A Proposal for Analysis and Prediction for Software Projects using Collaborative Filtering, In-Process Measurements and a Benchmarks Database

Yoshiki Mitani[1,2], Nahomi Kikuchi[1], Tomoko Matsumura[2], Naoki Ohsugi[2], Akito Monden[2], Yoshiki Higo[3], Katsuro Inoue[3], Mike Barker[2], Ken-ichi Matsumoto[2]

[1]IPA/Software Engineering Center (SEC), Tokyo, JAPAN
{y-mitani|n-kiku}@ipa.go.jp
[2]Nara Institute of Science and Technology (NAIST), Nara, JAPAN
{ymitani|tomoko-m|naoki-o|akito-m|mbaker|matumoto}@is.naist.jp
[3]Osaka Univ., Osaka, JAPAN
{y-higo|inoue}@ist.osaka-u.ac.jp

Abstract. This paper proposes a new method for developing predictions and estimates for ongoing projects by comparing in-process measurements of the current project with benchmark data from previous projects. The method uses collaborative filtering to identify groups of similar projects in the benchmark database and then to develop predictions and estimates based on the in-process measurements of the current project and the comparison data from the similar projects. The authors base this proposal on experiments with multidimensional in-process project measurement in a middle-scale multi-vendor development that lacked transparency in its processes. The authors' measurement trial verified the usefulness of the measurement methods, especially in project management, as reported in the paper.

Keywords: Empirical Software Engineering, In-process project measurement, Collaborative filtering, Software project database.

1 Introduction

This paper first provides a bird's-eye view of past research by the authors and fundamental methods. Then the paper explores a new project measurement and feedback method which has evolved from that past research.

Specifically, the paper describes the function and structure of a project measurement platform called the Empirical Project Monitor (EPM). The Empirical Approach to Software Engineering (EASE) project, an academic based project for collaboration between industry and academia, developed EPM [1][2][3][4]. Next, it presents experimental results from the application of EPM and related tools in a governmental project for multi-vendor software development called the Advanced Software Development (ASD) project [5][6]. After that, it presents a project to collect

benchmark data from software projects which has collected data from over 1000 projects in 15 software companies [7]. The Software Engineering Center (SEC), an industry-based organization for collaboration of industry and academia, conducted this project [8].

After that, the paper introduces a method for data analysis using collaborative filtering technology which is effective for data sets with missing elements [9][10]. The paper presents two trials of this method of data analysis.

Finally, the paper describes a general method for performing such analysis and projections using dynamic measurements of software process and a database of past project measurements, based on the described research experiences. We propose to experimentally verify this proposed method in future research.

2 EPM: The in-process project measurement platform

EPM automatically collects software development management data from development tools such as a configuration management system, Concurrent Versioning System (CVS), bug tracking system, GNATS, and mailing list management system, mailman. Drawing especially from the configuration management system, EPM automatically collects source code and operational histories of source code development, the basic information concerning transitions that occur in the software development process. Fig. 1 shows an example of its displays.

EPM translates collected data into a standard XML format and stores them in a relational database for analysis. The EPM analysis functions display information in visual formats. The information includes changes in the source lines of code, timing analysis of check-in and check-out, changes in bug numbers, analysis of inter-company mail volumes, and the Software Reliability Growth Model (SRGM) curve.

By using EPM with such basic development tools as a configuration management tool, bug tracking tool, and mailing list management tool, a software project can receive the benefits of automatic measurements and visually presented analyses of project data without the burden of intrusive manual tracking.

3 Experience with in-process project measurement

3.1 Target project description and measurement

The ASD project started in Spring 2005. Funded by the government, the project focuses on development of kernel software for an experimental information system platform for collecting probe information. This probe information system will collect car location information from various automotive elements called probe cars, such as

taxis, trucks, and busses. From this information, the probe information system will generate various useful public information formats.

The project period is two years, separated into two parts. The project is currently in the integration test phase of the 2nd part.

The project was organized as a development consortium, composed of seven companies, including six major software development companies and an automobile manufacturer. The automobile manufacturer acts as the evaluator and the other six companies develop the platform. One of the six companies acts as Project Manager (PM). The six companies are rivals in the probe information system field, so the project clearly distinguishes between collaboration and competitive materials. Information in the collaborative field is shared and in the competitive field is confidential. For example, detail design, source code, and source line of code (SLOC) productivity are confidential. However, the PM needs SLOC information for meaningful project management. Normally this situation would force the PM into a kind of blind management. During the companies' individual development phase, management would be based on declarations. Only in the inter-company integration test phase would all members share the real situation of the developed software.

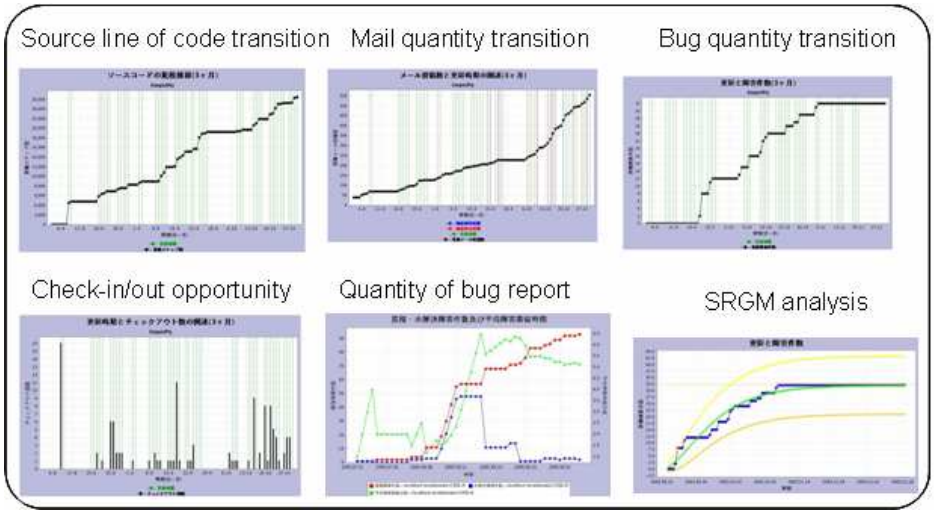

Fig.1 Empirical Project Monitor (EPM) display example

The target software is written in C/C++ and runs on several Linux servers with a relational database for data processing and personal computers for data display. Each consortium company measured project data, which was collected by SEC and analyzed for software engineering research. Analyzed data was fed back to the individual companies with respect for confidentiality. The PM was provided with a bird's-eye view of the total information, again with respect for confidentiality. This allowed more than the blind management that had been expected.

For this project, the following five methods of measurement, shown in Fig. 2, were used:

1) EPM measurement and analysis

EPM collected development process and product information, and produced analysis results.

2) Collection and analysis of review reports

An electronic data form with 30 items was used to collect information concerning basic and detailed design reviews.

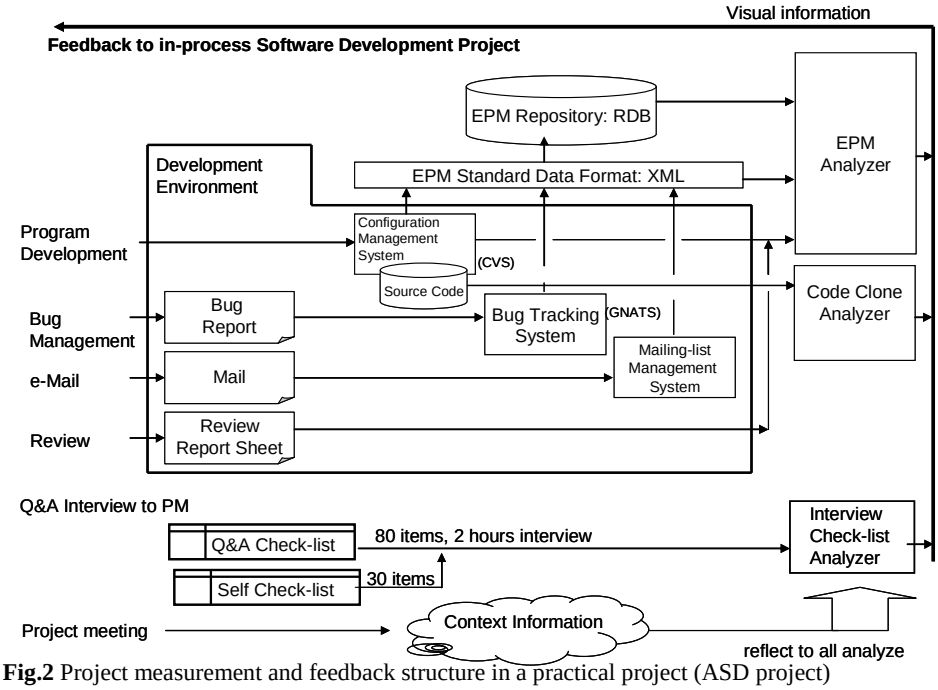

Fig.2 Project measurement and feedback structure in a practical project (ASD project)

3) Code Clone Analysis

A code clone is a code fragment in source code which is identical or similar to each other. Code clone fingerprints such as code clone distribution or content ratio represent software product characteristics. In this trial, we used CCFinder[11], which is a code clone detection tool. CCFinder displays the result of code clone detection by using Scatter Plot.

4) Questionnaire and interview of project leader and PM to collect project context information

SEC developed a questionnaire for collecting context information which includes 30 self-assessed items and 80 items for interviewing. These lists were developed in the context of the PMBOK knowledge areas. The self-assessment and interview collect various context data items which are difficult to obtain through automatic tools such as EPM.

5) Collect project context information by participation in project meetings

To collect more data about the project context, research staff attended all the project meetings. This was very useful in collecting information that could not be collected in other ways.

3.2 Results of the project measurement

In this project, the measurement effort brought development out of the black box into the daylight and helped to form a consensus about how to handle project management. For example, the following characteristics of this project were identified:

- Measurement provided a bird's-eye view of each company's project based on transitions in source line of code count and transitions in bug numbers.
- Code clone analysis helped identify the origin of source code and the development approach taken by different groups. For example, this analysis helped identify whether they mostly used code developed from scratch, by cut and try methods, or appropriated and reused code. The analysis also suggested characteristics about the source code such as whether it was produced by a less experienced coder or not, possible issues with future code maintenance, and the status of the refactoring process.

The project structure did not make source code from individual companies available to the project manager, so it was very useful to share and discuss code clone data between the development companies and the project manager. The fact that the project manager referred to code clone analysis data caused positive effects on the project. For example, one company explained their system design concept when a code clone was detected. This clone was the result of design considerations intended to increase future extendibility. This kind of discussion helped raise morale and provide opportunities for inter company coordination.

-Analysis of file renewal suggested differences in the development process, such as use of waterfall type process or cut-and-try development. This analysis clearly showed the stability of file renewal, the impact of design changes, and attention to bug detection in the late developed process.

- Analysis of bug reports showed clear relationships between various bug factors. In particular, analysis of relationships between the bug injection process and the bug detection process, along with consideration of when bugs should ideally be detected, were particularly useful in evaluating the early development process.
- The analysis of review reports clearly indicated the different attitudes towards the review process. Some companies invested significant effort in the review process, reducing problems in later stages of their waterfall development process. Other companies slighted the review process, expecting problems to be caught by later testing instead. We could expect some conflict on software parts from these different companies will be combined at the system integration test phase.
- The leader questionnaire and interview, recorded using checklists, provided information about the development structure as part of the context information which was generally confidential inside each company in the consortium. By providing some part of this information to the project manager, the project manager could understand how the different companies worked, allowing better project management.

For example, development in some companies with low code clone content was largely development from scratch, with some cut-and-try development in the logical processing portion of the project. Another company with high code clone content had a large level of reuse, and considered a key factor in success of this development to be adaptation of reused code. Finally, one company apparently expected the integration test to uncover problems instead of finding them by review activities.

4 Post-process benchmark data collection

4.1 Benchmark data collection from over 1000 projects and building a national database

The SEC has started to collect software project benchmark data from industry to build a national level database. As the first step in this process, the SEC has collected data from 1009 projects in 15 software industries. Preliminary analysis results have been published as the Software Data White Paper [12].

The data items collected were defined by the SEC in reference to data items previously collected by Japanese software industries and also data items collected by the International Software Benchmark Standard Group (ISBSG). The list contains about 490 items in 10 categories. An example is shown in Table 1. The data items are collected through an electronic data form. Preliminary analysis has identified some useful database attributes such as program size, total effort, productivity, reliability, and some correlations between basic data items.

Table 1 Example of Benchmark Data Items

Classification	Items
General items charactering projects	Type of development, new development, new customers, level of success
Application domain	Domains, types of application, characteristics of users
System characteristics	Usage of ERP, development platforms, development languages
Development procedure States of projects	Life cycle models, usage of tools, rate of reuse States of development teams, work environments
Requirements management	Ambiguity, commitment by users
Personal skills	Skills of Project Manager (PM), skills of team members
Development size	Function Point (FP) methods, FP, Source Line Of Code (SLOC)
Term	Terms of development (actual, planned)
Development effort	Total efforts (actual, planned), efforts by phases (actual, planed)
Quality	Total defects, defects by phases

4.2 Collaborative filtering of the benchmark database

The collected database includes many data sets with missing elements, and various kinds of projects. To analyze them required a technology that can handle missing elements and perform grouping and categorization of the projects. Collaborative filtering technology was applied to group similar projects from data sets with missing elements. A key feature of collaborative filtering for this application is that it can analyze data sets directly without any special operation for missing elements in included data sets or any special variable selection.

To validate the technology, a kind of experimental project prediction was performed using the project benchmark database. First, benchmark data from one project was selected as a key, and one data item from it, such as total development effort, was hidden. Next, the collaborative filtering tool retrieved a group of projects with similar data sets. Then the project total effort was estimated from the retrieved set of similar data and compared with the hidden real value. This makes a prediction or estimate based on the similar projects found in the SEC database. Ohsugi et al's research [13] (in Japanese) reports on this as a case study, but it suggests the potential of this prediction method. In this experiment, "total effort" was selected as an estimation target item from the 490 data items collected. The filtering key data consisted of 97 data items of planning data and actual data from the beginning to the end of the detailed design phase. Data from 378 projects which had no missing "total effort" measure were selected from the data on 1009 projects as target data for filtering. In the data for selected projects, the ratio of missing data items to total data items was 67%. Collaborative filtering allowed estimation of the total effort based on several other indicators. The average relative error, comparing the predicted value to the measured value, was 0.64. This result suggests the usefulness of this method despite the significant missing elements in the database. This research was only a case study, but indicates the importance of the activity of the SEC in creating a national database for software project benchmarks.

The ASD project has collected planning data and actual results from the beginning to the end of the basic design phase. The project total effort was estimated using the collected partial benchmark data as a key and the SEC benchmark database. After the end of the project, this can then be compared to final actual results data. In other words, the final results of the project will be compared with an estimate predicted using the current project benchmark data and the SEC database of previous software projects.

For example, in this experiment, the collaborative filtering tool calculated a project similarity grade indicating project similarity in 10 steps from 0.0 to 1.0, which was used to illustrate a similarity distribution graph. In the ASD project trial, collaborative filtering using the partial benchmark data from five companies at the end of the basic design phase retrieved about 70 similar projects in the similarity range from 0.9 to 1.0 from the 1009 projects in the SEC database. Both manual review and some statistical processing allow extraction of useful information for project operation from the characteristics of the retrieved group of similar projects.

Since the ASD project is now in the integration test phase, final evaluation results for this method have not yet been completed. However, we expect to verify the usefulness of this method in this project and confirm Ohsugi's research results.

5 A proposed method for using in-process measurements

Accumulated in-process data about process and product form a valuable database after completion of the project. However, it is not easy to use this information during the project with only simple accumulation of data.

Generally, in-process measurements are plotted with time as the horizontal axis and changes in various indices as the vertical axes. Typically the macro trend of the changes has meaning instead of the absolute numerical values. In most cases, the visual patterns of the graph or chart provide useful guidance for project management and operation.

As a main point of this paper, the authors propose in-process measurements and groups of similar projects extracted from the project benchmark database as described in 4.2. Fig. 3 illustrates the outline of this method, and the following describes the procedure. Number in parentheses corresponds to those in Fig. 3.

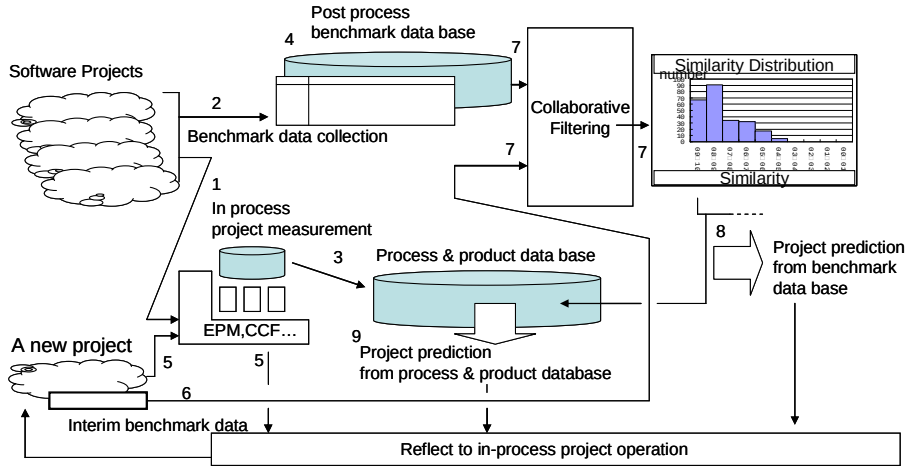

Fig.3 Project prediction by collaborative filtering with two kinds of project database

- First, from every project, benchmark data as described in 4.1 and measurement data about process and product as described in 3.1 are collected in a dataset (1)(2). This data is accumulated in a database (3)(4)
- Second, for a new project, interim benchmark data is collected, and collaborative filtering used to retrieve a group of similar projects from the benchmark database (7)
- Finally, the process and product measurements for the group of similar products (8) are used to generate estimates for the new project (9). These data, predictions based on the benchmark database (8) and in-process data measurements (5) are referenced to project operation in all.

6 Initiatives for future research

In early 2006, the authors initiated a new plan to experimentally validate the above approach. The plan includes the following elements.

1. First year: Development of easy-to-use distribution kit of measurement platform (EPM tools).
2. Second year: Execute practical experiment of measurement and database construction with ten trial projects.

The project measurement, analysis, and feedback mechanism shown in Fig. 3 depends on construction of a database of project measurements. Such databases built inside companies are useful, however, a national database like the SEC benchmark database is considered more valuable.

Popularizing the measurement platform is an important first step in process and product data measurement. The new plan includes distribution of a useful tool kit or environment for measurement and practical experiments applying it. The authors' aim is to make the measurement platform highly popular and to share the evaluation results from the practical experiment widely, then to build a national level mechanism which includes the measurement database shown in Fig. 3.

7 Conclusion

In the field of software project measurement, there are two broad kinds of measurement, post-process collection of benchmark data and in-process measurements of process and product.

In Japan, the SEC has been building a national database for benchmark data since 2005.The authors' experiment has demonstrated one method for using this database. The method provides predictions or estimates for projects by applying collaborative filtering to retrieve groups of similar projects from the benchmark database using interim measurements of the current project as the retrieval key.

In terms of in-process project measurements, useful tools such as the measurement platform called EPM and code clone analysis tool called CCFinder have been provided, and experimentally shown to be useful in projects such as the ASD project. This paper provided a bird's-eye view of the authors' work and proposed an approach to build and use a database of process and product measurements, integrating the previous experiments of the authors. The approach uses collaborative filtering to extract groups of similar projects from a benchmark database using interim benchmark data from a current project. Finally, the authors describe a future initiative to develop the proposed environment and verify its usefulness.

Acknowledgements

This work is supported by IPA/SEC, METI and the MEXT of Japan, the Comprehensive Development of e-Society Foundation Software program. We thank researchers in the SEC and EASE project who kindly support our project.

REFERENCES

[1] EASE project: http://www.empirical.jp/English/index.html

[2] Yoshiki Mitani, Mike Barker, Koji Torii, Seishiro Tsuruho: An Experimental Framework for Japanese Academic-Industry Collaboration in Empirical Software Engineering Research, International Symposium on Empirical Software Engineering ISESE)2004,Vol.2, Redondo Beach, USA, Aug. (2004) pp.35-36

[3] Masao Ohira, Reishi Yokomori, Makoto Sakai, Ken-ichi Matsumoto, Katsuro Inoue, Michael Barker, Koji Torii: Empirical Project Monitor: A System for Managing Software Development Projects in Real Time , International Symposium on Empirical Software Engineering ISESE 2004, Vol.2, Redondo Beach, USA, Aug (2004) pp.37-38

[4] Naoki Ohsugi. EASE Project: Introducing Empirical Software Engineering into Japanese Industry, International Workshop on Future Software Technology (IWFST) 2005, Shanghai, China, Nov.(2005)

[5] Yoshiki Mitani, Nahomi Kikuchi, Tomoko Matsumura, Satoshi Iwamura, Mike Barker, Ken-ichi Matsumoto:An empirical trial of multi-dimensional in-process measurement and feedback on a governmental multi-vendor software project. International Symposium on Empirical Software Engineering (ISESE) 2005, Vol.2, Noosa Heads, Australia, Nov. (2005) pp.5-8

[6] Yoshiki Mitani, Nahomi Kikuchi, Tomoko Matsumura, Satoshi Iwamura, Yoshiki Higo, Katsuro Inoue, Mike Barker, Ken-ichi Matsumoto: Effect of Software Industry Structure on a Research Framework for Empirical Software Engineering: 28th International Conference on Software Engineering(ICSE2006), Far East Experience Track, Poster Session; Shanghai, China, May (2006) pp.616-619

[7] Nahomi Kikuchi: Experience from Analysis of 1000 Collected Project Data, International Workshop on Future Software Technology (IWFST) 2005, Shanghai, China, Nov.(2005)

[8] SEC: http://www.ipa.go.jp/english/sec/index.html

[9] Naoki Ohsugi Akito Monden, Shuuji Morisaki: Collaborative Filtering Approach for Software Function Discovery: International Symposium on Empirical Software Engineering (ISESE) 2002, vol.2, Nara, Japan (2002) pp.45-46

[10] Naoki Ohsugi, Masateru Tsunoda, Akito Monden, Ken-ichi Matsumoto: Effort Estimation Based on Collaborative Filtering: 5th Intl. Conf. on Product Focused Soft. Process Improvement (PROFES) 2004, Nara, Japan (2004) pp.274-286.

[11] Toshihiro Kamiya, Shinji Kusumoto, Katsuro Inoue: CCFinder: A Multi-Linguistic Token-based Code Clone Detection System for Large Scale Source Code. IEEE TSE 28 (2002) pp.654-670

[12] IPA/Software Engineering Center (SEC): Software Development Data White Paper 2005: NIKKEI BP, 2005-5(2005) p137 (in Japanese)

[13] Naoki Ohsugi, Masateru Tsunoda, Akito Monden, Tomoko Matsumura, Ken-ichi Matsumoto, Nahomi Kikuchi: Using Cross-company Data to Estimate the Effort of Software Development Projects; SEC journal No.5 2006-2(2006) (in Japanese)