Software Size Analysis of Small Real-Client Projects

Ali Afzal Malik, Barry W. Boehm

Center for Systems and Software Engineering – University of Southern California
Los Angeles, CA 90089, U. S. A.

alimalik@usc.edu, boehm@usc.edu

Abstract. *This paper presents an analysis of the results of an empirical study conducted to examine the relationship between the estimated and actual sizes of small real-client projects. Estimates of software size are collected at two different points of the software development life cycle – one immediately before rebaselining the project mid-stream and the other immediately after rebaselining. It is found that estimates made after rebaselining are usually more accurate. The accuracy of estimates based on the category of projects is also analysed. The results indicate that the correlation between estimated and actual size is much stronger in case of web-based projects vis-à-vis non-web-based projects.*

Keywords: Actual size, estimated size, SLOC, software cost estimation, software sizing

1. Introduction

Software size is one of the most important inputs of software cost estimation models such as Putnam's SLIM [Putnam 1978], RCA's PRICE-S [Freiman and Park 1979], and COCOMO II [Boehm et al. 2000]. The accuracy of the output of these models thus heavily banks on the accuracy of the size estimates fed to these models. An unrealistically low value of size for one of the main modules of a software-intensive system, for instance, can lead to an underestimate of the overall effort and time needed to build that system.

Given that size estimates play a crucial role in the overall software cost estimation process it becomes vital to monitor these estimates. Comparing and contrasting size estimates with actual values of software size obtained from past projects allows us to gain useful insights on the gap between these values. This, in turn, allows us to figure out ways to improve our estimates so that this gap can be narrowed.

An opportunity to study and analyze the relationship between estimated and actual size of small real-client projects is provided by USC's annual two-semester-long series of graduate-level, team-project-based software engineering courses. Using the terminology of Rational Unified Process (RUP) [Kruchten 2003], the first course – Software Engineering I – introduces students to the activities of the Inception and Elaboration phases of the software development life cycle while the second course – Software Engineering II – exposes them to the activities carried out during the Construction and Transition phases.

Project teams mostly consist of four to six on-campus students and two off-campus students. The off-campus students are mainly responsible for verification and validation of the artifacts generated by the development core of the team consisting solely

of on-campus students. Clients of these projects are usually neighborhood organizations (e.g. California Science Center) or departments within USC (e.g. USC Libraries). Negotiations between the student teams and the clients help in prioritizing the requirements for these projects in order to facilitate their completion within two semesters (approximately 24 weeks). This empirical study analyzes the software size data of 19 such projects [SE I 2008, SE II 2008] done at USC during the past few years.

The rest of this paper is organized as follows. Section 2 presents the methodology used for gathering the size data for this empirical study. Section 3 summarizes the results while Section 4 contains a discussion on the salient quantitative and qualitative aspects of these results. Finally, Section 5 concludes with a summary of our major findings along with a brief description of the future work in this area.

2. Methodology

The first main step in gathering data for this empirical study was identifying projects appropriate for analysis. We analyzed projects of four different years: 2004, 2005, 2006, and 2007. After filtering out COTS-intensive projects and projects with incomplete data we ended up with a total of 19 projects. A brief summary of these projects appears in Table 1.

Table 1. Projects summary

S#	Year	Project	Type
1	2004	Online Bibliographies on Chinese Religions in Western Languages	Web-based database
2	2004	Data Mining of Digital Library Usage Data	Data mining
3	2005	Develop a Web Based XML Editing Tool	Web-based application
4	2005	EBay Notification System	Stand-alone application
5	2005	Rule-based Editor	GUI
6	2005	CodeCount™ Product Line with XML and C++	Code Counter Tool
7	2006	California Science Center Newsletter System	Web-based database
8	2006	California Science Center Event RSVP System	Web-based database
9	2006	USC Diploma Order/ Tracking Database System	Web-based database
10	2006	USC Civic and Community Relations (CCR) web application	Web-based database
11	2006	Student's academic progress web application	Web-based database
12	2006	New Economics for Woman (NEW)	Web-based database
13	2006	Early Medieval East Asian Tombs	Web-based database
14	2006	USC CONIPMO	Cost model
15	2007	USC COINCOMO	Cost model
16	2007	BTI Appraisal Projects	Stand-alone database
17	2007	LAMAS Customer Service Application	Web-based database
18	2007	BID review System	Stand-alone database
19	2007	Los Angeles County Generation Web Initiative	Web-based database

All of these projects were custom-development projects that followed the same MBASE/RUP [Boehm et al. 2005, Kruchten 2003] software development process. As shown by the last column in Table 1, these projects were of different types. The majority (11 out of 19), however, were web-based.

The other main step in gathering size data was selecting the appropriate anchor point milestones [Boehm 1996] which could act as viable sources of estimates. We choose the Life Cycle Architecture (LCA) and the Rebaselined Life Cycle Architecture (RLCA) milestones to gather two sets of estimated size values – one immediately before rebaselining and the other immediately after rebaselining. The reason for choosing the LCA milestone is that it marks the end of the Elaboration phase. Thus estimates available at this stage are much more mature as compared to estimates produced during the Inception phase. The RLCA milestone, which occurs at the beginning of the Construction phase, was chosen since rebaselining of a project may result in the modification of its scope thereby enabling the team to come up with a revised estimate of software size.

Our first set of estimated size values was obtained using the last version of the Life Cycle Planning (LCP) document [Boehm et al. 2005] produced in the Fall semester. This version was produced at the time of the LCA milestone. The second set of estimated size values were obtained from the LCP of the RLCA package produced in the Spring semester.

Both the estimated and the actual sizes of these projects were available in terms of logical source lines of code (SLOC). The actual logical size was measured according to the SLOC counting standard defined in [Nguyen et al. 2007]. The values for the actual size of these projects were obtained from size reports or CodeCount™ [CodeCount™ 2009] runs available at the Transition Readiness Review (TRR) milestone or as part of the Final Deliverables in the Spring semester. In the rare event when the development language used by the project was not supported by USC's CodeCount™ tool, a manual count of the actual logical SLOC was available.

3. Results

Table 2 summarizes the size data of the 19 projects listed in Table 1. For the sake of brevity, project names have been omitted from Table 2. The serial numbers (first column), however, are the same as in Table 1. Estimated logical SLOC at two different milestones – LCA and RLCA – are given along with the actual logical SLOC of the delivered product. Also shown is the percentage difference between estimated size at each of the two milestones and the actual size. The last column indicates whether the latter estimates (i.e. at RLCA milestone) are better, worse, or the same as the earlier ones (i.e. at LCA milestone).

Table 2. Estimated size versus actual size

	Estimates			% Difference with Actual		
S#	LCA (SLOC)	RLCA (SLOC)	Actual (SLOC)	LCA	RLCA	Is RLCA better?
1	5014	5014	5931	18.29	18.29	Same
2	3450	2535	2134	-38.14	-15.82	Yes
3	3350	4100	3876	15.70	-5.46	Yes
4	2000	2000	899	-55.05	-55.05	Same
5	2462	2462	2555	3.78	3.78	Same
6	2150	2150	1235	-42.56	-42.56	Same
7	3625	3625	3007	-17.05	-17.05	Same
8	3400	3400	4943	45.38	45.38	Same
9	3200	2900	1202	-62.44	-58.55	Yes
10	***3350***	***2100***	***4716***	***40.78***	***124.57***	***No***
11	2750	2750	457	-83.38	-83.38	Same
12	2500	2500	2488	-0.48	-0.48	Same
13	3500	3300	1973	-43.63	-40.21	Yes
14	3282	4203	5337	62.61	26.98	Yes
15	6219	6219	3386	-45.55	-45.55	Same
16	2212	2568	3709	67.68	44.43	Yes
17	2805	2805	2948	5.10	5.10	Same
18	2756	2756	5251	90.53	90.53	Same
19	4942	5115	7228	46.26	41.31	Yes

Figure 1 shows the relationship between the estimated size values at LCA milestone and the actual size values. Figure 2, on the other hand, shows the relationship between size estimated at RLCA milestone and the actual size.

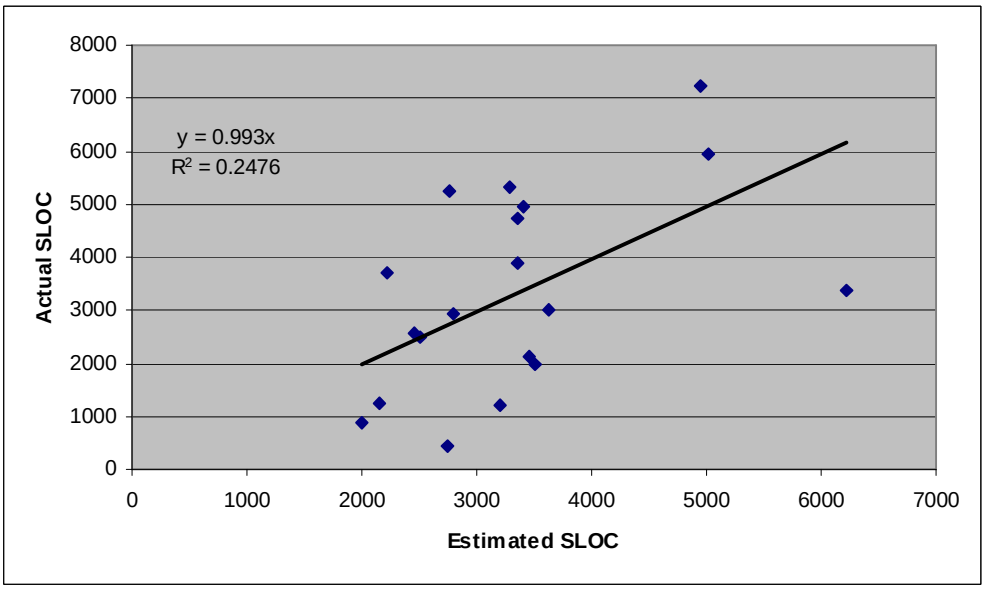

Figure 1. LCA estimates versus actuals

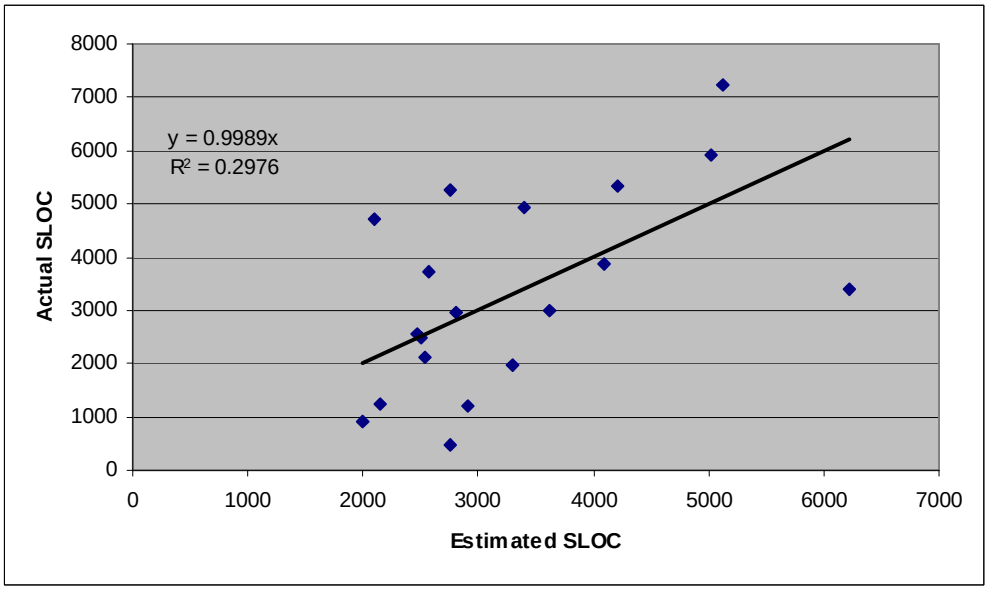

Figure 2. RLCA estimates versus actuals

Figure 3 shows the relationship between size estimated at RLCA milestone and the actual size for web-based projects only. Figure 4 shows the same relationship for non-web-based projects.

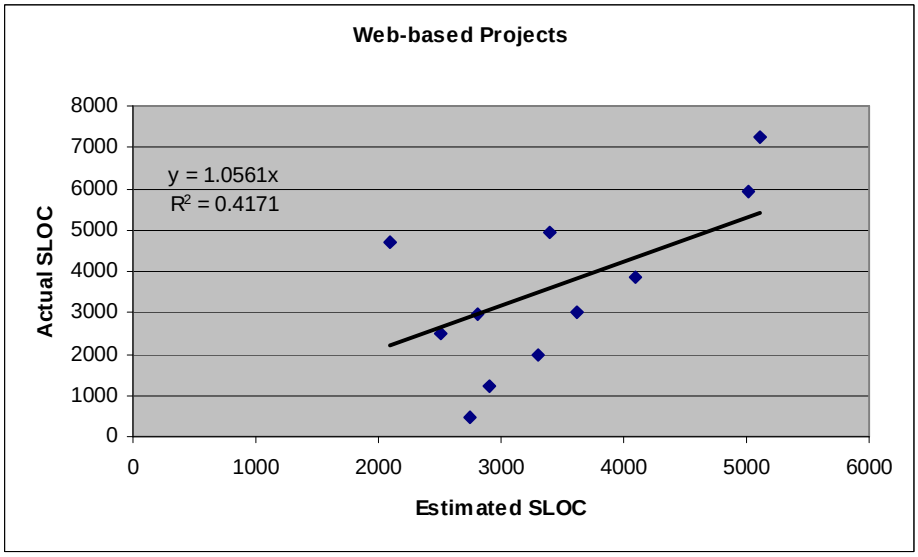

Figure 3. RLCA estimates versus actuals for web-based projects

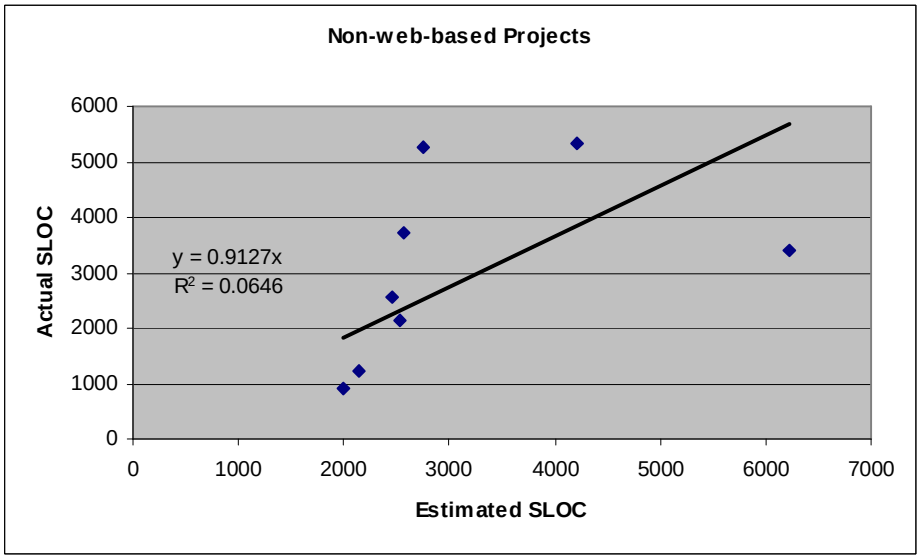

Figure 4. RLCA estimates versus actuals for non-web-based projects

Each of the four figures above (i.e. Figure 1 through Figure 4) contains a scatter plot along with a fitted trendline. The equation of the fitted regression line as well as the coefficient of determination (R^2) is included in these figures. The former shows the slope of the trendline while the latter indicates the strength of the relationship. Here it must be pointed out that the y-intercept of all four trendlines has been set to zero. This makes intuitive sense since a project which is not expected to contain even a single SLOC will most likely not be implemented in terms of SLOC.

4. Discussion

A quick glance at the results shown in the previous section reveals a few important points. First and foremost, as shown by the last column of Table 2, the RLCA size estimate is worse than the LCA size estimate in ***only one*** of the 19 cases (shown by the row in bold italics). In all of the rest, it is either better than the LCA estimate or the same. This evidence strongly suggests that rebaselining the project leads to a better estimate of the actual size of the product.

As far as the correlation between the estimated size and the actual size is concerned, RLCA estimates exhibit only a slightly better correlation. A comparison between Figure 1 and Figure 2 indicates this difference. LCA estimates (Figure 1) show an R^2 of 0.2476 while RLCA estimates (Figure 2) exhibit an R^2 of 0.2976.

Both Figure 3 and Figure 4 depict the relationship between RLCA estimates and actual size. Figure 3, however, shows the relationship for web-based projects while Figure 4 shows the relationship for non-web-based projects. Out of a total of 19 projects, 11 are web-based and 8 are non-web-based. A comparison of these two figures reveals another important point: the correlation between estimated size and actual size is much stronger in case of web-based projects ($R^2 = 0.4171$) versus non-web-based projects ($R^2 = 0.0646$).

5. Conclusions and Future Work

The results of this empirical study on the relationship between estimated size and actual size of small real-client projects suggest a couple of important points. First of all, size estimation is an on-going process. Size estimates should not be confined to the start of the project only. They should be revised as more information becomes available (e.g. during rebaselining) so that one can get more accurate estimates of a project's budget and schedule. Secondly, estimation of certain types of projects (e.g. web-based projects) is much more accurate than that of other types of projects. As part of our future work, we will try to explore the reasons for this divide between the size estimation accuracy of different types of projects.

6. References

Boehm, B. (1996). "Anchoring the Software Process", *IEEE Software 13(4)*, pages 73–82.

Boehm, B., Abts, C., Brown, A., Chulani, S., Clark, B, Horowitz, E., Madachy, R., Reifer, D., and Steece, B. (2000), Software Cost Estimation with COCOMO II, Prentice Hall.

Boehm, B., Klappholz, D., Colbert, E., et al. (2005). "Guidelines for Lean Model-Based (System) Architecting and Software Engineering (LeanMBASE)", Center for Software Engineering, University of Southern California.

CodeCount™ (2009). USC's Center for Systems and Software Engineering, http://csse.usc.edu

Freiman, F.R. and Park, R. E. (1979). "PRICE Software Model–Version 3: An Overview", *Proc. IEEE-PINY Workshop on Quantitative Software Models*, pages 32-41.

Kruchten, P. (2003), The Rational Unified Process: An Introduction, Addison-Wesley.

Nguyen, V., Deeds-Rubin, S., Tan, T., and Boehm, B. (2007). "A SLOC Counting Standard", *The 22nd International Annual Forum on COCOMO and Systems/Software Cost Modeling*.

Putnam, L. H. (1978). "A General Empirical Solution to the Macro Software Sizing and Estimating Problem", *IEEE Trans. Software Engr.*, pages 345–361.

SE I (2008). Links to websites of all past semesters of Software Engineering I (CSCI 577A) course at USC, http://sunset.usc.edu/csse/courseroot/course_list.html#577a

SE II (2008). Links to websites of all past semesters of Software Engineering II (CSCI 577B) course at USC, http://sunset.usc.edu/csse/courseroot/course_list.html#577b