Cost Estimation for Global Software Development

Patrick Keil
Institut für Informatik – I4
Technische Universität München
Garching, Germany
49.89.289.17386
keilp@in.tum.de

Daniel J. Paulish
Siemens Corporate Research, Inc.
755 College Road East
Princeton, NJ, USA
1.609.734.6579
daniel.paulish@siemens.com

Raghvinder S. Sangwan
Engineering Division
The Pennsylvania State University
Malvern, PA, USA
1.610.725.5354
rsangwan@psu.edu

ABSTRACT

Global software development has gathered pace in recent years. Many software projects now involve asynchronous collaboration among geographically distributed teams several time zones apart. Software cost estimation for such projects becomes challenging due to factors such as effort expended in team building and knowledge transfer, creating an architecture of the software product that can be easily distributed and that minimizes cross-site communication, facilitating communication among remote teams collaborating on parts of the architecture that are interrelated and their day-to-day governance.

In this paper we structure the additional cost drivers of distributed development and examine the significance of each of these factors as a contributor to the overall cost of a software development project. We suggest ways in which COCOMO II, the most widely used software development cost estimation model, can be tailored to account for these additional complexities.

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management – *Cost estimation.*

General Terms

Economics, Human Factors, Management, Measurement

Keywords

Cost Estimation, Global Software Development, COCOMO

1. INTRODUCTION

With an expanding global marketplace, a trend towards developing software in low cost countries, and the growing complexity and size of software systems, the percentage of projects that are globally distributed has been steadily increasing [7]. Siemens Corporate Research, Inc. (SCR) has been doing research aimed at developing a better understanding of the issues and impact of various practices with respect to Global Software Development (GSD) since GSD increases the requirements regarding development processes, project management practices, architecture, quality, collaboration tools and so forth.

The 1999 Standish Group report shows a significant correlation between project or team size and the project's success [21]. We believe that one major determinant for this correlation is the physical separation of teams which is more often the case the bigger the projects get. This physical separation, especially across several time zones, requires additional activities and effort; e.g. for team building, knowledge transfer for asynchronous collaboration, creating an architecture that is easily distributed and that minimizes cross-site communication, and facilitating communication among teams working on parts of the architecture that are interrelated [4][5][6][11][12][18]. This additional effort translates into a substantial planning, coordination and control overhead in the day-to-day governance of GSD projects.

Yet many corporations are choosing to partner with software development companies in Eastern Europe, South America, and Asia hoping for substantial cost savings, mainly based on lower labor rates offered by these organizations. This can, however, be misleading considering these other factors that play a significant role in offshore sourcing. In this paper, we present an approach to improve the precision of COCOMO cost estimations for GSD. After a short introduction to COCOMO II, a widely used model for software development cost estimation, we analyze the sources of complexities in distributed software development projects and how well they are represented in COCOMO. We then propose some refinements to COCOMO II. Finally, we draw some conclusions pointing out the shortcomings of our approach and avenues for future research.

The results of our analysis can be used for the calculation of trade-offs between the decision to collocate or distribute the development of a software product. The overall goal of our work on these subjects is to provide a decision making framework for managers when faced with such decisions.

2. COST ESTIMATION WITH COCOMO

The Constructive Cost Model (COCOMO) was introduced in 1981 by Barry Boehm [2]. The enhanced version, COCOMO II, was presented in 1985 and has been further adapted since then [3]. COCOMO II is today's most used method for estimating cost of software projects [23]. It is accepted internationally and in organizations of all sizes.

Like the Function Point method [1][14][15], COCOMO II is an algorithmic estimation model with an explicit functional form.[1] It relates the dependent variable, effort as measured in person months, to independent variables. Total project costs can be easily derived from the resulting number of person months. The basic equation is

$$Effort = A \cdot x \cdot (Size)^B$$

with *Size* being measured in KDSI (kilos of delivered source instructions). The coefficient *A* is set at 3.0 to calibrate COCOMO II to the original COCOMO project database.

The exponent *B* in the equation represents the scaling factor which models the economies and diseconomies of scale encountered as a software project increases in size. If *B* equals 1, the economies and diseconomies of scale are in balance which corresponds to linear extrapolation. This is an approach used for estimations for small projects. But most often, $B > 1.0$ is assumed which means that the project exhibits diseconomies of scale. This is due to planning, communication and integration overhead (which can be partially eliminated by early risk management, by using thorough validated architecture specification, or by stabilizing requirements) as a project increases in size. If $B < 1.0$, the project exhibits economies of scale due to specialization gains such that if the project size is doubled the effort is less than doubled.

Looking at the details, *B* is a result of the following additive factors:

$$B = 1.01 + 0.01 \cdot \sum_i W_i$$

where W_i is the five factors precedentedness, development flexibility, architecture/risk resolution, team cohesion, and process maturity.

The obtained nominal person-month estimate can be refined substantially by multiplying it with several cost factors called effort multipliers (*EM*). They are estimated independently from each other and then added which leads to

$$PM_{adjusted} = PM_{no\min al} \cdot x \cdot (\prod_i EM_i).$$

There are 18 effort-multiplier cost drivers divided into four categories as shown in Table 1: Product, Platform, Personnel, and Project Factors.

As can be seen from Table 1, there is a single project factor that needs to cover all additional efforts arising through GSD called Multisite Development. This cost driver is assessed by averaging the two factors multisite collocation (ranges from fully collocated to international) and multisite communications (ranges from some mail to interactive multimedia). As we point out in the next section, this may be insufficient if we want to achieve a more precise estimation of cost for GSD projects. For example, the decision for the architecture has a great impact on effort and project cost since it influences the distribution and coordination of work between the sites. This is one of the aspects missing in the factor Multisite Development.

[1] Jones [14][17] analysed the relationship between numbers of LoC and Function Points for different programming languages.

Table 1: Effort-Multiplier Cost Drivers

Product	Required Software Reliability Database Size Product Complexity Required Reusability Documentation match to life-cycle needs
Platform	Execution Time Constraint Main Storage Constraint Platform Volatility
Personnel	Analyst and Programmer Capability Application Experience Platform Experience Language and Tool Experience Personnel Continuity
Project	Use of Modern Programming Practices Use of Software Tools Multisite Development Required Development Schedule Classified Security Application

3. SOURCES AND CHARACTERISTICS OF COMPLEXITIES IN GSD

There is an extensive body of research on remote and global collaboration, but only a few contributions present issues and practices that are extracted from real case projects [18]. Most of our findings have been validated in distributed Siemens projects of various sizes as reported in [13]. In addition, the literature is often focused either on communication, team building or tool support.

All these aspects are undoubtedly important, but our focus is on the combination of those factors and on their respective impact on project costs. For example, Bruegge et al. [4] found that even with a rich set of collaboration tools and some face-to-face meetings between certain team members, actual collaboration and information sharing between geographically remote teams was difficult and infrequent. Thus, as soon as you move one part of a project team to a different physical location, the original cost estimation, independent of the method that was used to calculate it, will be too low. When teams vary also in time zone, culture and language, the estimations get even more imprecise. Therefore it is necessary to analyze explicitly the determinants of "intra-project variation".

As sketched in section 2, the Project Factor Multiside Development in COCOMO II is determined by multisite collocation and multisite communications. This is our starting point. In the following, we try to identify, evaluate and estimate the drivers of these two factors and introduce additional factors that proofed to be relevant in GSD. For tailoring the COCOMO framework further, we start with a list of complexity factors resulting from geographic distribution of work using a taxonomy similar to that of Table 1. These complexity factors which we enumerate in the subsections below have been identified in several projects and presented in different contributions.

3.1 Product Factors

- *Precedentedness*: novelty of the software to be developed, indicating the degree of innovation which is directly proportional to the level of spontaneous communication, the need for specific domain knowledge and the frequency of unforeseen changes.

- *Architectural Adequacy*: work assignments have to be carefully crafted taking into account the organizational structure and the functional coupling among software units [24]. Therefore, the architecture has major influence on the efforts needed to coordinate the development phase. Indicators for the degree of Architectural Adequacy might be modularity, interface match and dependencies, communicability of the architecture, etc.

3.2 Personnel Factors

- *Cultural Fit*: closeness of team members' mental models [21] which is influenced by the combination of countries involved, the international experience of the teams, etc.

- *Skill Level*: educational level and language skills, indicating the formal abilities of remote team(s) [24].

- *Shared Understanding*: tacit knowledge that is required, indicating the level of completeness of documentation and specification and the common knowledge about goals [19][20].

- *Information Sharing Constraints*: representing competitive restrictions on information distribution, e.g. when working with external subcontractors or in security-sensitive environments.

The role of less formal factors, often called "soft skills" has been made clear in the extensive body of research on remote and global collaboration. Hersleb and Grinter [11] found that while organizations attempt many mechanisms to coordinate cross-site work, this is vulnerable to imperfect foresight and unexpected events. Solving these requires workers skilled at informal communication and negotiation. Moreover, GSD makes all these activities more difficult; the lack of subtle modes of communication often results in decreased perception that remote coworkers are helpful, decreased ability to react to changes quickly and increased project delays [12]. GSD has stronger needs for planned communication patterns and – paradoxically – for a higher flexibility and adaptivity of processes. We tried to capture these aspects in the Personnel Factors presented.

3.3 Project Factors

- *Novelty of Collaboration Model*: initial cost for the search of offshore partners and contract negotiation.

- *Tools and Infrastructure*: representing the homogeneity of the tool chains used in all sites and potential ramp-up costs for setting up the infrastructure in remote sites.

- *Physical Distance*: representing the potential overlaps of working time and, accordingly, the intensity of use of asynchronous communication media and collaboration tools [9][25].

4. TOWARDS A TEST BED FOR ANALYZING GSD-RELATD COSTS

The idea for our future work is to justify the factors provided and to add suitable effort multipliers to the list of 18 already provided in COCOMO II. To increase our understanding of how product, personnel, and project COCOMO factors are affected by distributed development, we have been collecting data from the Global Studio Project (GSP) [22]. This project simulates an industrial global software development project using student staffed central and remote development teams at seven sites in four continents. What we have observed from this data is that there is not only a cost associated with communications among sites, but there is also an ongoing cost associated with the staff that is added to the project team in roles that are primarily communication oriented. For example, the GSP has roles for "supplier managers". These staff are members of the central team responsible for interfacing with remote teams with a rule of thumb that each supplier manager can effectively interact with no more than ten remote developers.

Today, when estimating the project costs for the GSP, we apply COCOMO for each software component that will be developed by the distributed teams based on the size estimate for the component. This gives us a cost for a component developed at a single location. However, the costs associated with component development must be added to the costs associated with integrating the components. Thus, central team roles like the supplier managers plus architects, requirements engineers, project managers, integrators, and testers must be added to the cost estimate. Since these central team roles must be in place in early phase activities as well as throughout development, they can be substantial over the life of the software product development.

There are also quality costs that arise in the GSP. As a variation of the children's game 'Chinese Whisper' where everyone whispers a phrase sequentially from child to child in a circle, requirements, designs, project plans and technical decisions are likely to be misunderstood when the opportunities for instantaneous communication interaction are limited by time and distance. It doesn't matter how inexpensive the software is to develop if it does not meet products requirements. For example, if the remote labor rates are 1/3 those of a central collocated team, but the remote team must redo the implementation 3 times to get it right, has there been any cost savings from an outsourced approach? We suspect not in this case, since redoing the development takes calendar time and can delay the revenue stream of the product in the market.

The GSP provides us with a test bed for collecting many types of cost date that have been used to define and/or create COCOMO project factors associated with distributed development. What is most difficult is to understand what is behind the data. For example, does it cost more to develop a component in the GSP distributed environment because there are misunderstandings in communicating the requirements to remote teams, the requirements specifications are incomplete, or the developers are inexperienced with the domain? Nevertheless, careful analysis of cost data and project factors helps gain insights into what are potentially good or bad practices for distributed software development.

5. CONCLUSION

Although there is a strong business driver to offshore software development to lower labor cost countries, some of the savings are offset by increased costs. Some of these offsetting costs are related to costs of quality resulting from misunderstandings and some of them are due to staffing roles that may not exist in collated projects. In this paper we promote analysis of project factors to gain insights into comparing development costs for distributed software development projects as compared to collocated projects. Future work for this research includes

- verifying and improving the factors (are there better ways to classify and measure characteristics of a global development project; e.g., cultural differences?
- applying our GSD approach on projects while collecting cost data to calibrate the relevance of each project factor.
- introducing those factors found to be sufficiently measurable, disjunctive and covering most of the relevant cost drivers into the COCOMO II model to further refine cost estimations for GSD.

6. REFERENCES

[1] Albrecht, A. J. Measuring Application Development Productivity. *Proceedings of the Guide/Share Application Development Symposium* (Oct. 1979). 83-92.

[2] Boehm, B. *Software Engineering Economics*. Prentice-Hall, Englewood Cliffs, NJ, 1981.

[3] Boehm, B., Abts, C., Brown, A., Chulani, S., Clark, B., Horowitz, E., Madachy, R., Reifer, D., and Steece, B. *Software Cost Estimation with COCOMO II*. Prentice Hall, Upper Saddle River, NJ, 2000.

[4] Bruegge, B., Dutoit, A. H., Kobylinski, R. and Teubner, G. Transatlantic project courses in a university environment. *In Proceedings of the Seventh Asia-Pacific Software Engineering Conference* (December 05 - 08, 2000). APSEC. IEEE Computer Society, Washington, DC, 30

[5] Carmel, E. *Global Software Teams*. Prentice-Hall, Upper Saddle River, NJ, 1999.

[6] Casey, V. and Richardson, I. Virtual Software Teams: Overcoming the Obstacles, *Proceedings of the 3rd World Congress on Software Quality*, (September 26 - 30, 2005). Munich, to appear.

[7] Davison, D. Offshore Outsourcing: Market Overview. *METAspectrum*. October 2004.

[8] Edwards, H. K. and Sridhar, V. Analysis of the Effectiveness of Global Virtual Teams in Software Engineering Projects. *In Proceedings of the 36th Annual Hawaii international Conference on System Sciences* (January 06 - 09, 2003). HICSS. IEEE Computer Society, Washington, DC, 19.2.

[9] Evaristo, R. and Priklandnicki, J. *Distributed Software Development: Toward an Understanding of the Relationship between Project Team, Users and Customers*. ICEIS, 2003.

[10] Garmus, D. and Herron, D. *Function Point Analysis: Measurement Practices for Successful Software Projects*. Addison-Wesley, 2000.

[11] Herbsleb, J. D. and Grinter, R. E. Splitting the organization and integrating the code: Conway's law revisited. *Proceedings of the 21st international Conference on Software Engineering* (May 16 - 22, 1999). ICSE '99. IEEE Computer Society Press, Los Alamitos, CA, 85-95.

[12] Herbsleb, J. D., Mockus, A., Finholt, T. A. and Grinter, R. E. Distance, dependencies, and delay in a global collaboration. *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work* (Philadelphia, PA, United States). CSCW '00. ACM Press, New York, NY, 319-328.

[13] Herbsleb, J. D., Paulish, D. J. and Bass, M. Global software development at Siemens: experience from nine projects. *In Proceedings of the 27th international Conference on Software Engineering* (May 15 - 21, 2005). ICSE '05. ACM Press, New York, NY, 524-533.

[14] IBM Inc. *The Function Point Method*. 1983.

[15] International Function Point Users Group. *Function Point Counting Practices Manual, Release 4.1.1*. 2001.

[16] Jones, T. C. Backfiring. Converting Lines of Code to Function Points. *IEEE Computer. 28, 11* (Nov. 1995), 87-88.

[17] Jones, T. C. *Estimating Software Costs*. McGraw-Hill, New York, NY, 1998.

[18] Lassenius, C. and Paasivaara, M. Collaboration Practices in Global Inter-organizational Software Development Projects. *Software Process Improvement and Practice 8, 4* (Oct. 2003), 183-199.

[19] McChesney, I. and Gallagher, S. Communication and co-ordination practices in software engineering projects, *Information and Software Technology*, vol. 46, no. 7, 2004, 473-489.

[20] McComb, S., Green, S. and Compton, W. Project goals, team performance, and shared understanding, *Engineering Management Journal*, vol. 11, no. 3, 1999, 7-12.

[21] O'Hara-Devereux, M. and R. Johansen. *Global Work: Bridging Distance, Culture and Time*. The Jossey-Bass Management Series. 1994.

[22] Richardson, I., Milewski, A., Keil, P. and Mullick, N., Distributed Development – an Education Perspective on the Global Studio Project, *Proceedings of the 28th international Conference on Software Engineering* (May 20 - 28, 2006). ICSE '06. Shanghai, to appear.

[23] Snell, D. *COCOMO*. SCEW, 1997.

[24] Sosa, M., Eppinger, S. and Rowles, C. The Misalignment of product architecture and organizational structure in complex product development, *Management Science*, vol. 50, no. 12, December 2004, 1674-1689.

[25] Sosa, M., Eppinger, S., Pich, M., McKendrick, D. and Stout, S. Factors that influence technical communication in distributed product development: an empirical study in the telecommunications industry, *IEEE Transactions on Engineering Management*, vol. 49, no. 1, February 2002, 45-58.

[26] Standish Group International, Inc. *CHAOS: A Recipe for Success*, 1999.