Delft University of Technology
Software Engineering Research Group
Technical Report Series

Do Estimators Learn? On the Effect of a Positively Skewed Distribution of Effort Data on Software Portfolio Productivity

Hennie Huijgens, Frank Vogelezang

Report TUD-SERG-2016-004

TUD-SERG-2016-004

Published, produced and distributed by:

Software Engineering Research Group
Department of Software Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4
2628 CD Delft
The Netherlands

ISSN 1872-5392

Software Engineering Research Group Technical Reports:
`http://www.se.ewi.tudelft.nl/techreports/`

For more information about the Software Engineering Research Group:
`http://www.se.ewi.tudelft.nl/`

Note: Accepted for publication in *Proceedings of the 7th International Workshop on Emerging Trends in Software Metrics*

Do Estimators Learn? On the Effect of a Positively Skewed Distribution of Effort Data on Software Portfolio Productivity

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

Frank Vogelezang
Ordina
Nieuwegein, The Netherlands
frank.vogelezang@ordina.nl

ABSTRACT

We study whether an assumed positively skewed distribution of effort data prevents software estimators to learn over time; leading to increasing differences between planned and actual effort and a deteriorating (worsening) trend on productivity. We analyze data of 25 software releases of one application, collected over a period of six years in a public sector institution in The Netherlands. We statistically test for distribution, trend on differences between planned versus actual effort over time, and productivity of software portfolios. The key contributions of this paper are that we show that a proposed assumption that assumes any relation between a positively skewed distribution of effort data and a deteriorating productivity is not applicable to the subject dataset. We find that the effort data is to be characterized as positively skewed distributed, and we do see a shift over time from under-estimation to over-estimation. We do not find evidence for a deteriorating productivity; on the contrary productivity improves over time, indicating that estimators in the subject organization did learn.

CCS Concepts

• General and reference → Cross-computing tools and techniques → Estimation.

Keywords

Software Economics, Software Estimation, Function Point Analysis.

1. INTRODUCTION

The goal of software economics is to use insights in relationships between economic aspects and technical software issues to improve software productivity, with a final result of a significant, quantified improvement in value created by investments in software projects and portfolios at different organizational levels: project, program, portfolio, and enterprise [1].

A consequence of this idea is that an organization should be able to not only look at its software projects performance at a project level, but also on higher levels, such as portfolio and organization. And this is also applicable to the concept of software estimation, versus actual software project performance.

Starting from the premise of software effort data, both planned and actual, an effect attracts our attention when working with effort data in practice: we notice that most sets of effort data show a positively skewed distribution; one whose elongated tail extends to the right end of the range. Based on this we develop an assumption that we expect estimators not to learn over time, because the long tail worse than average effort in historic software project data makes them go for deteriorating (worsening) estimations. We assume this effect to be especially applicable to situations where more estimators negotiate about new project estimations, more precisely in environments where Delphi-PERT approaches are common; an estimation method where a panel of experts analyzes the involved tasks in completing a given project or release, especially the time needed to complete each task, and the minimum time needed to complete the project as a whole.

In this paper we test this assumption by studying whether the described effect is applicable to data of 25 software releases that are collected over a period of six years in a public sector institution in The Netherlands. In order to understand the backgrounds of the described effect we investigate whether any relation can be found between a positively skewed distribution in effort data of that organization and a deteriorating trend in planned versus actual effort.

1.1 Problem Statement

Software is everywhere. Software project portfolios in industry are often big and complex. The ability of organizations and their products, systems, and services to compete, adapt, and survive will depend increasingly on software [2]. On one hand significant advances are made from the 1980's to now in the usage of function points and development of new estimation models [3]. On the other, in many software companies, software project estimation still leans heavy on estimation methods and parametric models and algorithms that are developed in the 1970s, while questions arise on the validity and accuracy of some of these approaches [4]. There is only limited insight on the long-term effects of software estimation approaches on a company's success or failure in software engineering, especially when success and failure are expressed in terms of improvement or deterioration of a software project portfolio as a whole.

Being an important problem, at the same time it is hard to find a realistic solution. In practice there is a limited availability of planned and actual effort data of one company's software project portfolio that's collected over a longer period of time. Measurement repositories usually do not contain such data; effort data on planned and actuals over a longer period (multiple years) is not to be found. The absence of such repositories seems to indicate that it is difficult to set up case studies on this subject together with industry; measuring project data over a longer period asks for long-term commitment of both industry partners and researchers. We observe

WETSoM'16, May 14, 2016, Austin, Texas, USA.
Copyright 2016 ACM 978-1-4503-4177-6/16/05 …$15.00.
DOI: http://dx.doi.org/10.1145/12345.67890

that in software companies where we performed research, it is often difficult, if not impossible to collect reliable effort data [5].

In earlier research [5] [6] we noticed that cost and effort in software project portfolios usually are characterized by a positively skewed distribution. This phenomenon of positively skewed distribution for software project data is well known. A good example is in the distribution analysis and graphical representations of the data from the ISBSG repository of software projects [7].

1.2 Research Objectives

In accordance with [8] we argue that a preferred solution for the above described problem statement is to build and test an assumption on the assumed effect. To do so we define our research objective: what is the relationship between a positively skewed distribution of cost data in the subject software organization and over- and under-estimation of software project effort and productivity in the organization's series of software projects over time? Based on this objective we define three research questions:

RQ1: Is a positively skewed distribution applicable to software effort data in the subject organization?

RQ2: Are differences between planned effort and actual effort becoming larger, when observed over time?

RQ3: Is overall productivity deteriorating when observed over time?

1.3 Context

In order to give an answer to the above questions, we study a subset of data from finalized software projects implemented within a public sector institution in The Netherlands for six consecutive years. It concerns data from 25 software releases that were performed on one specific application within this organization.

In late 2008 a public sector institution in The Netherlands commissioned the adaptive maintenance of a new information system that was built in Oracle Application Express (Apex). The purpose of this contract was to consolidate a number of small applications with dedicated functionality, ranging from spreadsheets to specialized third-party software [9]. The initial release of this information system had been built by the same contractor that was awarded the contract for the adaptive maintenance of the system.

Contractor fees for the adaptive maintenance in this contract are based on the functional size of enhancement releases. Functional size of each enhancement release is measured according to the Nesma functional size measurement method ISO/IEC 24570 [10].

The enhancement contract is carried out by a steady team that specializes in software enhancement of applications that are built with Apex. The team does this for multiple customers. Release estimates of the required effort and lead time are made by at least three team members, using a Delphi-PERT estimation approach. Function points were counted due to contractual agreements, however, they were not used for estimation purposes.

The functional size of each enhancement release was determined by the lead architect of the enhancement team and reviewed and approved by the quality management of the commissioning institution before contractor fees were made final. In 2013 the maximum contract period expired and the adaptive maintenance was commissioned again. The contract was awarded to a different contractor.

The study shows that our proposed assumption that assumes any relation between a positively skewed distribution of effort data and a deteriorating productivity is not applicable to the set of release data collected over a period of six years in a public government institution. The effort data is to be characterized as positively skewed distributed, and show a shift over time from under-estimation to over-estimation. We do not find evidence for a deteriorating productivity; on the contrary productivity improves over time, indicating that estimators did learn.

The remainder of this paper is organized in the following way. In Section 2, we survey earlier research on the effects of software estimation on a longer term. In Section 3, we outline the research approach. In Section 4, we present results. In Section 5 the results are discussed and compared with the state of the art and we evaluate validity. Finally, Section 6 includes conclusions and future work.

2. RELATED WORK

Actual research with regard to software estimation focusses on aspects such as quality of estimations, and reliability of estimations [11] [12], estimates uncertainty [13] [14], Structured Literature Study on existing research on software estimation in combination with Evidence-Based Software Engineering [3] [15], depth investigation in estimation techniques and algorithms [16] [3] [11] [17] [18] [19] [20], and use of Functional Size measurement as a source for software estimation [21] [22] [23] [24].

There are a number of authors who have explored the economic concepts and theories, including studies on economics or diseconomies of scale in software projects [7] [25].

With regard to future developments in software engineering, more especially in software estimation, several studies recommend that more theories should be build and tested, preferably with tighter links between academia and industry [8] [26].

Search-based approaches for effort estimation are getting more and more attention from the software engineering community, a comprehensive overview is given in [27].

3. RESEARCH APPROACH

We formulate our assumption that might explain an assumed phenomenon of deteriorating productivity occurring due to a positively skewed effort distribution. We derive testable hypotheses with regard to our research questions. In order to test the hypotheses, we perform a retrospective case study on the available data of the public sector institution. In the case study we limit the research to a quantitative study in order to statistically analyze effects in the subject data. Due to the fact that we do have only limited access to employees that performed estimations in the past we choose not to include qualitative research by performing structured interviews with experts.

As described in the preceding we build our study on an assumption that might explain an assumed phenomenon of deteriorating productivity occurring due to a positively skewed effort distribution. The central idea behind the assumption is that we notice in practice that effort data usually is characterized by a positively skewed distribution. As estimators get more experienced with the project and the domain, there are fewer unknowns. Therefore, we expect the difference between planned and actual estimations to decrease over time, indicating that estimators do learn from historic projects.

However, due to a positively skewed distribution we assume that when negotiating about software project estimations – as we assume estimators are doing often – a risk occurs that planned estimations end up in the area "on the wrong side" of a series of software project's average [6]; meaning the estimation plans for a deteriorating productivity. In order to test this assumption, we study three aspects of software projects: the distribution of effort data, the measure of over- or under-estimation, and the productivity over time in the subject public sector institution.

3.1 Research Questions

In this paragraph we formulate our assumption that might explain an assumed phenomenon of deteriorating productivity occurring due to a positively skewed effort distribution. We derive testable hypotheses with regard to our research questions. As a script for a quantitative research question, we define the following:

Does the proposed assumption explain the relationship between a positively skewed distribution of software effort data and deteriorating software project productivity? In other words, can we demonstrate any relationship between a positively skewed distribution of effort data and deteriorating software project productivity in the subject organization?

Based on this, our null hypotheses for each of the three research questions are:

- RQ1-H_0: The subject data of software project effort is not characterized by a positively skewed distribution.
- RQ2-H_0: The difference between planned effort and actual effort in the subject project data does not get larger when measured over a period of six years.
- RQ3-H_0: The productivity of the subject software projects does not deteriorate when measured over a period of six years.

3.2 Case and Subject Selection

From January 2008 to September 2013 data has been captured on 25 releases of this information system on Build and Test activities (see Table 1). Only data on the enhancements to the Apex system are reported. Effort spent on design of functionality and decommissioning of obsolete systems are out of scope. These activities were performed by commissioning institute teams.

We define growth as the increase of product size as a result of the release. The ratio between size and growth, relative growth, is an indicator of the percentage newly added functionality in a release. For example, releases 3.1, 4.0, 5.0 and 5.1 have a relative growth of 100% and consisted entirely of newly added functionality.

3.3 Data Collection Procedure

Release estimates of the required effort and lead time are made by at least three team members, using a Delphi-PERT estimation approach. This approach results in a 3-point estimate with a lower-bound, most-likely estimate and a higher bound. The (lowest + 4 x likely + highest)/6 estimate was used to estimate the required effort and lead time.

Person-hours for the main build contain all productive hours of the Apex maintenance team to build the software after the functional design of the release has been approved by the institution. These hours contain technical design, database changes, development, unit testing, code documentation, and updates to the user manuals.

The person-hours for testing contain all productive hours of the Apex maintenance team to test the software after it has been built to hand it over to the institution for acceptance testing. These hours contain system testing, functional testing, and updates to the test documentation. The person-hours do not include governance activities, idle time and training.

Table 1. Overview of the subject data set.

	Year	Actual Effort (Hrs)	Planned Effort (Hrs)	Added-Modified Size (FP)	Growth (FP)[1]
Release 1.1	2008	808	533	154	100
Release 2.0	2008	749	554	137	84
Release 2.1	2008	496	706	150	102
Release 3.0	2009	301	1,437	383	210
Release 3.1	2009	30	142	25	25
Release 4.0	2009	176	896	171	171
Release 5.0	2009	116	456	82	82
Release 5.1	2009	71	311	50	50
Release 5.2	2009	48	200	97	17
Release 5.3	2009	134	694	146	98
Release 6.0	2010	789	3565	945	270
Release 6.1	2010	123	683	150	102
Release 6.2	2010	179	859	325	14
Release 6.3	2010	109	509	203	21
Release 6.4	2011	269	1,189	276	174
Release 6.5	2011	76	252	76	27
Release 6.6	2011	33	129	24	20
Release 6.7	2011	57	289	102	12
Release 7.0	2011	48	192	62	10
Release 7.1	2011	311	1,191	286	138
Release 8.0	2012	397	1,677	507	257
Release 8.1	2012	155	687	147	99
Release 8.2	2012	73	353	106	10
Release 9.0	2013	164	900	244	143
Release 9.1	2013	69	317	101	24

[1] See paragraph 3.2 on how Growth is calculated.

The functional size of each enhancement release was determined by the lead architect of the enhancement team. The architect had received internal training from a certified function point analyst. The certified analyst verified most of the function point analyses that were made in 2008 and 2009. Occasional verification of larger releases took place in later years. The determined functional size of each release was approved by the quality management of the commissioning institution before contractor fees were made final.

Effort data was recorded in the time registration system of the contractor. Since contract fees were related to individual enhancement releases, time registration was done release-based. Effort data was differentiated to main build and test, based on the involved staff member, either recognized as programmer or tester.

3.4 Analysis Procedure

We compute the skewness on the data subset Actual Effort using the Fisher-Pearson standardized third moment coefficient [28]. We define two hypotheses:

- RQ1-H_0: the data do not follow a positively skewed distribution.
- RQ1-H_A: the data follow a positively skewed distribution.

We reject the null hypothesis that the data does not follow a positively skewed distribution when the skewness of Actual Effort distribution is larger than significance levels usually referred in statistical literature (skewness > 2). As estimators get more experienced with the project and the domain, we assume there are fewer unknowns, yet due to a positively skewed distribution the estimations are deteriorating over time. We test this effect statistically by splitting the Actual Effort data in two groups:

- Data subset A: per project; prediction differences at the project beginning (year 2008 to 2010).
- Data subset B: per project; latest prediction differences (year 2011 to 2013).

Subsequently we do a pair-wise group comparison by performing a Wilcoxon test. In case the groups differ significantly, we hypothesize about the experience being improved:

- RQ2-H_0: both the data subset A and B are identical populations, indicating no change occurs in the difference between planned and actual effort in the subject project data when measured over a period of six years.
- RQ2-H_A: both the data subset A and B are non-identical populations, indicating the difference between planned effort and actual effort in the subject project data changes when measured over a period of six years.

We reject the null hypothesis that both data subsets are identical populations (and therefore accept the alternative hypothesis that the distributions of both datasets significantly differ) when the p-value is lower than the significance level usually referred in statistical literature (p-value < 0.05).

In case the alternative hypothesis is accepted we perform a Cliff's Delta test to examine how pronounced the difference between both data subsets is.

As a final test we analyze whether the achieved productivity of the subject software releases deteriorates over time. To do so we perform a test equal to the preceding one. We test this effect statistically by splitting the Actual Effort data in two groups:

- Data subset C: per project; productivity (in FPs per hour) at the project beginning (year 2008 to 2010).
- Data subset D: per project; productivity (in FPs per hour) at project ending (year 2011 to 2013).

Subsequently we do a pair-wise group comparison by performing a Wilcoxon test. In case the groups differ significantly, we hypothesize about the experience being improved:

- RQ3-H_0: both the data subset C and D are identical populations, indicating productivity does not change when measured over time.
- RQ3-H_A: both the data subset C and D are non-identical populations, indicating productivity changes when measured over time.

We reject the null hypothesis that both data subsets are identical populations (and therefore accept the alternative hypothesis that both data sets significantly differ) when the p-value is enough lower than the significance level usually referred in statistical literature (p-value < 0.05). When the alternative hypothesis is accepted we perform a Cliff's Delta test to examine the difference between both data subsets.

All tests mentioned above are performed in R; the applied R-code including results from the tests are included in a Technical Report [29].

3.5 Validation Procedure

We validate our assumption that a positively skewed distribution in actual effort data correlates with a deteriorating productivity by testing three hypotheses. In case one or more of the three defined null-hypotheses are rejected we argue that our assumption is not valid for the subject public sector institution.

4. RESULTS

We subsequently present the results of the analysis on the project data from the subject public sector institution. In the first section we describe the results with regard to research question RQ1; in the second section we go into detail on the results with regard to research questions RQ2 and RQ3.

4.1 Case and Subject Description

In order to analyze the subject data on distribution pattern, with regard to *RQ1-H_0: The subject data of software project effort is not characterized by a positively skewed distribution*, we apply the function skewness from the e1071 package in R on the data subset Actual Effort to compute the skewness. The skewness of Actual Effort is 2.47 (see Table 2). It indicates that the Actual Effort distribution is skewed towards the right, thus positively skewed. We reject the null hypothesis RQ1-H_0, and accept the alternative hypothesis RQ1-H_A, that the data follow a positively skewed distribution.

Table 2. Overview of distributions for Actual Effort, Planned Effort and Project Size.

	Actual Effort (Hrs)	Planned Effort (Hrs)	Size (FP)
Maximum	3,709	3,565	945
Upper Quartile	1,006	896	244
Median	636	554	147
Lower Quartile	364	311	97
Minimum	126	129	24
Skewness	2.47	2.43	2.37

The skewness of actual effort within the dataset, one whose elongated tail extends to the right end of the range is showed too in the distributions as depicted in Table 2. Actual effort ranges from 126 to 3,709 Hours, while the median is 636 Hours. The project size ranges from 945 to 24 FPs, the median project size is 147 FPs (see Table 2). Table 2 shows that a positively skewed distribution is not only applicable to Actual Effort data, yet this goes for Planned Effort and Project Size (FP) data as well.

4.2 Analysis

In the following paragraph we examine *RQ2-H_0: The difference between planned effort and actual effort in the subject project data does not get larger when measured over a period of six years*. To test whether estimators do not learn over time, and prepare estimations with a larger difference between planned and actual effort over time, we perform a Wilcoxon rank sum test on two data subsets, holding the difference between planned and actual effort in the period from 2008 to 2010 and the difference between planned and actual in the period from 2011 to 2013. The null hypothesis is that the two data subsets are identical populations. As the p-value turns out to be 0.01, and is less than the 0.05 significance level, we reject the null hypothesis, and accept the alternative hypothesis that the two data subsets are non-identical populations.

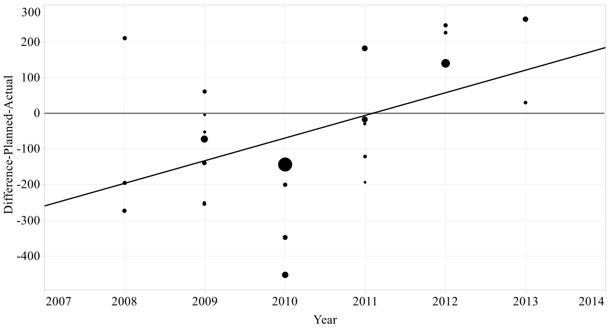

Figure 1. Trend of Difference between Planned and Actual Effort for all projects in scope over time (r^2 = 0.24; Standard Error 174.08.; *p* = 0.01). Size of the dots: bigger dots indicate larger projects in FPs.

In order to examine how pronounced the difference between both data subsets is we perform a Cliff's Delta test in R. The test results in a delta estimate of -0.68, indicating a large difference between both data subsets, where differences get smaller over time.

Figure 1 does confirm an effect that is related to this finding; when the difference between planned effort and actual effort is plotted against year, a positive trend can be seen, indicating that a shift is made over time from under-estimation to over-estimation. Where in the early years of the measurement period on average a tendency can be observed that planned effort is less than actual effort, in the second part of the measurement period a shift is made towards planned effort that on average is higher than actual effort.

In order to examine *RQ3-H_0: The productivity of the subject software projects does not deteriorate when measured over a period of six years*, we test whether the productivity of the software releases deteriorates over time, as expected due to our assumption on the effect of a positively skewed distribution on productivity over time, we perform a Wilcoxon rank sum test on two data subsets, holding the productivity of releases in FPs per hour in the period from 2008 to 2010 and the productivity in the period from 2011 to 2013. The null hypothesis is that the two data subsets are identical populations. The test results in a p-value of 0.01, less than the 0.05 significance level. Therefore, we reject the null hypothesis, and accept the alternative hypothesis that the two data subsets are non-identical populations.

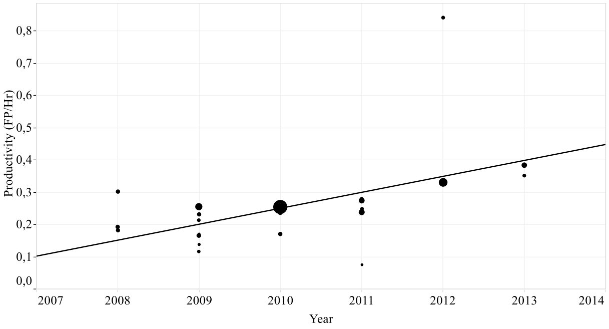

Figure 2. The trend of Productivity (FPs per hour) for all projects in scope (r^2 = 0.27; Standard Error = 0.12; p = 0.01). Size of the dots: bigger dots indicate larger projects in FPs.

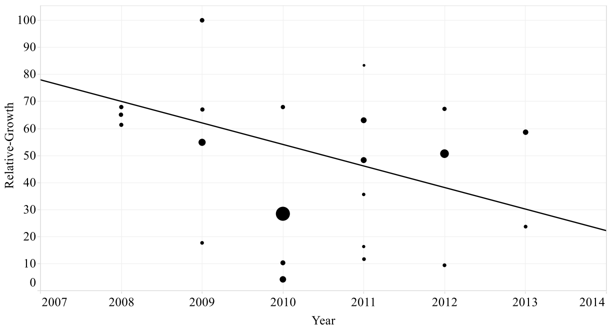

Figure 3. The relative growth of Size (FPs) for all projects in scope over time (r^2 = 0.15; Standard Error = 29.28; p = 0.06). Size of the dots: bigger dots indicate larger projects in FPs.

In order to examine how pronounced the difference between both data subsets is we perform a Cliff's Delta test in R. The test results in a delta estimate of -0.69, indicating a large difference between both data subsets. Figure 3 confirms this finding; when the productivity is plotted against year, a positive trend can be seen, indicating that the productivity improves over time.

Figure 4 illustrates an observation with regard to this improving productivity over time. The subject data set holds besides Size in FPs of every release also the number of FPs that are newly added to the system (see Table 1, column 'Growth'). In Figure 4 the relative growth of the system (percentage New / Size) is plotted against Year; showing a downwards trend indicating that over time a smaller part of the releases was about new functionality, and that a shift occurred towards enhancement of existing functionality.

5. DISCUSSION

Four observations are subject to a closer look. First, a possible explanation for the shift from under-estimation to over-estimation over time can be the fact that in earlier releases many new functionality is delivered, laying close to the core of the system, while in later releases besides enhanced core functionality very specific calculation functionalities are integrated that had a very weak relation with the other parts of the system.

Second, the team size of every release fluctuated between two and six people, all coming from the Apex maintenance team. The total team size fluctuated between four and nine people, with a maximum of one FTE to be scaled up or down each month. This might have affected the realized productivity, however, no additional data on team changes were available for our study.

Third, the decline in relative growth over time, as depicted in Figure 4, indicates a shift from releases that include a large amount of new functionality towards a focus on enhancements on existing functionality. We assume this to be a normal pattern in situations where systems get more mature. Any form of relation could be assumed here with the improving productivity as depicted in Figure 3. After all, we assume the team members got to know the system better over time, and the amount of adjustments on existing (and therefore known) functionality grew, both helping to improve the average productivity.

Fourth, and last, we see that productivity went up over time, but we don't know whether more improvement would be feasible. Although the assumption that we described at the start of this study, stating that any correlation exists between a positively skewed distribution of effort data and a deteriorating productivity, is not confirmed by the tests that we performed, these tests do not say anything about the possibility that the improvement of productivity might be higher in case effort was not characterized by a positively skewed distribution.

Yet, the tests we perform seem to show clearly that our initial assumption is not true for the subject public sector institution and the applicable data sample.

5.1 Validity of Evaluation

With regard to construct validity, the degree to which a test measures what it claims to be measuring, a remark is in place on FPA. Functional documentation was used as a source for FPA; a consequence is that low quality documentation could have led to low quality FPAs, however, the functional documentation was assessed as of good quality, FPAs were prepared by people who knew the system well, and we thoroughly reviewed all on completeness and correctness.

The functional size of each enhancement release was determined by the lead architect of the enhancement team. The architect had received internal training from a certified function point analyst. The certified analyst verified most of the function point analyses that were made in 2008 and 2009. Occasional verification of larger releases took place in later years. The determined functional size of each release was reviewed and approved by the quality management of the commissioning institution before contractor fees were made final.

Concerning internal validity, we warranted the extent to which a conclusion is based on our study, by normalizing all project data with the functional size in FPs. By doing so we were able to objectively compare performances of all releases in order to minimize systematic error. Still the limited number of releases in our sample holds a risk that outliers can have an effect on the outcomes of the study, and that bias can be applicable. However, we tried to mitigate these risks where possible.

A remark can be made on testing our hypotheses. The more you test your hypotheses, the more you can be confident about your assumption. Therefore, we clearly state that more research is needed to make exclusive claims about our proposed assumption.

On external validity we need to emphasize that due to the limited size of the subject sample it is far too early to generalize the study results to settings outside the study. We conducted the study only within one public sector institution, so the results may not generalize elsewhere. Since we did not find any other study on a comparable assumption, we cannot predict what the outcome of our assumption will be once studied in other institutions or companies. Besides that, the survey has limited generalizability due to the relatively small sample of 25 releases.

5.2 Relation to Existing Evidence

From our analysis of related work, it is clear that the effects of distributions of effort data on the performance of software projects is a topic that has received little attention from the research community. Yet this is a topic of high practical value, which can have a major impact on success or failure of a software portfolio as a whole. However, one needs to keep in mind that other studies might find different outcomes; e.g. the results of [20], which analyzes the usage of temporal data for predictive modelling of software defect, indicate that estimators did not learn over time.

5.3 Impact / Implications

Realizing the impact that the distribution of effort can have on a software portfolio as a whole, we argue that more research is needed to understand the real effects of this phenomenon. Our study emphasizes a portfolio approach, in which performance analysis is considered for the full software portfolio of an organization. A major pre-requisite for this approach is the availability of historical project data; implying an organizational aim for a long term research approach.

The necessity of historical project data for such research is likely also one of the causes why studies on long-term effects of effort data have received limited attention in the research community, since few researchers have access to such data. A way out of this dilemma may be opening up performance data for government-funded projects, making them available for researchers, by including this in contract conditions.

6. CONCLUSIONS AND FUTURE WORK

The key contributions of this paper are that we show that a proposed assumption that assumes any relation between on the one hand a positively skewed distribution of effort data and on the other differences between planned and actual estimations becoming larger and a deteriorating productivity, is not applicable to a set of release data collected over a period of six years in a public sector institution in The Netherlands.

Although we find that the effort data is to be characterized as distributed positively skewed, and shows a shift over time from under-estimation to over-estimation, we do not find evidence for larger differences between planned and actual estimations and a deteriorating productivity; on the contrary, productivity improves over time. This might indicate that estimators in the subject organization have learned from historic projects. However, we argue there is no relationship between estimation and productivity, since many other factors might influence productivity too.

This study opens up insights on future research. What we describe in this paper is in fact theory building and testing. We formulate an assumption that attempts to describe the phenomena we are observing. Based on this assumption, we formulate a few predictions and consequently, hypotheses that need to hold, if our assumption holds as a whole.

However, an optimal way to test such an assumption would be to test our hypotheses as many times as we can. Testing can be done using many instruments; retrospective case studies, structured interviews with experts, and longitudinal studies. In the end, the more our hypotheses are tested, the more confident we can be about our assumption.

ACKNOWLEDGMENTS

We thank Georgios Gousios for his help and valuable feedback.

REFERENCES

[1] B. W. Boehm and K. J. Sullivan, "Software Economics: A Roadmap," in *ACM Future of Sofware Engineering*, 2000.

[2] B. Boehm, "Some future trends and implications for systems and software engineering processes," *Systems Engineering*, vol. 9, no. 1, pp. 1-19, 2006.

[3] 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.

[4] H. Suelmann, "Putnam's Effort-Duration Trade-Off Law: Is the Software Estimation Problem Really Solved?," in *Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA)*, 2014.

[5] H. Huijgens, G. Gousios and A. v. Deursen, "Pricing via functional size: a case study of 77 outsourced projects," in *ACM 9th International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2015.

[6] H. Huijgens, R. v. Solingen and A. v. Deursen, "How to build a good practice software project portfolio?," in *ACM Companion Proceedings of the 36th International Conference on Software Engineering (ICSE)*, 2014.

[7] ISBSG, C. Jones and Reifer Consultants, "The Impact of Software Size on Productivity," ISBSG.

[8] D. Sjoberg, T. Dyba and M. Jorgensen, "The future of empirical methods in software engineering research," in *IEEE Future of Software Engineering (FOSE)*, 2007.

[9] F. Vogelezang and J. d. Vries, "The Added Value of Enhancement Function Points," in *IEEE Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA)*, 2014.

[10] NESMA, NESMA functional size measurement method conform ISO/IEC 24570, version 2.2, Netherlands Software Measurement User Association (NESMA), 2004.

[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] L. Eveleens and C. Verhoef, "Quantifying IT forecast quality," *Science of Computer Programming , Volume 74,* pp. P.934-988, 2009.

[13] M. Harman, F. Ferrucci and F. Sarro, "Search-Based Software Project Management," in *Software Project Management in an Changing World*, G. Ruhe and C. Wohlin, Eds., Springer, 2014, pp. 373-399.

[14] K. Moløkken and M. Jørgensen, "A review of surveys on software effort estimation," in *IEEE Proceedings of the International Symposium on Empirical Software Engineering (ISESE)*, 2003.

[15] 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.

[16] 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.

[17] F. Sarro, A. Petrozziello and M. Harman, "Multi-Objective Effort Estimation," in *Proceedings of the International Conference on Software Engineering (ICSE)*, 2016.

[18] 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.

[19] M. Shepperd and S. MacDonell, "Evaluating prediction systems in software project estimation," *Information and Software Technology,* vol. 54, no. 8, pp. 820-827, 2012.

[20] W. B. Langdon, J. Dolado, F. Sarro and M. Harman, "Exact Mean Absolute Error of Baseline Predictor, MARP0," *Information and Software Technology,* 2016.

[21] A. Abran, I. Silva and L. Primera, "Field studies using functional size measurement in building estimation models for software maintenance," *Journal of Software Maintenenace and Evolution: Research and Practice,* vol. 14, no. John Wiley & Sons, Ltd., pp. 31-64, 2002.

[22] A. F. Minkiewicz, "The Evolution of Software Size: A Search for Value," *Software Engineering Technology,* vol. March/April, pp. 23-26, 2009.

[23] 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.

[24] 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.

[25] A. Abran, Software Project Estimation: The Fundamentals for Providing High Quality Information to Decision Makers, Wiley-IEEE Computer Society Press, 2015.

[26] 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.

[27] M. Harman, S. Islam, Y. Jia, L. Minku, F. Sarro and K. Srivisut, "Less is More: Temporal Fault Predictive Performance Over Multiple Hadoop Releases," in *Proceedings of the 5th International Symposium on Search-Based Software Engineering (SSBSE)*, 2014.

[28] D. P. Douane and L. E. Seward, "Measuring Skewness: A Forgotten Statistic?," *Journal of Statistics Education,* vol. 19, no. 2, 2011.

[29] H. Huijgens and F. Vogelezang, "Do Estimators Learn? On the Effect of a Positively Skewed Distribution of Effort Data on Software Project Productivity - Technical Report TUD-SERG-2016-004," Delft University of Technology, Delft, The Netherlands, 2016.

APPENDIX A – RESULTS FROM THE TESTS IN R

```
> Actual <- c(808, 749, 496, 1511, 146, 1037, 709, 364, 455, 632, 3709, 883, 1312,
857, 1008, 273, 322, 410, 222, 1209, 1536, 440, 126, 636, 287)
> library(e1071)
> Actual <- c(808, 749, 496, 1511, 146, 1037, 709, 364, 455, 632, 3709, 883, 1312,
857, 1008, 273, 322, 410, 222, 1209, 1536, 440, 126, 636, 287)
> library(e1071)
Warning message:
package ‘e1071’ was built under R version 3.1.3
> skewness(Actual)
[1] 2.464956
> kurtosis(Actual)
[1] 7.227966
```

TUD-SERG-2016-004
ISSN 1872-5392

SERG