Empirical Study Evaluating Component Reuse Metrics

Pentti Virtanen

Abstract

A new method for estimating the software development effort, Component Reuse Metrics, CRM, combines the best means of component technology and the analysis of human behaviour into simple calculation rules. This study presents the results of the survey of the tenets of CRM and the field experiments about its use. The survey was sent to all of the project managers who were members of the Finnish Information Processing Association. These experts, who had a long experience of the effort estimation, assessed the importance of the project and the human factors of software development establishing the default values of the quantitative parameters of CRM.

1. Introduction

Component Reuse Metrics, CRM, is a new effort estimation method, which considers software development as a series of tasks of assembling software components [1]. CRM adds assessments of project and human effects of the development project to the component-based effort estimates. The approach of this study is empirical. The main result of this paper is an evaluation of the CRM method by a survey and by case studies.

Chapter 2 gives an overview of CRM and its calculation rules. CRM uses a questionnaire to assess the project and human effects of the project, which it is estimating. The main part of the survey of this study is to assess the importance of the questions of the CRM questionnaires. The numerical value describing the importance of a question gives the default weight of the factor of the effect for CRM calculations. Chapter 3 describes the realisation of the survey. The results of the survey are in the following chapters and in the appendixes, which contain the survey data in table form. The case studies, which are described in chapter 7, focus on the practical issues of the usage of CRM. The main question is the existence of a component structure in different kinds of software.

2. Overview of CRM

A software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. It can be deployed independently and it is subject to composition by third parties [2]. Component Reuse Metrics is based on the counts of the components of the product of the project. There are several kinds of components and the granularity of these components can vary [10]. In practical component-based development user interface components are best known. In CRM historical data about tasks of reusing components is collected. In addition to effort information this data contains assessments of the influence of project and human effects to the effort. The estimation process of CRM resembles the estimation process of PROBE because it uses components as proxies of effort [3]. The first step is to define the component structure of the product to be developed. The history data gives the effort of the tasks in the baseline situation. In the baseline case a highly skilled and motivated person without a need for teamwork or any risk accomplishes the task without additional features or process changes. Finally the project will be manned and project and human effects will be assessed and added to the calculations. In project tracking these effects will be reassessed before the data is stored in the history database.

The estimated effort of a project is

(1) $\mathbf{P = \sum I}$, where

- P = estimated effort of the new project, where the efforts of all tasks of all of the persons of the project have been summed and
- I= personal share of a person of the effort of a task of the project.

The personal share of the effort of a task is

(2) $I= cc*ci*s*m*f*t*r*\sum_i(nc_i *p*Ec_i)$, where

- I= personal share of a person of the effort of the task,
- cc= the share of the task of the total effort of the component,
- ci= personal share of the person of the total effort of the task,
- s= personal skill effect coefficient of the person in the task,
- m= personal motivation effect coefficient of the person in the task,
- f= project change effect coefficient of the task,
- t= team effect coefficient of the task,
- r= risk effect coefficient of the task,
- nc_i= number of components of type i in the task,
- p= coefficient for process effect and
- Ec_i= effort of the component of type i from the project history.

The value range of the shares cc and ci is from 0 to 1 and they are normally expressed in percentages. The value of the effect coefficients s, m, f, t, r and p is 1.00 if the baseline effort can be used.

CRM does not have a unique size metric because software products are composed of non-unique parts and the proportions of different kinds of parts vary widely. It counts the components and calculates the effort of each type of component separately. The productivity can be assessed in a component type level, which is an improvement compared to task based estimation. Productivity in producing an average component is useful if the composition of the products remains the same. By the analogy with function-points, the average component could be called component-point.

Each effect coefficient is estimated by a questionnaire, which analyses a group of affecting factors (appendix A). The answers are scaled to a small number (1 to 5) and weighted by a number, which corresponds to the importance of the factor. The importance of the effect is

(3) $imp = (\sum ans*w) /(\sum w)$, where

- imp = importance of the effect,
- ans = answer of the question (scaled from 1 to 5) and
- w= weight of the question given by the results of this study.

The initial values of the weights are calculated from the results of the survey by scaling the assessment of importance from 1 (very small) to 5 (very large) and counting the average (appendix A). The least squares method can calculate the coefficients for each effect when histories of actual efforts and weighted averages of the answers of the questionnaires are available. In equation form

(4) $corr = a * imp + b$, where

- corr= any of the effect coefficients (s, m, f, t, r or p),
- imp = importance of the effect and
- a and b = coefficients of the equation.

3. Survey

The addressees of all 955 project managers and system managers were picked from the member database of Finnish Information Processing Association. In May 2000 516 letters were sent and in June 434 letters. 5 foreign addresses were ignored. Totally 70 replies were received, three of which were empty. The percentage of the replies was 7,1 % (67 replies out of 950), which was a little better than expected. Because the number of replies was larger than 50, the results of individual questions are statistically significant. However, comparisons of the answers of different questions must be omitted and conclusions must be careful. The questions that could not pass the Chi-square test with at least 98% of confidence have been omitted, because the answers were distributed so evenly that the zero-hypothesis of randomness could be true.

A phone survey was done in June in order to evaluate the differences between the actual replies and the replies of people who did not respond. 30 randomly selected people were called. One of them had changed his job and had not received the survey, 14 of them would have responded like the actual respondents and 15 did not reply because estimation of software development projects does not (currently) belong to their job. The phone survey confirms that the results are representative of the population of the project managers.

The experience of estimation of software development projects of the 67 respondents was on average 14,6 years (confidence interval 1,5 years). Eight of them were interested in taking part in a small case study and the results of 6 case studies are presented in this paper and two of them gave up due to schedule reasons.

4. Background information of the respondents

In order to obtain some background information from the respondents, a few questions were asked about the current practices of software development and estimation.

Analysis, design, testing and delivery were often done in more than one phase. 55% of the respondents used iterative models always or mostly and correspondingly 41% of them used waterfall models. This result shows that it is important that estimation methods are suitable for iterative processes. 40 % of the respondents used prototyping for finding the customer requirements always or mostly and 31% similarly for technical reasons. The usage of design methods is low. 29% of the respondents used data flow analysis always or mostly and only 19% used object-oriented methods always or mostly.

The usage of component technologies is low. Only 7 % of the respondents used ActiveX always or mostly. The corresponding figures for DCOM and CORBA were 3% and 5%. Self-made module (37% always or mostly) and class libraries (34% always or mostly) are the most important reuse methods.

Task based estimation is the most common method and numbers of windows, reports and database tables are the most common metrics (appendix C). Function-point method (10% mostly or always) and COCOMO are not used generally. Estimation was seen important and it is often used in adapting the scope of the project to the available budget. Project history was seen useful but its use is not robust.

5. Results of the survey

The survey was used to assess whether the effort of assembling a component is really constant. In the example of the survey a business graphics component (like Microsoft Chart Control) was assessed. The distribution of the effort due to the reuse case was estimated to be less than 10% in 17% of cases, 10%-25% in 26% of the cases, 25%-50% in 30% of the cases, 50%-100% in 17% of the cases and more than 100% in 9% of the cases.

The respondents of the survey saw that the distribution of the effort of a typical software development project was: planning and implementation of the components 35 %, added features 21 %, training for the task 11 %, added effort due to low motivation 10 %, team work 13 % and risks 11%. The default values of the coefficients s, m, f, t and r of the equation (2) can be calculated from this result (1.30, 1.29, 1.60, 1.38 and 1.32 respectively).

The process effect in CRM is used to calculate the influence of the process changes to the effort. In the easiest form methods and tools are the same in the new project as in projects in the history database. Table 1 shows the factors of the process effect. The result implies that there is no single tool or process change, which could dramatically improve the productivity of software development, which requires many co-ordinated methods and tools. Testing, development environment and quality were seen the most influential parts of the development process.

The survey about the amounts of added and removed features showed that adding new features is common. The amount of added effort (answered as always or mostly) is less than 10% in 33% of the cases, 10-20% in 43% of the cases, 20-50% in 22% of the cases and more than 50% in 1% of the cases. The amount of cancelled effort (answered as always or mostly) is less than 10% in 46% of the cases, 10-20% in 15% of the cases, and more than 20% in 0% of the cases.

Table 2 shows the factors of project change effect. The most significant factors are requirement errors, inaccurate analysis and end-users' views. The case studies confirmed that the estimator has a view of possible areas of new features, because the scope of the project is not unambiguous at the beginning of the project. The end user organisation normally pays for these features and that can often be done without renegotiating the contract.

Teamwork introduces communication to the project. It includes meetings, project management and continuous interpersonal communication. Because software development cannot be partitioned into even approximately separate independent tasks, a large amount of communication is needed [9].

Table 3 shows the factors needed for estimating the team effect coefficient. Relative importance of the factors depends on the project. The numbers of the developers and users and the organisation add another view to the factors of the team effect. One of the case studies revealed that travel time is significant in international projects though it is a minor factor on the average. The case study number 3 noticed the effect of synergy, which would decrease the effort because it increases the productivity in teamwork.

Risks are a major part of software development. Table 4 shows the factors needed for estimating the risk effect coefficient. The respondents thought that personnel risks are the most significant in their projects. The risk analysis and common risk factors are well known [6], but their relative importance depends on the project.

In order to be able to produce the desired software, the developers must understand the problem and its solution and express the solutions using the available programming environment. Software development requires many types of skill [5]. The assessment of learning time is in principle task and developer specific. There are two kinds of learning time: the time used to participate in training and time used within the productive work. Learning within work is a good method to learn and we need to estimate the effort used to it. Table 5 shows the survey results about skill factors. In addition to personality, the matching of the experience with the tasks, was seen important.

The respondents estimated that the additional effort of an expert for learning during the assembly of the example component is less than 10% in 42% of the cases (answered always or mostly), 10%-25% in 39% of the cases, 25%-50% in 13% of the cases and more than 50% in 4% of the cases. The same effort for a normal developer is less than 25% in 16% of the cases, 25%-50% in 34% of the cases, 50%-100% in 39% of the cases and more than 100% in 9% of the cases. The additional effort for a novice is much larger: less than 50% in 9% of the cases, 50%-100% in 33% of the cases, 100%-200% in 43% of the cases and more than 200% in 16% of the cases.

There are many references to motivation of the software developers. The most important motivation factors of programmers and analysts are achievement, possibility of growth, work itself, personal life and technical supervision opportunity [6],[7]. Management practices are also important. A wide survey is given by Humphrey [8].

Table 6 shows the results of the survey about the motivation factors of the team members. The Pearson correlation between these replies and replies concerning the motivation of the respondents themselves is from 0,79 to 1,0 excluding the questions of possibility for initiative and independence (0,61) and responsibility (0,67) showing that it is acceptable to use only one set of motivation factor weights in the calculations.

6. Practical issues

In its most accurate form CRM estimation process requires a considerable amount of calculations. There exists a Microsoft Excel-sheet for CRM calculations. Summary level assessments will decrease the effort needed for assessments.

The lack of historical data can be compensated by manual estimates. However, collecting classified data is necessary for increasing the quality of the estimates. Table 7 shows that the respondents of the survey trusted in the possibility to estimate the effects needed in CRM calculations especially after the project (appendix B). The respondents of the survey considered the amount of estimation work to be reasonable if these estimates are made on (sub)project and on personal level and more exactly only when necessary. The case studies confirmed that assessment.

It is possible to create useful project histories because estimating and tracking the effects of CRM separately after the project is feasible. Tracking of the co-ordination work has been found successful also in another study [4]. The case studies showed that the assessment of the project manager is needed in addition to the questionnaires.

7. Case studies

This paper reports six case studies, which contained a half-day interview and a trial of CRM in selected corporations.

The problem of component structure was studied further in the case studies. These revealed that it is difficult to define the components of software, which is not component-based. It was easy to estimate a small application, which contains clear user interface components. Component counting was also feasible in a large application, where the parts of the application can be handled as components. The boundaries of the components in traditional applications and HTML-applications were obscure and there were no reused components, which could be used as proxies of the development effort.

In case 1 the effort of a small mobile application was estimated in two hours. Components were easy to find because they were common user interface components included in the development environment. The effort of a component was easy to assess because the project manager had experience of similar components in PC environment. The assessment of project and human effects looked easy, but their actual influence was surprising. The first implementation of the project was discarded after 585 hours. The programmer was seen incompetent for the task. The estimate was 184 hours and the actual effort of the second, successful, implementation was 204 hours.

In case 2 the effort of a large mostly traditional application was recalculated successfully within the session. The parts of the application were seen as components, because they are logically identical and they have been good proxies of the effort in previous projects.

In case 3 the effort of a HTML application was seen difficult to estimate using CRM, because the structure of the software was unclear. There were large differences in the effort of using a same user interface component in various places because large amount of hand coding was needed in each case.

In cases 4,5 and 6 there were no on-going projects to assess within the session. The problems of component definition were addressed and the project managers thought that the problems could be solved in each of the cases.

8. Conclusions

Task based estimation is the most commonly used effort estimation method. The most important metric is the numbers of windows and database tables. CRM can be seen as an extension to that method.

The case studies confirmed that component-based development creates an acceptable component structure for CRM calculations. This is also true in selected traditional applications. The weights of factors of the project and human effects were estimated and there was a confidence of the possibility to assess these effects. The case study, where an actual project was estimated and accomplished showed that CRM is quite accurate if these assessments are accurate. However, the assessment of project and human effects proved to be difficult at least without experience and any historical data.

Because of the importance of the productivity part of the effort estimates, there is a need to focus on more thorough analysis about the factors of productivity. CRM creates a framework for that.

9. References

[1] Virtanen P.: Component Reuse Metrics - Assessing Human Effects in Katarina Maxwell, Rob Kusters, Erik van Veenendaal, Adrian Cowderoy (eds), Proceedings of the combined 11th European Software Control and Metrics Conference and the 3rd SCOPE Conference on Software Product Quality, page(s) 171-179, Shaker Publishing, April 2000.

[2] Szyperski, C., "Component Software - Beyond Object-Oriented Programming", ACM Press, New York, 1998.

[3] Humphrey W. S., A discipline for Software Engineering, Addison-Wesley Publishing, 1995.

[4] Toffolon, C., Dakhli, S.: The Cost of Co-ordination in Software Engineering in Katarina Maxwell, Rob Kusters, Erik van Veenendaal, Adrian Cowderoy (eds), Proceedings of the combined 11th European Software Control and Metrics Conference and the 3rd SCOPE Conference on Software Product Quality, page(s) 171-179, Shaker Publishing, April 2000.

[5] Hohmann L.: Journey of the Software Professional, Prentice-Hall, 1997.

[6] McConnell S.: Rapid Development, Microsoft Press 1996.

[7] Boehm B. W.: Software Engineering Economics, Prentice-Hall, Englewood Cliffs NJ, 1981.

[8] Humphrey W. S., Managing Technical People, Addison-Wesley Publishing, 1997.

[9] Brooks F. P. Jr.: The Mythical Man-Month, Anniversary Edition, Addison-Wesley Publishing Company, 1995.

[10] Herzum,P., Sims,O.: Business Component Factory, Wiley Computer Publishing, 2000.

Appendix A. Questionnaire.

Table 1. The factors of process effect.

Estimate the effect of the following methods and tools to the development effort (%)	Very large	Large	Medium	Small	Very small	Don't know	I don't use	Weight
Programming language	7	31	34	22	3	0	1	4.13
Development environment	10	31	37	10	1	0	9	4.45
Database management tools	9	18	43	22	3	0	4	4.06
Tools for analysis and design	4	28	37	22	4	0	3	4.03
Project management tools	1	13	33	37	10	3	1	3.55
Configuration management tools	3	25	27	25	4	4	10	4.07
Documentation tools	4	16	37	34	7	0	0	3.71
Tools for testing	7	36	28	19	4	1	3	4.22
The method of quality assurance	10	24	43	16	3	0	3	4.20
Special quality requirements	15	36	30	13	3	0	3	4.44
Project management methods	9	30	36	21	3	1	0	4.16
Testing methods	10	45	34	9	1	0	0	4.47
Documentation standards	3	21	37	36	3	0	0	3.79
Partitioning of the project	7	40	27	24	1	0	0	4.22
Finding and assessing components	3	12	34	30	4	6	10	3.88
Assuring of generality	7	27	34	18	3	3	7	4.25

Table 2. The factors of project change effect.

Factors influencing new unexpected features during a project (%)	Always	Mostly	Often	Rarely	Never	Don't know	Weight
Error in requirement specification	6	42	33	15	3	1	3.33
Inaccuracy of analysis	7	43	42	7	0	0	3.51
Inaccuracy of design	3	22	57	16	0	1	3.12
Partition of the project	0	6	31	54	6	3	2.38
Views of the customer's management	0	16	42	34	6	1	2.70
Views of the IT-management	0	6	27	58	6	3	2.34
End user views	9	46	36	4	4	0	3.51
Programmers' views	4	12	43	36	4	0	2.76
Views of the project manager	4	15	40	40	0	0	2.84
Good idea invented during the project	3	24	55	16	0	1	3.14
Technological surprises	3	19	40	34	3	0	2.85
Change control	6	19	24	45	4	1	2.77
Commercial factors	3	12	22	49	10	3	2.46
Inaccurate contract	0	25	33	24	12	6	2.76
Invoicing method (by hours, contract price)	1	7	24	30	30	7	2.15

Table 3. The factors of team effect.

What is the influence of the following factors of team work in your projects(%)?	Very large	Large	Medium	Small	Very small	Don't know	Weight
Meetings included in the project plan	18	36	27	13	3	3	3.54
Ad hoc meetings	21	52	21	3	0	3	3.94
Unnecessary meetings	4	9	25	36	15	10	2.47
Meeting practice (being punctual, preparation, …)	7	46	30	13	0	3	3.49
Travel time	0	1	36	42	16	4	2.23
Slack time due to travelling	0	4	24	46	19	6	2.14
Phone calls and faxes	3	19	30	33	10	4	2.70
Writing email	12	27	34	19	4	3	3.23
Reading email	12	31	36	13	4	3	3.34
Personal supervision of work	18	37	33	7	1	3	3.65
Discussions with the users	28	46	13	7	0	4	4.00
Discussions with customer's management	27	40	24	6	0	3	3.91
Writing minutes and other documents	10	25	42	16	3	3	3.25
Reading minutes and other documents	7	34	33	21	1	3	3.26
Unanimity about the objectives	24	43	18	10	0	4	3.84
Unanimity about the working methods	22	39	22	12	0	4	3.75
Interruptions of work	19	37	27	12	1	3	3.63
Disturbances in the information flow	34	37	16	7	0	4	4.03

Table 4. The factors of risk effect.

A risk is an incident, which has a probability of influencing a project. If a following risk comes true, how large is its expected influence (%)?	Very large	Large	Medium	Small	Very small	Don't know	Weight
Changes in personnel	34	48	16	0	0	1	4.18
Sickness	12	37	34	13	1	1	3.45
Technical disturbances (black outs, equipment failures …)	4	12	30	34	18	1	2.50
Failures in technology	24	39	16	16	3	1	3.65
Unexpectedly difficult software bugs	10	49	24	12	3	1	3.53
Errors due to carelessness	7	25	46	13	4	3	3.18
Sabotage (viruses, hacking …)	13	9	13	24	31	9	2.44
Unpunctuality of the contractor, subcontractor or customer	24	43	22	9	0	1	3.83
Organisational changes	1	30	43	19	4	1	3.05
Failures in subcontracting and purchasing	15	43	18	13	4	6	3.54
Disputes	10	25	25	33	4	1	3.05
Estimation errors	19	46	27	6	0	1	3.80
Economic risks	3	27	39	30	0	1	3.03
Juridical risks	3	6	21	43	21	6	2.22

Table 5. The factors of skill effect.

What is the influence of the following factors of the skill of software development (%)?	Very large	Large	Medium	Small	Very small	Don't know	Weight
Education	12	21	42	21	4	0	3.15
Courses	7	16	52	22	1	0	3.06
Length of experience	15	42	43	0	0	0	3.72
Quality of experience	46	39	15	0	0	0	4.31
Familiarity of the applications area	39	45	16	0	0	0	4.22
Experience of team work	4	27	46	18	3	1	3.12
Familiarity with the program (to be maintained)	43	33	21	1	0	1	4.20
Knowledge of methods and tools	27	51	16	4	0	1	4.02
Personality (intelligence, emotional intelligence, sense of responsibility, diligence)	42	34	21	1	1	0	4.13

Table 6. The factors of motivation effect.

Estimate the influence of the following motivation factors on increasing or decreasing the productivity of software development. Assess the motivation factors of your team(%)	Very large	Large	Medium	Small	Very small	Don't know	Weight
Challenge or lack of it	16	55	22	3	0	3	3.88
Ambition	9	36	40	12	0	3	3.43
Possibility to accomplish something, achieve results	9	43	40	4	0	3	3.58
Possibility for initiative and independence	9	31	48	9	0	3	3.42
Good/bad leadership	33	37	24	1	1	3	4.02
Too large, appropriate, too small pressure	12	43	33	7	0	4	3.63
Possibility for career	12	24	40	19	1	3	3.26
Good/bad relationships especially within the team	27	42	24	1	1	4	3.95
Appreciation, respect	22	37	31	6	0	3	3.78
Salary and benefits	13	31	40	12	0	3	3.48
Responsibility	9	27	46	13	1	3	3.29
Being noticed or lack of it	19	42	30	4	1	3	3.75
Possibility to develop oneself, to learn new things	21	46	27	3	0	3	3.88
Working conditions	10	34	36	15	0	4	3.42
Interesting work, the work itself	18	55	16	4	0	6	3.92

10. Appendix B. Practical issues of CRM

Table 7. The capability of the project manager to estimate the effects.

The project manager can estimate the influence of the effect in a software development project (%)?	Always	Mostly	Often	Rarely	Never	Don't know
Process when new methods and tools have not been tested?	4	24	31	24	6	10
Process when new methods and tools have been tested	4	34	39	15	0	7
Process when also new methods and tools have been used in actual projects	16	40	24	12	0	7
Project change after the project	24	61	10	3	0	1
Teamwork before the start of the project	3	27	49	13	3	4
Teamwork after the end of the project	13	54	22	3	1	6
Risk before the start of the project	3	33	43	15	4	1
Risk after the end of the project	12	61	21	3	0	3
Skill before the start of the project	6	43	37	10	3	0
Skill after the end of the project	24	58	16	0	1	0
Motivation before the start of the project	4	31	42	15	1	6
Motivation after the end of the project	18	51	22	1	1	6
Is the effort needed for estimating reasonable?	7	63	18	9	0	3
When must human effects be estimated for each component separately?	4	16	36	28	1	13

11. Appendix C. Background information

Table 8. Estimation methods and metrics.

I use the following methods and metrics estimating software development effort(%)	Always	Mostly	Often	Rarely	Never	Don't know
No estimate before the project start	1	1	10	19	66	1
Project is adapted to the budget	12	27	22	33	6	0
Task based estimation	25	63	12	0	0	0
Comparison with similar projects	13	49	31	6	0	0
COCOMO	0	0	3	4	46	46
Function-point Analysis	4	6	28	18	33	10
Comparison with project tracking history	6	22	37	27	6	1
Number of subsystems	4	30	22	19	15	9
Number of windows, reports and database tables etc.	12	58	18	7	1	3
Number of classes	4	13	13	27	31	10
Lines of code	0	0	27	19	49	4
Number of customers of the component	1	1	6	18	57	16