raised/resolved. The light-gray vertical lines indicate when the checked-in files by developers were uploaded to the configuration management system.

EPM also makes measurement results comparable between multiple projects. Figure 4 represents the relationship of the growth of lines of code between two projects. Such the graph could help people who have to manage multiple projects at the same time or for comparing productivity. EPM can also generate a distribution map such as in Figure 5. The graph shows a distribution map of 100 projects, which represents the relationship between lines of code (X-axis) and number of checkins (Y-axis). Such a graph can help managers identify "unusual" projects with extremely high or low values.

4. Future Research

EPM provides a framework for collection and analysis of empirical data. The EASE project is looking at extending this with tools such as collaborative filtering, Bayesian nets, and other modeling and analysis tools. We are working with industry to determine what kind of support allows the best application of empirical methods for software process improvement.

5. References

[1] V. R. Basili and D. M. Weiss, "A methodology for collecting valid software engineering data", *IEEE Transactions on Software Engineering*, Vol.SE-10, No.6, pp.728-738, 1984.

[2] The EASE Project, http://www.empirical.jp/

[3] Masao Ohira, Reishi Yokomori, Makoto Sakai, Ken-ichi Matsumoto, Katsuro Inoue, Koji Torii, "Empirical Project Monitor: A Tool for Mining Multiple Project Data", *International Workshop on Mining Software Repositories (MSR2004)*, pp. 42-46, Scotland, UK, 2004.