RESEARCH ARTICLE OPEN ACCESS

Review of Effort Distribution in IT Companies

Reetuparna Mukherjee[1], Taniya Gupta[2], Mythili Thirugnanam[3]
School of Computer Science and Engineering
VIT University, Vellore

Abstract:
Effort distribution, be it by phase or activity, is an important aspect of SDLC. Yet it is often overlooked in the process of cost estimation. Poor effort allocation is one of the root causes of rework owing to early activities being insufficiently resourced. This paper presents various phase effort distribution patterns and variation sources. The analysis results of these patterns show some consistency in effects of size of software and team size on code and test phase distribution variations, and some considerable deviations in design, requirements, and transition phases, compared with recommendations in the COCOMO model. Software size, in turn, depends on the small-medium, medium-large companies having different schemes. Finally, the major findings of this paper discusses about threats to validity and presents general guidelines in directing effort distribution across the various software development methods, time duration of the phases of SDLC and the team strength. Based on the above factors, effort distribution can be estimated.

Keywords — ***COTS, COCOMO, RUP, SLIM, SEER-SEM, ANOVA***

INTRODUCTION

Effort distribution methods have been studied and evolved over the past few decades. Different methods employed include knowledge-based modeling, parametric modeling, fuzzy logic, neural networks, dynamic modeling or case based reasoning to increase the accuracy of estimation and to improve the estimation capability with respect to existing development paradigms, e.g. COTS or open source-based development . However, despite the intensive study on estimation methods, it still is a great challenge for software projects to be completed successfully as per recent studies. The major limitations that impede software project from meeting the success criteria, viz. within budget, on time, and with specified functionalities. Limited progress to fully understand the characteristics of emergent development paradigms and its implications to cost estimation complicates process-oriented decisions in design imperatives and cost reconciliation. According to the waterfall development model, there is restricted research effort towards understanding and analyzing the root causes that lead to the variations in effort distribution patterns. Effort distribution, be it by phase or activity, is an important aspect of SDLC. Yet it is often overlooked in the process of cost estimation. Bad effort distribution is among the significant causes of reworking that is due to insufficiently resourced activities. Distribution of effort in software engineering process has been the platform for facilitating more reasonable software project development planning, and is provided as one of the major functionalities in most planning or estimating methods. For example, a work breakdown structure of the total estimate over different phases or activities with corresponding percentage of effort associated with each phase. With numerous estimation methods and tools becoming available, users are often left with greater difficulty and confusion in effort distribution because of the large variation in the SDLC activities. Many software practitioners solely rely on Rule of Thumb or expert judgment to handle project phase effort planning. Compared with a total estimate, it is more important to develop an accurate phase-based distribution in order to facilitate strategic resource allocation and planning. This paper investigates the general effort distribution profiles of different development type, software size, team size, business area, etc.

EXISTING WORK ANALYSIS

The assessment that is conducted on software development project data collected from various software firms, resulted in the production of several widely adopted bases to guide software estimation practices. Norden [14], in his work to cost estimation field, found the Rayleigh Curve to be a good staffing level approximation to hardware projects. Though inapplicable in software development projects owing to slow early build-up and long-tail effects, Rayleigh distribution has influenced many later models in terms of effort distribution, such as the COCOMO [6, 11] and the SLIM [15]. Based on studies conducted on effort distribution data, the waterfall activity breakdown quantities in the COCOMO model [6, 11] was presented by Boehm, and the lifecycle effort distribution structure used by the Rational Unified Process (RUP) [12] was given by Krutchen. The way of handling phase distribution, as per COCOMO 81 model's [11], is to provide effort ratios for every development phase of SDLC. It also defines 5 levels of project scale, 3 development modes, phase-sensitive effort multipliers to help derive more accurate allocation decision according to specific project needs. As per the COCOMO model, with increase in software size, integration and test phase should be given more focus, than the code phase. This enables the users and effort multipliers to get the more estimated and accurate result i.e. concluded phase wise, by keeping the record of the different effort ratings . This offers a method to adapt the basic distribution scheme which is better and in accordance with the project requirements and situation. But, the use of detailed COCOMO model leads a complex procedure to produce the forms and tables having restrictive steps, which requires intensive project knowledge. The COCOMO II [6] model combines all the effort multipliers that is same for all the phases used for the development purposes owing to lack of detailed COCOMO calibration data and simplification of model usage. The COCOMO II extends the COCOMO 81 model to include the "plans and requirements" and "transition" phases into its waterfall lifecycle effort distribution scheme, and a MBASE/RUP phase/activity distribution scheme adopted from [12]. This takes a toll on flexibility in providing more relevant effort distribution guidance to meet the project needs as compared to COCOMO 81 model.

There exists large variation in the SDLC phases in different methods of estimation, resulting in lack of phase distribution data. This in turn complicates other processes viz. synthesis of lifecycle concept, model usage, collection and analysis of data, and cost model calibration.

COCOMO 81 [11] distributes effort among Plan and requirement, preliminary design, detailed design (25%), code (33%), Integration & Test (25) as shown in the figure below.

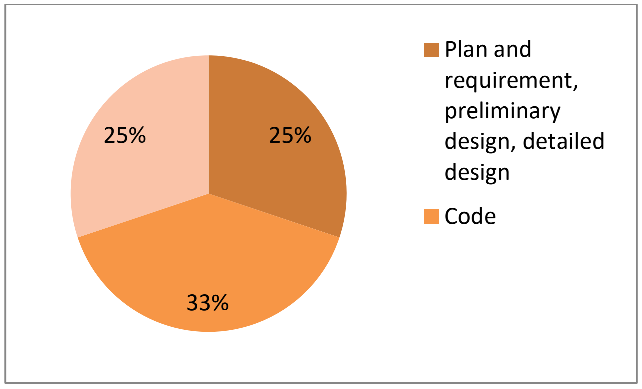

Fig 1. Effort Distribution in COCOMO 81

Figure 2 shows how COCOMO II [16] distributes effort :Waterfall distribution scheme: Planning & requirement (7%), preliminary design (17%),detailed design (25%), code (33%), Integration & Test (25), deployment & maintenance (12%);

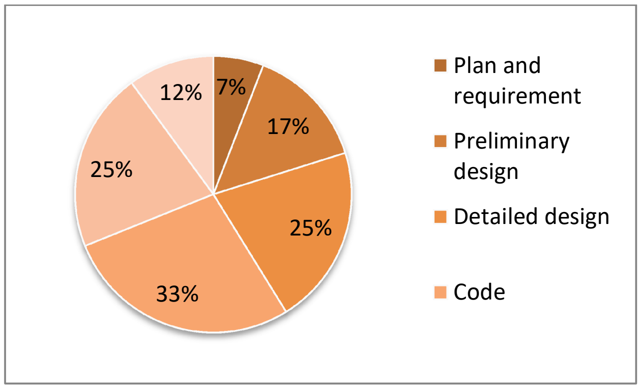

Fig. 2. Effort Distribution in COCOMO II

COCOMO II[16] MBASE/RUP distribution scheme: Inception(6%),Elaboration(24%), Construction(76%), Transition(12%) as shown in the figure below.

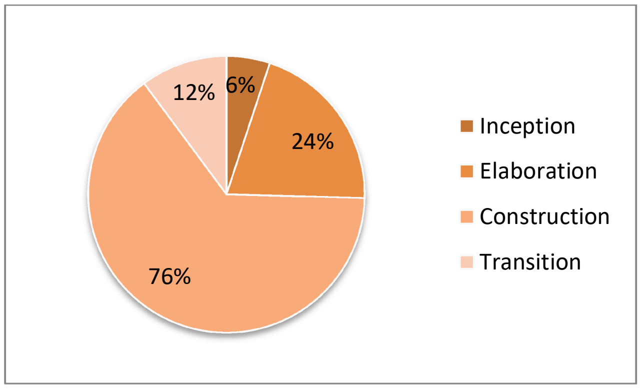

Fig. 3. Effort Distribution in COCOMO II MBASE/RUP

RUP [12] allots 5% to Inception, 20% to Elaboration, 65% to Construction, 10% to Transition as shown in the figure below.

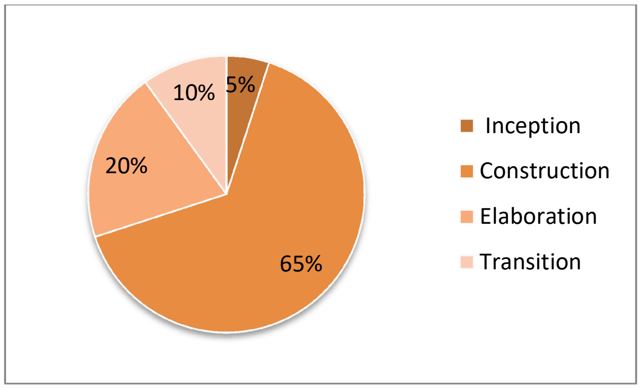

Fig. 4. Effort Distribution in RUP

COCOTS [6] distributes effort among Assessment, Tailoring, Gluecode & Integration. There is no unified distribution guideline. SLIM [16] distributes among Concept Definition, Requirement & Design, Construct & Test, Perfective Maintenance. SEER-SEM allots effort among early specification, design, development, delivery & maintenance.

Heijstek and Chaudron [10] reported empirical data on effort distribution used model-based development, and confirmed the similarity between the original RUP hump-chart and the reported distribution [12]. Yiftachel et. al. [18] came up with an economic model for optimal resource allocation across various phases of development. Optimal effort distribution across phases based on defect-introduction and defect-slippage considerations have been studied [19, 20] but there still exists notable inconsistency in results from different researchers regarding phase distribution.

DATA COLLECTION AND CLEANSING

Data collection: A study conducted by CSBSG sent an electronic questionnaire to a number of organizations. The questionnaire had some capability to check some errors automatically, such as data inconsistency and spelling error. On receipt of data from the organizations, the data quality is checked by a group of experts to confirm whether it can be added to the database. During this method, there is a specialized contact personnel between that expert group and the organization, and all the information about the organization is hidden in the data submitted to the expert group; the contact personnel reacts to a data problem, if found.

Data Cleaning: We only make use of a subset of the data, that too with only focused attributes for the purpose of our study. The data cleaning steps and attributes are summarized.

Table 1. Attributes covered in the study

METRIC	UNIT	DESCRIPTION
Size	Sloc	Total lines of code
Effort	Person-hour	Summary work effort
Plan phase effort	Person-hour	Work effort of plan phase
Requirements phase effort	Person-hour	Work effort of requirements phase
Design phase effort	Person-hour	Work effort of design phase
Code phase effort	Person-hour	Work effort of code phase
Test phase effort	Person-hour	Work effort of test phase
Transition phase effort	Person-hour	Work effort of transition phase
Development life cycle	Nominal	Waterfall, iterative, rapid
Team size	Person	Maximum size of the development
Development type	Nominal	New development, enhancement, redevelopment

RESULT AND ANALYSIS

The table below summarizes the statistical features of effort distribution percentage for each of the five development phases. Data has been collected from the CSBSG.

Table 2. Summary of phase distribution profile

Phase	Plan&Req%	Design%	Code%	Test %	Trans%
Min	1.82	0.62	6.99	4.24	0.06
Max	35	50.35	92.84	50.54	36.45
Median	15.94	14.21	36.36	19.88	4.51
Mean	16.14	14.88	40.36	21.57	7.06
Stdev	8.62	8.91	16.82	11.04	7.06

*Plan&Req Planning and Requirement
**Trans Transition

As per [6], "major variations in both waterfall and RUP phase distribution quantities come in the phases outside the core development phases". The collected data shows that coding and testing phases exhibit greater variation than other phases, e.g. the project with the least design phase effort distribution quantity is also the one with greatest coding effort distribution quantity.

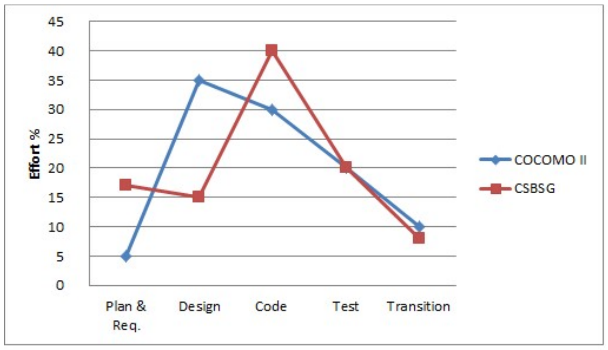

Fig..5 Comparison of the mean distribution profile of CSBSG dataset and the COCOMO II

Waterfall distribution quantities helps us examine the differences of individual phase distribution between CSBSG dataset and COCOMO II recommendations.

The similarity lies only in the last two phases whereas there are certain significant differences. The CSBSG dataset lays greater emphasis on Planning and Requirements phase. CSBSG allots 16.14% of overall effort to this phase as compared to a mere 6% in COCOMO II. Also, the design phase receives only 14.88% effort in CSBSG whereas COCOMO II allots 35%. The average distribution for code phase is 40.36 % in CSBSG and 27.8% in COCOMO II.

ANOVA analysis is used to examine the variance in term of effort distribution percentage for each phase that is explained by class variables, i.e. software size, team size and development type.

The data summarized in Table shows the degree of variance for the variation in individual phase effort distribution explained by an influencing factor. For example, it also indicates that for measuring effective software size and predictive development effort , FP is a stronger predictor and for determining appropriate adjustment for code and test phases, development type is also one of the major factor. However, team size is considered as less significant factor in effort distribution.

Table 3. ANOVA Results

Phase Distribution (%)	Development Type	Software Size		Team Size
		LOC Scale	FP Scale	
Planning & Requirement	3.96	14.80	13.65	5.50
Design	2.57	3.67	3.36	0.62
Code	12.22	7.93	20.56	0.02
Test	7.47	7.16	14.59	3.05
Transition	2.72	9.36	22.45	1.52

Through such model, these parametric model are considered as current method for estimation and benchmarking purposes. Due of the wide scope and variety of software projects, it is infeasible to deploy a single distribution scheme encompassing all projects. Analysis of empirical data helps to understand the variations in phase effort distribution and their possible causes. Identification of most frequently presented effort distribution patterns can help derive insightful conclusions leading to improved project planning practices. Following are general guidelines for software estimation and management.

- Phase effort distribution analysis should be performed along with cost estimation
- FP based software size and development type must be considered in effort distribution
- For enhancement type of projects, percentage of development effort in code phase decreases by 10.67%, and that for testing increases by 5.4%.
- As software size increases, distribution has an intensive focus on both coding and testing.
- Team size has a significant effect on testing phase. Hence team size should be decided accordingly.

CONCLUSION

Lack of recognition to process variations and lack of understanding of effort distribution patterns leads to poor effort distribution. This study aims to examine effort distribution profile and gain in-depth understanding of the variations and causes of phase effort distribution. It identifies significant differences and similarities with COCOMO II and further analyses patterns in phase effort distribution. Guidelines for efficient effort distribution amongst phases of SDLC have been provided for improved project management.

REFERENCES

[1] Boehm, B. and C. Abts, Software Development Cost Estimation Approaches-A Survey. Annals of Software Engineering, 2000. 10(1-4): p. 177-205.
[2] Jorgensen, M. and M. Shepperd, A System Review of Software Development Cost Estimation Studies. IEEE Transaction on Software Engineering, 2007. 33(1): p. 33-53.
[3] Basili, V.R.: Software development: a paradigm for the future. Proceedings of the 13th Annual International Computer Software and Application Conference, (1989) 471-485.
[4] A Framework Analysis of The Open Source Software Development Paradigm Proceedings of the 21st International Conference in Information Systems(ICIS 2000).(2000),58-69.
[5] The Standish Group. 2004 the 3rd Quarter Research Report, (2004). http://www.standishgroup.com
[6] Boehm, B.W., et al.: Software Cost Estimation with COCOMO II. Prentice Hall, NY(2000)
[7] Reifer, D.: Industry software cost, quality and productivity benchmarks, software. Tech News 7(2) (July 2004)
[8] Kroll, P.: Planning and estimating a RUP project using IBM rational SUMMIT ascendant. Technical report, IBM Developer works (May 2004)
[9] QSM Inc.: The QSM Software Almanac: Application Development Series (IT Metrics Edition) Application Development Data and Research for the Agile Enterprise.Quantitative Software Management Inc., McLean, Virginia,USA (2006)
[10] Heijstek, W. and Chaudron, M. R. V.: Effort distribution in model-based development. 2nd Workshop on Model Size Metrics (2007)
[11] Boehm, B.W.: Software Engineering Economics. Prentice Hall, (1981)
[12] Kruchten, P.: The Rational Unified Process: An Introduction. Addison-Wesley Longman Publishing Co., Inc., Boston,MA, USA (2003)
[13] Wohlin, C.: Distribution Patterns of Effort estimation. EUROMICRO Conference (2004)
[14] Norden P.V.: Curve Fitting for a Model of Applied Research and Development Scheduling. IBM J. Research and Development, Vol. 3, No. 2, (1958), 232-248.
[15] Putnam, L. and Myers, W. (1992), Measures for Excellence, Yourdon Press Computing Series.
[16] He, M., et al.: An Investigation of Software Development Productivity in China. ICSP 2008, (2008) 381-394
[17] SEER-SEM. http://www.galorath.com/index.php
[18] Peleg Yiftachel, et al.: Resource Allocation among Development Phases: An Economic Approach. EDSER'06, May 27, 2006, Shanghai, China, (2006) 43-48
[19] Huang, L., Boehm, B.: Determining how much software assurance is enough?: a value-based approach. In: EDSER '05: Proceedings of the seventh international workshop on Economics-driven software engineering research, NY, USA,ACM Press (2005) 1–5
[20] Yiftachel, P., Peled, D., Hadar, I., Goldwasser, D.: Resource allocation among development phases: an economic approach. In: EDSER '06: Proceedings of the 2006

international workshop on Economics driven software engineering research, New York, NY, USA, ACM Press(2006) 43–48.
[21] Jiang, Z., Naudé, P. and Comstock, C.: An investigation on
the variation of software development productivity. International Journal of Computer, Information, and Systems Sciences, and Engineering, Vol. 1, No. 2 (2007) 72-81
[22] ISBSG Benchmark Release 8. http://www.isbsg.org
[23] SPR Programming Languages Table 2003, Software Productivity Research. ttp://www.spr.com
[24] Agrawal, M., Chari, K.: Software Effort, Quality and Cycle
Time: A Study of CMM Level 5 Projects. IEEE Transactions on Software Engineering, Vol. 33, No. 3 (2007) 145-156
[25] Software Measurement Services Ltd. 'Small project', 'medium-size project' and 'large project': what do these terms mean?"
http://www.measuresw.com/library/Papers/Rule/RulesRelativeSizeScale%20v1b.pdf
[26] http://www.csbsg.org