ESTIMATION OF A SOFTWARE DEVELOPMENT PROJECT USING COCOMO II

Huanzhong Gu
Department of Computer Science and Operations Research
North Dakota State University
Fargo, ND 58105, USA
Huanzhong.gu@ndsu.nodak.edu

Jingpeng Tang
Department of Computer Science and Operations Research
North Dakota State University
Fargo, ND 58105, USA
Jingpeng.Tang@ndsu.nodak.edu

Vijayakumar Shanmugasundaram*
Department of Math and Computer Science
234 C Ivers, 901 8th St S
Concordia College
Moorhead, MN, 56562
shanmuga@cord.edu

Abstract

This paper describes the intricacies of estimation of a software development project using COCOMO II estimation/planning software. This COCOMO II estimation software is used to estimate the effort and schedule required to develop performance-monitoring software for a company. This performance monitoring software utilizes the centralized, integrated database system for monitoring the performance of various departments with in a multidisciplinary company. The purpose of developing such performance-monitoring system is to eliminate the various standards existed in different departments and to prevent small, and isolated data information maintained individually using spreadsheets and personally developed software. The individual requirement of four major selected departments as well as the requirement of integration with central database system of this monitoring software is collected. The effort, schedule, and costs for developing this performance-monitoring software project are evaluated and presented.

Keywords: Software Estimation, COCOMO II.

*** Contact Author**

INTRODUCTION

In an ever-changing computer technology world, database system has been an essential part of any company. However, the evolution of the technology have made

many companies' database system obsolete compared to the new advances. Therefore, many companies have either converted or in the process of converting their database system to utilize new technologies. Instead of pieces of individualized information floating around different personal computer in various departments maintained by individuals, the company we are working with has made commitment and effort to collect the information into a centralized database system such as Oracle.

However, individualized system still needs to be developed to maintain the performance, and maintenance monitoring services of these departments otherwise performed by individuals at their own specified times and standards. Such monitoring services are meant to provide easy and understandable presentation of the vast data collected by central database system. These services have to be suitable for each department. They need to have easy interface and simple, yet accurate performance information with regarding to either products, or processes, or customers depending upon the departmental functionalities.

To develop these monitoring services, many software package or high level programming language can be used. The only constrain is that it should be easy to communicate with the Oracle database. The software packages considered are Crystal Report (a reporting tool for large databases), Visual Basic, and other valid options. In this project, Visual Basic 5, Query type of software, and Report Generator are considered in comparison. They are rough equivalent of the tools that are likely to be used for developing these monitoring systems.

In this paper, COCOMO II is used as software estimation and planning software. It is an improved version of the COCOMO 81 [1] initially developed by Boehm in 1981. It can be used to make investment or financial decisions involving software development effort and setting budgets and schedules as a basis for planning and control [2].

METHODS

REQUIREMENTS DEVELOPMENT

Necessary information such as the requirements has to be obtained for estimation of the effort, schedule, and cost. This information usually would be gathered as official requirements given by company representatives. However, since the scope of the project is limited, no official requirements are used. Instead, an interviewing process that involved essential personnel is conducted to gather as much requirement information as possible. Those departments that participated in the interviewing process included departments such as processing, manufacturing, accounting, project accounting, program manager, and project key lead. Requirements are gathered from these various sources and noted down. For this project, four main representative departments are used for estimation. They are processing, manufacturing, accounting, and overall managerial departments.

FUNCTION POINTS

The gathered requirements are carefully studied to produce size of the motoring system project for each department. In this project, function points are used to represent size. This approach is based on the amount of functionality in a software project and a set of individual project factors [3, 4, 5]. The basic process and steps involved in converting requirements to function points are discussed as following:

1. Five components of function points are identified and categorized as,
    A. Internal Logical Files (ILF)
    B. External Interface Files (EIF)
    C. External Inputs (EI)
    D. External Outputs (EO)
    E. External inquiries (EQ)
2. From requirements, list of the number of each of these five components are determined,
3. For each identified instance of a functional component, the functional complexity is determined using Table 1, for EQ, EI and EO tables are used, and which ever one gives the higher value, that value is used,
4. The number of low, average, and high items in each of the five functional categories are counted,
5. The counts are placed in the appropriate places in Table 2,
6. Multiply and sum to obtain the total unadjusted function points, and that would be the input to COCOMOII software.

Table 1 Complexities in Function Point Components

Record Element Types	Data Element Types	Data Element Types	Data Element Types
	1 - 19	20 - 50	51 -
<2	Low	Low	Low
2 - 5	Low	Average	High
>5	Average	High	High

Complexity of ILF and EIF

File types Referenced (FTRs)	Data Element Types	Data Element Types	Data Element Types
	1 - 4	5 - 15	15 -
<2	Low	Low	Low
2	Low	Average	High
>2	Average	High	High

Complexity of EIs

File types Referenced	Data Element Types	Data Element Types	Data Element Types
	1 - 5	6 - 19	20 -
<2	Low	Low	Low
2 - 3	Low	Average	High
>3	Average	High	High

Complexity of Eos

Table 2. Unadjusted Function Points Table

	Low	Average	High
EI	X 3	X 4	X 6
EO	X 4	X 5	X 7
ILF	X 7	X 10	X 15
EIF	X 5	X 7	X 10
EQ	X 3	X 4	X 6

COCOMOII Model

COCOMOII post-architecture model is used to develop the estimates. Four modules are included in the model representing the four departments evaluated. The assumption for using the post-architecture model is that the project is ready to develop and sustain a fielded system and the life-cycle architecture is in place. We would be able to put more accurate information for various cost drivers and thus enable more accurate estimations. Total of seventeen effort multipliers are used to adjust the nominal effort. Total of five scale factors are used to account for the relative economies or diseconomies of scale encountered for software projects of different sizes. All these are maintained the same for all four modules.

SCALE FACTORS

The scale factors' settings are shown in Figure 1. Their range of values is from very low to very high. Precedentedness (PREC) reflects the similarities of this project to projects that the development team has undertaken in the past. This factor is dimmed to be very low because the development team that will implement the system has no prior similar project experience at all.

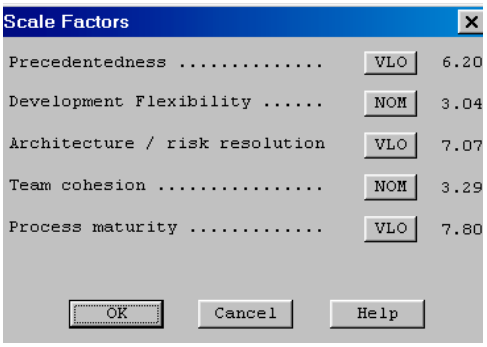

Figure 1. Scale Factors used in COCOMO II

The team would most likely to be assembled in-house, and in-house programmers in this case, has not been involved in this type of projects. This indicates that the team has considerable understanding of the objectives, but moderate related experience and has considerable need for innovation. Development flexibility (FLEX) refers to the amount of given in what actually must be developed. It is assumed to be nominal since little information is available.

The architecture/risk resolution (RESL) is the extension of architecture being completely specified and major risks being eliminated. The scope of this project has time limitation for going into details of risk management and thus is assumed to be very low. Team cohesion (TEAM) is assumed to be nominal. The development team members would be coming from likely the same department and would have basic teamwork skills. However, due to lack of team development project experiences in similar situations, even though they might have worked together before, they still would not get high team cohesion evaluations but nominal. Process maturity (PMAT) is one of the more influential factors because it measured different key process areas in a software project team. These key areas extend from requirement analysis, project planning, to quality assurance, and team training issues. Since the team is most likely dealing with this type of project the first time and is far from being a professional development team, the PMAT factor was given a very low value for representing the low level of process maturity of the development team.

EFFORT MULTIPLIERS

The effort multipliers' settings are shown in figure 2. They are grouped into four different categories: product, platform, personnel, and project. Their ranges of values are also from very low to very high.

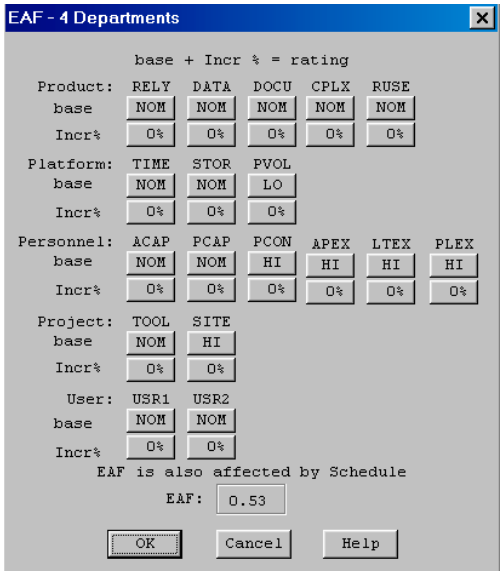

Figure 2. Effort Multipliers used in COCOMO II

Required software reliability (RELY) measures the extent to which the software must perform its intended function over a period of time. The failure of the developed system might cause moderate, easily recoverable losses for the fact that if the monitoring process would fail, even though automatic control would be off, the human control can still be assumed. The failure might cause some losses, but not considerable enough to be a major concern. Therefore, nominal value is assigned. Database size (DATA) is determined to be nominal as well in this case because the ratio of bytes in the testing database to SLOC (source lines of codes) would most likely between 10 and 100 and resulting in a nominal case. Product complexity (CPLX) and reusability (RUSE) are assumed to be nominal by virtual of estimation. The documentation (DOCU) suitability for life-cycle needs is given a nominal value because right-sized documentation is assumed. However, because the team would communicate more using other means of tools such as telephone, emails, team meetings, the documentation is more formed to show managers in this case. Therefore, nominal amount of documentation is assumed.

Execution time (TIME) and storage (STOR) constrains are not as significant in this case, hence we assume both nominal values. The platform volatility (PVOL) refers to the change of hardware/software these services calls on to perform their own tasks. In this company, the major change is not expected in a 12 months period while minor changes might occur monthly, therefore, a low value is assigned representing low volatility.

Since our programmers have previous programming experience and have worked on other projects related to using their analytical ability as well as programming ability, the analyst capability (ACAP) and programmer capability (PCAP) multipliers are assumed to be nominal. However, we believe that they have relatively high applications experience (APEX), platform experience (PLEX), and language and tool experience

(LTEX) because the amount of years they have been working on programming projects. Personnel continuity is considered high because annual personnel turnover for this job function was low (< 3% per year).

Toolsets used by the developers would be basic lifecycle tools with moderate integration, and thus bear the nominal value. The multisite development multiplier is set to be high because the degree of site collocation and communication support is relatively high.

The overall required development schedule (SCED) measures the schedule constraint imposed on the project team developing the software. The ratings are defined in terms of the percentage of schedule stretch-out or acceleration with respect to a nominal schedule for a project requiring a given amount of effort. In this case, we have no detailed discussion with regarding to stretch out and thus, assume a nominal value.

RESULTS AND DISCUSSIONS

The results of the function points' evaluation for the manufacturing engineering department are shown in Figure 3.

Manufacturing Engineering Department
Extracted from requirements extracted from interview

Data Functions:	**Transactional Functions:**
List of Internal Logical Files: 1. Product List o Product Type o Product Name o Product Cost o Time to Produce o Current Quantity o Time Duration o Specification o Special Notes o Service Lines o Expired Time o Error Rates o Placement Rates List of External Interface Files 1. Equipment List 2. Line Specific Process Data File 3. Customer List	List of External Inputs: 1. User Commands (text boxes) 2. Buttons 3. User Login/off 4. Product selection list List of External Outputs: 1. Production graphs (indicator) 2. Warning signs for outdated products 3. Product service line change 4. Product specification change 5. Updating of product list 6. Administrator login/off 7. Other screens (may be 4 to 5) List of External Queries: 1. Product selection 2. Report generation

Figure 3. Function point counting procedure for a particular department

The list of the components and appropriate tables for constructing function points are shown in Table 3. Twenty percent are added to all the obtained function points adjusting for any missed components due to unspecified requirements. The unadjusted function points for input modules in COCOMO II are 102, 101, 182, 151 for processing, manufacturing, accounting, and managerial departments respectively.

Table 3. Constructing Function Points

Internal Logical File

Logical File	Data Element Types	Record Element Types	Complexity
1	12	3	Low

External Queries

External Query	File Types Referenced	Data Element Types	Complexity
1	< 2	12	Low
2	< 2	12	Low

External Interface Files

Logical File	Data Element Types	Record Element Types	Complexity
1	9	3	Low
2	5	3	Low
3	10 - 20	2	Low

External Outputs

External Outputs	File Types Referenced	Data Element Types	Complexity
1	< 2	6 - 19	Low
2	< 2	6 - 19	Low
3	< 2	6 - 19	Low
4	< 2	6 - 19	Low
5	< 2	6 - 19	Low
6	< 2	1 - 5	Low
7	< 2	1 - 5	Low
8	< 2	1 - 5	Low
9	< 2	1 - 5	Low
10	< 2	1 - 5	Low
11	< 2	1 - 5	Low

External Inputs

External Inputs	File Types Referenced	Data Element Types	Complexity
1	< 2	5 - 15	Low
2	< 2	1 - 4	Low
3	< 2	1 - 4	Low
4	< 2	5 - 15	Low

Unadjusted Function Points

	Low	Average	High
EI	(4) X 3	X 4	X 6
EO	(11) X 4	X 5	X 7
ILF	(1) X 7	X 10	X 15
EIF	(3) X 5	X 7	X 10
EQ	(2) X 3	X 4	X 6

Total Unadjusted Function Points: 84

Input for COCOMO II: 84 * 1.2 = 101 (20% more due to other uncounted ones that might rise when more thorough requirements review can be completed)

COCOMO II post-architecture is run three times with the 4 modules representing these departments for Visual Basic 5, Report Generator, and Query type of Software. COCOMO II generated total of four phases for each software development cycle. They are plan & requirement, product design, programming, and integration and test. Cost per person month is estimated at $8000. The project schedule per phase, effort per phase, cost per phase, effort per module, and cost per module for all three languages are generated. They are summarized in the Tables 4-7. Table 4 lists effort (in person month), schedule (in months), and cost per phase (in dollars) for all three languages tools. The query software shows lowest amount of effort (15.5 person month), schedule (10.5 months), and total cost ($123,366). The report generator is the highest at 132.6 person month, 22.1 months, and a cost of $1,060,587. However, in this case, the report generator that is included in COCOMOII calibration models probably would not be comparable to something like Crystal Report. The Visual Basic estimates, would give better representation of the current software available technology. For all three languages, programming phase has the largest percentage of four phases in terms of effort, schedule, and cost.

Table 5 through 6 shows the estimates for three different software tools per module versus per phase. The accounting department has accounted for the largest percentage of development effort for all three languages tools. It would take 14.42, 8.9, and 47.97 person months for developing the monitoring services in accounting department using Visual Basic, Query Software, and Report Generator respectively.

Table 4. Effort, Schedule, Cost Per Phase for Three Different Software Tools

Phases	Visual Basic (14,587 SLOC)			Query Software (6,539 SLOC)			Report Generator (40,240 SLOC)		
	Effort	Schedule	Cost	Effort	Schedule	Cost	Effort	Schedule	Cost
Plan & Requirement	2.6	2.3	$20,868	1.0	1.6	$8,071	8.7	3.7	$69,384
Product Design	6.3	3.1	$50,679	2.5	2.2	$19,600	21.1	4.8	$168,504
Programming	22.4	6.3	$179,395	8.9	4.7	$71,173	71.5	8.8	$572,345
Integration and test	8.5	2.9	$68,040	3.1	2.0	$24,523	31.3	4.8	$250,353
Total	**39.8**	**14.6**	**$318,983**	**15.5**	**10.5**	**$123,366**	**132.6**	**22.1**	**$1,060,587**

* Effort is in terms of person month, schedule in months

Table 5. Estimates per Module for Visual Basic

Phases	Visual Basic (14,857 SLOC)			
	Processing	Manufacturing	Accounting	Management
Plan & Requirement	0.53	0.52	0.94	0.61
Product Design	1.28	1.27	2.29	1.49
Programming	4.54	4.5	8.11	5.26
Integration and test	1.72	1.71	3.08	2.00
Total	**8.07**	**8.0**	**14.42**	**9.36**
% Total Cost	**20.3%**	**20.1%**	**36.2%**	**23.5%**

Table 6. Estimates per Module for Query Software

Phases	Query Software (6,539 SLOC)			
	Processing	Manufacturing	Accounting	Management
Plan & Requirement	0.2	0.2	1.8	0.62
Product Design	0.5	0.5	1.79	0.62
Programming	4.54	0.89	3.22	1.11
Integration and test	1.72	0.57	2.09	0.72
Total	**6.96**	**2.16**	**8.9**	**3.07**
% Total Cost	**20.3%**	**20.1%**	**36.2%**	**23.5%**

Table 7. Estimates per Module for Report Generator

Phases	Report Generator (40,240 SLOC)			
	Processing	Manufacturing	Accounting	Management
Plan & Requirement	1.75	1.74	3.14	2.03
Product Design	4.27	4.23	7.62	4.94
Programming	14.51	14.37	25.89	16.78
Integration and test	6.35	6.28	11.32	7.34
Total	**26.88**	**26.62**	**47.97**	**31.09**
% Total Cost	**20.3%**	**20.1%**	**36.2%**	**23.5%**

CONCLUSION

The estimates are established for 4 identified departments with different software tools. The realistic estimate would require 39.8 total person month, 14.6 months, and at a cost of $318,983 to implement the monitoring system for these departments utilizing the central Oracle database. These figures are preliminary, further requirement analysis, and

team interview could provide even more accurate estimates and provide a workable project scope.

REFERENCES

[1] B. Boehm (1981). *Software Engineering Economics.* Prentice Hall, Englewood Cliffs, N.J.

[2] B. Boehem, C. Abts, A. Winsor Brown, S. Chulani, B.K. Clark, E. Horowitz, R. Madachy, D. Reifer, B. Steece (2001) *Software Cost Estimation with COCOMO II*, Upper Saddle River, N.J.

[3] C. Behrens (1983). *Measuring the Productivity of Computer Systems Development Activities with Function Points.* IEEE Transactions on Software Engineering.

[4] J. Kunkler (1985). *A Cooperative Industry Study on Software Development/Maintenance Productivity.* Xerox Corporation, Xerox Square-XRX2 52A, Rochester, NY 14644, Third Report.

[5] *Function Point Counting Practices: Manual Release 4.0*, International Function Point Users' Group, Blendonview Office Park, 5008-28 Pine Creek Drive, Westerville, OH 43081-4899.

BIOGRAPHY OF AUTHORS

Huanzhong Gu is currently a Research Specialist in the Agri&BioSystems Engineering Department, North Dakota State University, Fargo. He is currently working on his MS in Computer Science in NDSU. His research interests include image-processing, software engineering.

Jingpeng Tang received his Ph.D. in Engineering from North Dakota State University in 2002. He is a Research Associate in the Department of Computer Science and Operation Research at North Dakota State University at Fargo. He is currently doing his PhD in Computer Science under Dr. Kendall Nygard. His current research interests include; Networks, Artificial Intelligence, Bio Informatics. He is a member of the ACM.

Vijayakumar Shanmugasundaram is currently working as an Instructor in the Mathematics and Computer Science Department of Concordia College, Moorhead, Minnesota. He received double MS in Engineering and Computer Science from North Dakota State

University. He is working on his PhD program in Computer Science under Dr. Paul Juell in North Dakota State University. His research interests include program visualization in teaching, scientific visualization, network and Web based learning. He is a member of the ACM, an affiliate of IEEE, Computer Society, member of ISCA, member of CUR.