Evidence-Based Software Portfolio Management: A Tool Description and Evaluation

Hennie Huijgens
Delft University of Technology and
Goverdson, The Netherlands
h.k.m.huijgens@tudelft.nl

ABSTRACT
Context: In this paper we describe and evaluate a tool for Evidence-Based Software Portfolio Management (EBSPM) that we developed over time in close cooperation with software practitioners from The Netherlands and Belgium. Objectives: The goal of the EBSPM-tool is to measure, analyze, and benchmark the performance of interconnected sets of software projects in terms of size, cost, duration, and number of defects, in order to support innovation of a company’s software delivery capability. The tool supports building and maintaining a research repository of finalized software projects from different companies, business domains, and delivery approaches. Method: The tool consists of two parts. First, a Research Repository, at this moment holding data of for now 490 finalized software projects, from four different companies. Second, a Performance Dashboard, built from a so-called Cost Duration Matrix. Results: We evaluated the tool by describing its use in two practical applications in case studies in industry. Conclusions: We show that the EBSPM-tool can be used successfully in an industrial context, especially regarding its benchmarking and visualization purposes.

CCS Concepts
• General and reference → Cross-computing tools and techniques • Evaluation.

Keywords
Evidence-Based Software Portfolio Management, EBSPM, Benchmarking, EBSPM-tool, Software Portfolio Management.

1. INTRODUCTION
Benchmarking is an important part of learning how and where to innovate for software companies. In this paper we describe a software project benchmark tool that compares the performance of software projects in terms of cost, duration, number of defects, and size with a measurement repository of finalized software projects from different companies.

For a period of seven years and ongoing we collected performance data of finalized software projects in industry, in close cooperation with a number of large banking and telecom companies in The Netherlands and Belgium. Based on this we built a research repository of core metrics data of, by now, approximately 500 software projects.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

EASE ’16, June 01-03, 2016, Limerick, Ireland
© 2016 ACM. ISBN 978-1-4503-3691-8/16/06...$15.00
DOI: http://dx.doi.org/10.1145/2915970.2916012

1.1 We noticed four shortcomings
During the collecting and analyzing of project data in industrial practice we experienced four major shortcomings.

First, many software companies and software researchers look upon success and failure of software projects in a way that is strongly related to realizing the estimated plan, as for example described in Standish CHAOS research [2]. In recent research we argue that this focus might be wrong because finalizing a project according to its plan does not imply that the achieved performance is good too (maybe the plan was just bad?) [3].

Second, although many software benchmark repositories are available – Jones [4] identifies in 2011 a number of no-less than twenty-five sources of software benchmarks; Menzies and Zimmermann [5] mention thirteen repositories of software engineering data – we experience in practice that many practitioners and researchers struggle with the question on ‘how to convince decision makers based on facts and evidence from benchmark repositories’. Although some more or less open solutions are found (e.g. ISBSG, Promise), the majority of the benchmark sources are data collections that are available from commercial companies. Most benchmark sources give no insight in the raw data; especially commercial benchmarks tend to offer trends and aggregated data only.

Third, almost none of the benchmark sources include cost data of software projects as a source for productivity. All benchmarks use effort as a core metric for productivity indicators. In itself, the choice of effort as a core metric for productivity is correct. In the practice of collecting data for our research repository, however, we noticed that it is challenging to collect reliable effort data of software projects.

Fourth, in practice this vast variety of benchmark sources makes it difficult for practitioners to decide on how to implement a mature measurement and analysis capability that suits the needs of a company’s decision makers. As an effect of on the one hand great variety of benchmark approaches and on the other low degree of standardization, we observe that a change of a measurement and analysis approach often can be linked with changes in decision makers and with changes in a primary development approach. As an example: a misunderstanding that apparently lives in many companies is that “going agile means opting for a new measurement approach too”.

1.2 A conceptually new solution
In order to deal with these shortcomings, and driven by the somewhat lagging evidence-based software engineering capability in both research and industry [6], and our deeply felt importance of software companies to get more attention for evidence-based software engineering from a software portfolio point of view, we developed a conceptually new instrument, the Cost Duration Matrix. The primary goal of this matrix is to identify good practice software
We named our approach Evidence-Based Software Portfolio Management (EBSPM). Hence, we refer to our tool as the EBSPM-tool. In particular, the main contributions of the tool are twofold. First, it positions a Cost Duration Matrix as core instrument within a Performance Dashboard for analysis of good practice and bad practice in company-wide portfolios of software projects. Second, it provides a research repository, holding data of industrial software projects from different companies, on a standardized set of metrics: size, cost, duration, and defects.

The remainder of this paper is organized in the following way: In Section 2 we outline relevant prior work. In Section 3 we describe the EBSPM-tool and its functional components. In order to evaluate the EBSPM-tool we describe in Section 4 how the tool was used in two scenarios in industrial practice. In Section 5 we discuss the applicability of the tool for research purposes, in a practical setting.
in industry, and in education. Finally, Section 6 includes a summary of the current status of the EBSPM-tool.

2. RELATED WORK

Although many sources for benchmarking of software engineering are available [4] [5], the need for good economic models will grow rather than diminish as software becomes increasingly ubiquitous [7]. There is a lack of successful and agreed upon standards metrics set and selection processes [8]. We assume that especially follow-up on studies with regard to software estimation techniques and algorithms [9] [10] [11] [12] [13] [14] [15], in combination with Evidence-Based Software Engineering [16], and in depth investigation in estimation and use of Functional Size Measurement as a source for software analysis [17] [18] [19] [20] will help to find answers on the challenges in the field. With regard to future developments in software engineering several studies recommend that more theories should be build and tested, preferably with tighter links between academia and industry [21] [22].

Varieties of value-based software engineering are examined [23] [24] [25]. However, a clear link with existing approaches that focus on measurements such as project size, project cost, project duration, and number of defects is not to be found.

3. THE EBSPM-TOOL

The EBSPM-tool offers two basic (initial) features, a Research Repository, and a Performance Dashboard, including a Cost Duration Matrix. These initial features are described in the following paragraphs.

3.1 Research Repository

An important feature of the EBSPM-tool is the availability of a research repository, including data of approximately 500 finalized software projects from different software companies in The Netherlands and Belgium, which is the source for the benchmark functionality. Table 1 gives an overview of the research repository, including the different aspects that are analyzed in our research. The data in the research repository is stored in a MS Excel file.

All software deliveries within the repository are measured over a period of nine years in four different companies (banking, telecom, supplier of a business-to-business software solution). Deliveries are measured by experienced, often certified, specialists. Delivery data was based on formal project administrations and reviewed by stakeholders (e.g. project managers, product owners, finance departments, project support). All deliveries were reviewed thoroughly by the lead author before they were included in the research repository. An important difference of the data in the EBSPM research repository with other software engineering repositories is that we collected were possible a company’s software project portfolio as a whole. Where other repositories focus at projects, we focus at portfolios instead. This enables us to analyze good practice versus bad practice. A second major difference is that instead focus on collecting effort data of software projects, we focus at collecting cost data besides effort data too.

3.2 Performance Dashboard

In order to visualize the outcomes of the analysis we build a Performance Dashboard in the Business Intelligence (BI) solution.
Tableau. The core component within the Performance Dashboard (see Figure 1) is a Cost Duration Matrix. Each software project is depicted as a circle; the larger the circle, the larger the project is (in function points), and the 'redder' the project is, the more defects per function point are found. The position of each project in the matrix represents the cost and duration deviation of the delivery relative to the benchmark, expressed as percentages. The horizontal and vertical 0%-lines represent zero deviation, i.e. projects that are exactly consistent with the benchmark. A delivery at (0%, 0%) would be one that behaves exactly in accordance with the benchmark; a delivery at (-100%, -100%) would cost nothing and be ready immediately; and a delivery at (+100%, +100%) would be twice as expensive and takes twice as long as expected from the benchmark. Based on this percentage all deliveries from the repository are plotted in a Cost Duration Matrix, resulting in four quadrants:

1. **Good Practice** (upper right): projects that score better than the average of the total repository for both cost and duration.
2. **Cost over Time** (bottom right): projects that score better than the average of the total repository for cost, yet worse than average for duration.
3. **Bad Practice** (bottom left): projects that score worse than the average of the total repository for both cost and duration.
4. **Time over Cost** (upper left): projects that score better than the average of the total repository for duration, however worse than average for delivery cost.

The overall performance of the portfolio is furthermore summarized through the two red 'median' lines. For each project in the research repository three Key Performance Indicators are calculated as a measure for the performance with regard to Cost per FP, Duration per FP, and Defects per FP.

### 4. EVALUATION

In order to evaluate the usefulness of the EBSPM-tool, we describe its application in two recent case studies [3] [26] that we performed in industry. Within these studies we used the EBSPM-tool for analysis, benchmarking and visualization purposes. In both case studies we used the tool to analyze and visualize a specific subset of software projects against the tools repository.

In the first study [3] we analyzed a sample of 22 software projects that were performed within a Belgian telecom company. All sampled projects were in scope of an electronic survey among stakeholders from IT and business departments that were involved in a project. In the survey we assessed stakeholder satisfaction and perceived value. In the subsequent analysis we compared the outcomes of the survey with quantitative project metrics such as project size, project cost, project duration, number of defects, and Estimation Quality Factor (EQF) for both duration and cost. We calculated a Cost Duration Index, based on the relative position of a project in the Cost Duration Matrix of the EBSPM-tool [3].

In the second study [26] we performed causal analysis on a sample of nine software releases and eight single once-only projects, that all performed on the same CRM-system in a Belgian telecom company. In order to analyze the performances of each project we interviewed eleven stakeholders from both IT and business on the backgrounds of the projects. The case study resulted in a number of observations from both quantitative and qualitative analysis [26].

In both studies we used the EBSPM-tool to visualize the applicable research sample against the EBSPM research repository as a whole. We used the Performance Dashboard in two ways. First, we included print screens of the dashboard with a sample of projects in the applicable research papers [3] [26]. Secondly we made intensive use of the Performance Dashboard as an interactive visualization during presentations about our case studies to stakeholders within the Belgian telecom company. The fact that the dashboard enables fast interactions and fast selections in company-in-scope, business domain, project name, and development method helped us to gain management commitment and to advice on actionable follow-up actions.

### 5. DISCUSSION

#### 5.1 Evaluation of Validity

We observe large differences between both repositories. The difference in average size of projects in both repositories might partly explain these. However, within the somewhat limited scope of this tool description we cannot argue on any causes that might explain these dissimilarities. Further research is needed to find out any backgrounds of these remarkable differences.

#### 5.2 Impact / Implications

What can we do with the EBSPM-tool and the results from this short study? We try to answer this question looking from three angles; research, practice, and education.

**Research.** We offer our research repository available for other researchers for research and education purposes. We are currently discussing how we can make the repository publicly available via Promise with regard to compliancy issues with cooperating companies. Based on the example given in Figure 1, we argue that further research is needed on the differences found between both ISBSG data and our research repository.

**Practice.** Although already a vast amount of benchmarking sources is to be found, we observe that collecting historic data of software deliveries in itself is a task that learns data analysts and decision makers in software companies a lot about their software delivery capability. The outcomes of the evaluation of the EBSPM-tool in an industrial context learns that companies should be careful in adopting one single source of benchmarking, especially when collected outside their own environment, as the truth.

As a spin-off from our research on EBSPM, we are currently cooperating with a number of software companies in order to further improve our analysis approach and characteristics of metrics to be collected (e.g. we focus on value and satisfaction too), and to enlarge and further evaluate the content of our research repository.

**Education.** Software engineering economics, especially from an evidence-based point of view, is usually only as part of other disciplines included in the vocabulary of universities. We think that the EBSPM-tool can help to clarify the importance of economic aspects of software development to students and teachers as it visualizes the somewhat diffuse and large content of a companies’ software portfolio repository in a conveniently arranged way.

### 6. CONCLUSIONS

In order to evaluate the EBSPM-tool we looked at the applicability of both the research repository and the Performance Dashboard in an industrial context. We found that the tool supports analysis of software project and portfolio performances. Besides that, the tool enables both internal and external benchmarking. A major difference with other tools is that the EBSPM-tool values project cost data above project effort.
ACKNOWLEDGMENTS
We thank Tableau for allowing us to use their BI solution to build our Performance Dashboard. Furthermore, we thank Leandro Minku for his review support.

REFERENCES
[1] H. Huijgens, R. v. Solingen and A. v. Deursen, "How To Build a Good Practice Software Project Portfolio?,” ICSE Companion 2014 Companion Proceedings of the 36th International Conference on Software Engineering (SEIP), vol. 2014, no. IEEE, pp. 64-73, 2014.
[2] International Standish Group, "The Chaos Report," 1994.
[3] H. Huijgens, A. v. Deursen and R. v. Solingen, "An Exploratory Study on the Effects of Perceived Value and Stakeholder Satisfaction on Software Projects," in IEEE/ACM Proceedings of the International Conference on Evaluation and Assessment in Software Engineering (EASE), 2016.
[4] C. Jones, “Sources of Software Benchmarks,” Capers Jones & Associates, 2011.
[5] T. Menzies and T. Zimmermann, "Software Analytics: So What?,” IEEE Software, vol. July/August, pp. 31-37, 2013.
[6] T. Dyba, B. A. Kitchenham and M. Jorgensen, "Evidence-based software engineering for practitioners,” Software, IEEE, vol. 22, no. 1, pp. 58-65, 2005.
[7] M. Shepperd, "Software project economics: a roadmap,” in Future of Software Engineering (FOSE), Minneapolis, USA, 2007.
[8] T. Tahira, G. Rasoola and C. Gencel, "A systematic literature review on software measurement programs," Information and Software Technology, vol. 73, pp. 101-121, 2016.
[9] M. Jorgensen and M. Shepperd, "A Systematic Review of Software Development Cost Estimation Studies," IEEE Transactions on Software Engineering, vol. 33, no. 1, pp. 33-53, 2007.
[10] B. Boehm, C. Abts and S. Chulani, "Software development cost estimation approaches - a Survey,” Annals of Software Engineering, vol. 10, no. J.C. Baltzer AG, Science Publishers, pp. 177-205, 2000.
[11] T. Menzies, Z. Chen , J. Hihn and K. Lum, "Selecting best practices for effort estimation,” IEEE Transactions on Software Engineering, vol. 32, no. 11, pp. 883-895, 2006.
[12] F. Sarro, A. Petrozziello and M. Harman, "Multi-Objective Effort Estimation," in Proceedings of the International Conference on Software Engineering (ICSE), 2016.
[13] T. Menzies and M. Shepperd, "Special issue on repeatable results in software engineering prediction," Empirical Software Engineering, vol. 17, no. 1, pp. 1-17, 2012.
[14] M. Shepperd and S. MacDonell, "Evaluating prediction systems in software project estimation,” Information and Software Technology, vol. 54, no. 8, pp. 820-827, 2012.
[15] L. Minku, E. Mendes and F. Ferrucci, "How to make best Use of Cross-Company Data for Web Effort Estimation?,” in ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), 2015.
[16] B. A. Kitchenham, T. Dyba and M. Jorgensen, "Evidence-based software engineering,” in Proceedings of the 26th international conference on software engineering, IEEE Computer Society, 2004.
[17] A. Abran, I. Silva and L. Primera, "Field studies using functional size measurement in building estimation models for software maintenance," Journal of Software Maintenance and Evolution: Research and Practice, vol. 14, no. John Wiley & Sons, Ltd., pp. 31-64, 2002.
[18] A. F. Minkiewicz, "The Evolution of Software Size: A Search for Value,” Software Engineering Technology, vol. March/April, pp. 23-26, 2009.
[19] C. Gencel and O. Demirors, "Functional Size Measurement Revisited," ACM Transactions on Software Engineering and Methodology, vol. 17, no. 3, pp. 15:1-15:36, June 2008.
[20] S. S. Bajwa, C. Gencel and P. Abrahamsson, "Software Product Size Measurement Methods: A Systematic Mapping Study," in IEEE Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA), 2014.
[21] D. Sjoberg, T. Dyba and M. Jorgensen, "The future of empirical methods in software engineering research," in IEEE Future of Software Engineering (FOSE), 2007.
[22] D. E. Perry, A. A. Porter and L. G. Votta, "Empirical studies of software engineering: a roadmap," in ACM Proceedings of the conference on The future of Software engineering, 2000.
[23] B. Boehm, "Value-Based Software Engineering," ACM SIGSOFT Software Engineering Notes, vol. 28, no. 2, pp. 1-12, 2003.
[24] S. Biffl, A. Aurum, B. Boehm, H. Erdogmus and P. Grünbacher, Value-Based Software Engineering, Berlin Heidelberg: Springer, 2006.
[25] S. Faulk, D. Harmon and D. Raffo, "Value-Based Software Engineering (VBSE): A Value-Driven Approach to Product-Line Engineering," in First International Conference on Software Product-Line Engineering, 2000.
[26] H. Huijgens, A. v. Deursen and R. v. Solingen, "Success Factors in Managing Legacy System Evolution: A Case Study," in IEEE/ACM Proceedings of the International Conference on Software and System Processes (ICSSP), Austin, TX, USA, 2016.
[27] ISBSG, "International Software Benchmarking Standards Group," 2014. [Online]. Available: http://www.isbsg.org/.