Comparing Effort Estimates Based on Use Case Points with Expert Estimates

Bente Anda

Department of Informatics
University of Oslo
P.O. Box 1080 Blindern
NO–0316 Oslo
NORWAY
bentea@ifi.uio.no

IBM Norway AS
P.O. Box 500
NO–1411 Kolbotn
NORWAY

Abstract. Use case models are used in object-oriented analysis for capturing and describing the functional requirements of a system. Attributes of a use case model may therefore serve as measures of the size and complexity of the functionality of a system. Many organizations use a system's use case model in the estimation process.

This paper reports the results from a study conducted to evaluate a method for estimating software development effort based on use cases, the *use case points method*, by comparing it with expert[1] estimates. A system was described by a brief problem statement and a detailed use case model. The use case points method gave an estimate that was closer to the actual effort spent on implementing the system than most estimates made by 37 experienced professional software developers divided into 11 groups (MRE of 0.21 versus MMRE of 0.37).

The results support existing claims that the use case points method may be used successfully in estimating software development effort. They also show that the combination of expert estimates and method based estimates may be particularly beneficial when the estimators lack specific experience with the application domain and the technology to be used.

Keywords. Use Cases, Estimation

1 Introduction

Use case modelling is a popular and widely used technique for capturing and describing the functional requirements of a software system. A use case model has two parts, the use case diagram and the use case descriptions. The diagram provides an overview of actors and use cases. The use case descriptions detail the

[1] The term expert is used in this paper to denote experienced estimators.

requirements. An actor represents a role that the user can play with regard to the system, and a use case represents an interaction between an actor and the system.

The *use case points method* is a method for estimating software development effort based on use cases. This method has shown large potential in two case studies reported in [3,12], but apart from these two studies, the method has not, to the author's knowledge, been subject to evaluation.

In the case study reported in [3], the use case points method was compared with expert estimates and actual effort for three industrial development projects. These projects lasted from 3 to 7 months with a total effort of 3000 to 4000 person hours. The systems constructed were for e-commerce and call-centres within banking and finance. The use case points method gave estimates that were almost as close to actual effort as the estimates produced by very experienced software developers with good knowledge of the application domains and the technology to be used. The use case based estimates were between 4.5% and 30% lower than the actual effort for the projects.

A modified version of the use case points method produced very precise estimates for two industrial development projects with a total effort of 10000 and 14000 person hours, respectively [12]. The first system was an internet application for a bank aimed at facilitating communication with customers. The other system was a real-time application that was part of a large commercial solution.

Those case studies motivated a new study, described in this paper, to investigate how the use case points method perform compared with experts, that is experienced software developers and estimators, with less experience with the application domain and the technology to be used

The overall motivation is that many organizations use a system's use case model in the estimation process. However, to the knowledge of the author, there are no standards for use case based estimation. In [15] it is recommended that method based estimates should be used to improve expert estimates, the results in [5] show that it is sensible to combine methods and human judgement.

This study was conducted as part of three courses on use case modelling in a large international IT company. Two of the courses were held in Denmark, one in Norway. The participants were very experienced software developers and project managers from Denmark, Norway and Sweden. All together there were 37 participants in the three courses, divided into a total of 11 groups. The groups were asked to estimate effort necessary for implementing a software system. A brief problem statement and a detailed use case model described the system. The participants had read through the problem statement and had worked on the use case model for almost one day before estimating the effort. The use case points method was also used to estimate the same software system. The estimates were compared with the actual effort used to implement the system.

The estimation method gave an estimate with Magnitude of Relative Error (MRE) equal to 0.21 which was closer to the actual effort for the project than most of the estimates suggested by the groups of professional developers. The Mean Magnitude of Relative Error (MMRE) for the estimates made by the groups of experts was 0.37.

The results from this study therefore support existing claims that the use case points method may be used successfully in estimating software development effort. They also show that a combination of expert estimates and method based estimates

may be more accurate than expert estimates alone, in particular when the estimators lack specific experience with the application domain and the technology to be used.

The remainder of this paper is organized as follows. Section 2 describes the use case points method. Section 3 describes the study in details. Section 4 presents the results. Section 5 discusses the design of the study. Section 6 concludes and describes how we the work reported in this paper is continued.

2 The Use Case Points Method

The use case points method was initially developed by Gustav Karner [9]. It is based on the function points method [2], and the aim was to provide a simple estimation method adapted to object-oriented projects. This section gives a brief overview of the method as described in [13]. The method requires that it should be possible to count the number of transactions in each use case. A transaction is an event occurring between an actor and the system, the event being performed entirely or not at all.[2] The four steps in the use case points method are as follows:

1. The actors in the use case model are categorized as *simple*, *average* or *complex*. A simple actor represents another system with a defined API; an average actor is another system interacting through a protocol such as TCP/IP; and a complex actor may be a person interacting through a graphical user interface or a web-page. A weighting factor is assigned to each actor category:
   - Simple: Weighting factor 1
   - Average: Weighting factor 2
   - Complex: Weighting factor 3

   The total *unadjusted actor weight* (*UAW*) is calculated by counting the number of actors in each category, multiplying each total by its specified weighting factor, and then adding the products.

2. The use cases are also categorized as *simple*, *average* or *complex*, depending on the number of transactions, including the transactions in alternative flows. Included or extending use cases are not considered. A simple use case has 3 or fewer transactions; an average use case has 4 to 7 transactions; and a complex use case has more than 7 transactions. A weighting factor is assigned to each use case category:
   - Simple: Weighting factor 5
   - Average: Weighting factor 10
   - Complex: Weighting factor 15

   The *unadjusted use case weights (UUCW)* is calculated by counting the number of use cases in each category, multiplying each category of use case with its weight

[2] Appendix A shows a use case from one of the development projects used in this study. The basic flow of events in the use case consists of 6 transactions. The use case is documented according to a template used throughout the company. The template resembles those recommended in [Cockburn, 2000].

and adding the products. The UAW is added to the UUCW to get the *unadjusted use case points* (*UUPC*).

3. The use case points are adjusted based on the values assigned to a number of technical factors (Table 1) and environmental factors (Table 2). These factors are meant to account for effort that is not related to the size of the task.

Table 1. Technical complexity factors

Factor	Description	Wght
T1	Distributed system	2
T2	Response or throughput performance objectives	2
T3	End-user efficiency	1
T4	Complex internal processin	1
T5	Reusable code	1
T6	Easy to install	0.5
T7	Easy to use	0.5
T8	Portable	2
T9	Easy to change	1
T10	Concurrent	1
T11	Includes Security features	1
T12	Provides access for third parties	1
T13	Special user training facilities are required	1

Table 2. Environmental factors

Factor	Description	Wght
F1	Familiar with Rational Unified Process	1.5
F2	Application experience	0.5
F3	Object-oriented experience	1
F4	Lead analyst capability	0.5
F5	Motivation	1
F6	Stable requirements	2
F7	Part-time workers	-1
F8	Difficult programming language	-1

Each factor is assigned a value between 0 and 5 depending on its assumed influence on the project. A rating of 0 means that the factor is irrelevant for this project; 5 means that it is essential.

The TechnicalComplexity Factor (TCF) is calculated by multiplying the value of each factor in Table 1 by its weight and then adding all these numbers to get the sum called the *TFactor*. Finally, the following formula is applied:

$$TCF = 0.6 + (.01*TFactor)$$

The Environmental Factor (EF) is calculated accordingly by multiplying the value of each factor in Table 2 by its weight and adding all the products to get the sum called the *Efactor*. The formula below is applied:

$$EF = 1.4+(-0.03*EFactor)$$

The *adjusted use case points (UCP*) are calculated as follows:

$$UCP = UUCP*TCF*EF$$

4. Karner proposed a factor of 20 person hours per use case point for a project estimate. Schneider and Winters recommend that the environmental factors should determine the number of person hours per use case point [13]. The number of factors in F1 through F6 that are below 3 are counted and added to the number of factors in F7 through F8 that are above 3. If the total is 2 or less, use 20 person hours per UCP; if the total is 3 or 4, use 28 person hours per UCP. If the number exceeds 4, they recommend that changes should be made to the project so the number can be adjusted. Another possibility is to increase the number of person hours to 36 per use case point.

A spreadsheet is used to implement the method and produce an estimate. The method provides an estimate in total number of person hours.

The use case points method can be criticised from a theoretical point of view as has been the function points method. The addition and subsequent multiplication of ordinal values is, for example, theoretically invalid [10]. However, the function points method has been shown to predict effort reasonably well for many types of systems, and the aim of this study is to extend the empirical evaluation of the use case points method.

There are several other methods for use case based estimation. The methods differ in that size and complexity of the use cases are measured differently, and they also consider different technical and environmental factors. The company in which the study was conducted has its own method for use case based estimation. A commercial tool, *Optimize* [16,18], provides estimates when a use case model for the project is available.

Two alternative methods for estimation based on use cases are described in [7,14]. The method described in [7] maps attributes of the use case model into function points. In [14] a certain number of lines of code is assumed for each use case, and the total number of lines of code is used as a basis for the estimate. A metric suite for use case models, which can be used for estimation, is suggested in [11], but a complete estimation method is not presented.

The use case points method was chosen for this study because it had already showed good results in two previous case studies [3,12]. The company specific method is confidential and is therefore not described here.

3 The Study

This section describes the participants taking part in the study, the context, the procedure of the study and the material used.

3.1 Participants

The study was conducted as part of three courses on use case modelling in a large international IT company. The course schedule was the same in all the three courses. The participants were experienced developers, business analysts and project managers. The author was one of the instructors on the last two courses. All the

participants had extensive experience of requirements engineering, and they had experience from estimating their own work. None were previously familiar with estimation based on use cases, but some were familiar with the function points method. Table 3 gives some characteristics of the participants in the three courses.

Table 3. **Description of three courses on use case modelling**

Characteristic	Course 1	Course 2	Course 3
No. of participants	11	14	12
Nationality of participants	Danish, Norwegian and Swedish	Danish	Norwegian
No. of groups	3	4	4
Average experience with software development	10 years	7 years	16 years
Experience with use case modelling	All had attended previous courses. Some had professional experience.	Most had attended previous courses. Some had professional experience.	Some had attended previous courses. None had professional experience.
Experience with similar projects	Some.	Some. Mostly experience from large batch-systems.	Few. Mostly experience from mainframe and maintenance.

3.2 Context

The duration of the courses was two days. The course schedule consisted of presentations on different aspects of use case modelling, one small assignment on the first day and a larger assignment on the second day. At the end of the first day, the participants were given a problem statement and a system context diagram for a system to be constructed. They were asked to read this through to be prepared for the assignment. On the second day, the participants were randomly divided into groups. The groups worked on constructing a use case model for most of the day. At the end of this assignment, the groups were given a complete solution consisting of 5 actors and 22 use cases. They were then given a 30 minutes lecture on estimation in general and on how attributes of a use case model can be used as a measure of the size of the functionality. The use case points method was not presented in this lecture.

3.3 Procedure of the Estimation Task

Based on the use case model with 4 actors and 22 use cases that had been handed out, each group spent 15 minutes on estimating team size, elapsed project time in months and total effort in person months. Each group was asked to estimate the effort that they would have used themselves.

In all the groups the members discussed among themselves to reach a result. Some of the estimates were activity based, that is, made by adding up the expected effort for the activities in the development project more or less independently of the use cases.

Other estimates were produced by the groups attempting variants of use case based estimation ranging from simply counting all the use cases and assuming one person months for each, to classifying the use cases into easy, medium and complex and suggesting a number of weeks effort for each category of use case.

All the groups had an estimate ready after 15 minutes as they were used to following a tight course schedule for almost two days.

3.4 Material

This section gives a brief description of the problem statement that the groups used as a basis for constructing a use case model, the use case model that was handed out before the estimation exercise, and the project that implemented the actual system. Since the course material was taken from an actual project, the company does not permit the presentation of the complete problem statement and use case model.

Problem statement

The description was as follows:

The system will be the IT part of a service for shopping through the Internet and the delivery of products to the customers' homes. The service should be available through a call-centre or the Internet. The system will consist of an order taking facility for both the server and the clients. A list of functional and non-functional requirements to the system was included.

Examples of functional requirements:

1. Register new customer
2. Search for product
3. Make an order

Examples of non-functional requirements:

1. Customer details, especially credit card information, should be protected
2. The solution should be scaleable to handle potentially large increases in use
3. Product details should be kept up-to-date

The system will be implemented using Smalltalk, C++ and Java. The software will be supported on a UNIX machine, and orders will be stored in an Oracle database.

There will be an interface from the order taking facility to a warehouse management system supporting order fulfilment. There will also be an interface from the order taking facility to a product maintenance system and an interface to a bank to provide on-line credit authorisation.

Use case model

The use case model contained 4 actors, 2 primary and 2 secondary, and 22 use cases. Both the actors and the use cases were described using a template as shown in Appendix A. The use cases were described with much detail, and they are similar to the use case models in the case study reported in [3].

Development project
The system was developed over a period of 3 months, and 8 people were involved. The total effort was therefore approximately 24 person months. The development project was similar to the projects described in [3] in terms of length, total effort, number of people involved and also with regards to functionality.

4 Results

This section presents the estimates produced by the groups and the use case points method together with actual effort for the project.

4.1 Estimates

Table 4 shows which course the participants in the study attended and the number of people in each group. The table also shows what each group suggested as team size, elapsed time and total effort for the system to be constructed.

Some of the groups suggested a total effort in hours, days, weeks or years instead of months. The basis for the formula used to convert into person months was the following load: 7 hours pr. day, 5 days pr. week, 4 weeks pr. months and 11 months per year.

Table 4. The estimates from each group

Group information			Suggestions		
Course no.	**Group id.**	**Number of people**	**Team size**	**Elapsed time in months**	**Total effort in person months**
1	11	4	5	6	20
1	12	4	5	8	30
1	13	3	4	4	16
2	21	4	9	4	33
2	22	4	3	4	11
2	23	3	5	3	15
2	24	3	3	5	12.5
3	31	3	6	2.25	11
3	32	3	8	5	35
3	33	3	5	5	22
3	34	3	5	4.2	12.5

The author had exactly the same information about the development project as the groups, that is, the problem statement and the use case model, and spent approx. 10 minutes to produce an estimate using the use case points method. The value for each technical factor was assessed from the description in the problem statement. Detailed characteristics of the actual team that developed the system were unknown, the only information available was that there were no particular problems in terms of skills or motivation. Table 5 therefore has 2 use case points estimates. The first estimate, 29

person months, was produced by assigning the environmental factors F1-F6 the value 3, and the factors F7-F8 the value 2 under the assumption that the team was slightly better than average. The second estimate, 31 person months, was produced by omitting the environmental factors from the estimate.

Table 5 also shows actual effort for the project. The formula described in Section 4.1 was used to convert between person hours and person months. The use case points method does not suggest team size or elapsed time and the actual effort in person hours is not available. Therefore, the three corresponding cells are empty.

Table 5. Method estimate and actual effort

Method	Team size	Elapsed time in months	Effort in person hours		Effort in person months	
Use case points method			4086	4370	29	31
Actual effort	8	3			24	

4.2 Discussion of the Results

The results in Tables 4 and 5 show that both the experts and the use case points method produced reasonably accurate estimates based on the available information. On average the groups of experts estimated a total effort of 19.8 person months, and the mean Magnitude of Relative Error (MMRE) is 0.37. The Magnitude of Relative Error for the use case points estimate is 0.21 or 0.29 depending on how the values for the environmental factors were assigned. However, we must expect inaccuracies in the estimates during the requirements phase. Boehm states that these inaccuracies range up to 60 percent or more [6].

The use case points method produced estimates that were closer to the actual effort than the estimates produced by 8 of the 11 groups of professional software developers. This shows that the estimation method successfully exploited the information in the use case model, that is, the abstraction level of the use case model must have been fairly appropriate. The groups of experts, on the other hand, did not know exactly how to use the information in the use case model. They chose different estimation strategies leading to estimates of various precision. The results therefore support the results in [3,12], which indicate that the use case points method may support expert knowledge in producing accurate estimates. In [3] the estimation method produced estimates that were almost as accurate as the estimates produced by experts. In the study reported in this paper, the estimation methods were more accurate than the groups. The results reported in [17] show that familiarity with the application domain and the technology is important to produce accurate estimates. An explanation why the method performed better in this study may thus be that the participants in this study on average were less experienced with the application domain and the technology than the estimators in [3].

5 Discussion of the Study

The author believes that the fact that the estimates were made by groups instead of individuals contributed to a realistic setting. In the actual organisation, estimates are usually made in groups, and the discussion between the three or four members of each group forced them to explain their individual estimates and correct each other.

One may argue that 15 minutes, the time available for estimation, is short for producing a serious estimate. However, time constraints are common in studies of this kind, see for example the experiments reported in [15]. Note that in our study the participants spent quite a lot of time getting familiar with the requirements for the system to be constructed before the estimation was done.

A challenge when evaluating the precision of an estimation method compared with expert estimates is that it is seldom feasible to have several teams of software developers developing the same system. This was the case also in this study, the estimators were asked to estimate how long they themselves would take to complete the system, but their estimates were compared with the time it took another team.

The actual effort spent on the development project used in this study was 24 person months. The claims for the use case points method made in this paper are therefore based on the assumption that the estimation groups also would have implemented this system in approximately 24 person months. There are, however, some threats to this assumption. For example, several of the groups suggested a team size smaller than the 8 persons who were involved in the actual project. The team size may affect the effort of a project; typically a decrease in team size results in lower total effort. On the other hand, the participants in this study have less experience with this type of project and the technology used than the developers who implemented the project. This aspect makes it likely that they might have spent more time than the actual developers.

The estimate of a software development project often impacts the actual effort [8]. For example, a development team may want to fulfil an estimate if there is a lack of time even if this means implementing less functionality than originally specified. Therefore, if the actual effort for a project is very close to the estimated effort, this may not necessarily mean that the estimate was good [1].

The format and quality of the use case model may impact the estimates. In this study, only one use case model, described using one particular format, was used. We have, however, discussed how the format of the use case model impacts the estimates in a paper reporting three case studies on the use case points method [2]. There are various sets of guidelines on writing use cases, we have reported the results from an experiment evaluating how three different sets of guidelines affect the quality of the resulting use case model in [3].

6 Conclusions and Future Work

A study was conducted in a large international software development company with the aim of investigating how an estimation method based on use cases, the use case points method, performs compared with groups of experienced software developers.

The use case points method produced estimates that were quite close to the actual effort spent on a development project. They were closer to the actual effort than most

of the estimates produced by the groups of professional software developers. The estimation method gave a best estimate with Magnitude of Relative Error (MRE) equal to 0.21, while the Mean Magnitude of Relative Error (MMRE) for the estimates made by the groups of experts was 0.37.

This supports earlier results indicating that when a use case model for a project is available, the use case points method may support expert knowledge. The results also show that the combination of expert estimates and method based estimates may be particularly beneficial when the estimators lack specific experience with the application domain and the technology to be used.

We are now continuing this work by investigating how the use case points method performs on different types of projects, in particular regarding size and level of detail in the use case model, and we are cooperating with several companies in order to evaluate the method on projects from different companies.

In parallel we are pursuing the work on how the use case points method best can be used in combination with expert knowledge. We are currently running a series of interviews with project managers in a particular company. The aim of these interviews are twofold; to determine in what situations they feel the need for the support of an estimation method, and to find which elements of their current, informal estimation process can be formalized and used in combination with the use case points method.

A controlled experiment is planned with professional software developers to assess whether the use case points method should be applied also when a detailed use case model is not available. The aim of the experiment will be to examine to what extent and under what conditions professional software developers can correctly identify the number of use cases and their complexity from a textual requirements specification.

Acknowledgements

Bruce Anderson and Paul Fertig made the course material for the course on use case modelling and made this research possible. Tor Svelle and Arne Kragh gave me the opportunity to teach the course. I thank Rune Markussen for teaching the course with me, and I thank the participants in the three courses. I also thank Magne Jørgensen, Dag Sjøberg and Ray Welland for useful comments on this paper.

References

1. Abdel-Hamid, T.K. and Madnick, S.E. Lessons Learned from Modeling the Dynamics of Software Development. Communication of the ACM, 32(12), December, pp. 1426-1438, 1989.
2. Albrecht, A.J. Measuring Application Development Productivity. Proceedings of Joint SHARE, GUIDE, and IBM Application Development Symposium. 1979.
3. Anda, B., Dreiem, D., Sjøberg, D.I.K., and Jørgensen, M. Estimating Software Development Effort Based on Use Cases - Experiences from Industry. In M. Gogolla, C. Kobryn (Eds.): UML 2001 - The Unified Modeling Language.Modeling Languages, Concepts, and Tools, 4th International Conference, Toronto, Canada, October 1-5, 2001, LNCS 2185, Springer-Verlag, pp. 487-502.

4. Anda, B., Sjøberg, D.I.K. and Jørgensen, M. Quality and Understandability in Use Case Models. In J. Lindskov Knudsen (Ed.): ECOOP 2001 – Object – Oriented Programming. 15th European Conference, Budapest, Hungary, June 18-22, 2001, LNCS 2072, Springer-Verlag, pp. 402-428.
5. Blattberg, R.C. and Hoch, S.J. Database models and managerial intuition: 50% model + 50% manager, *Management Science,* Vol. 36, No. 8, pp. 887-899. 1990.
6. Boehm, B.W. *Software Engineering Economics*. Prentice-Hall. 1981.
7. Fetcke. T., Abran, A. & Nguyen, Tho-Hau. Mapping the OO-Jacobson Approach into Function Point Analysis. Technology of Object-Oriented Languages and Systems, TOOLS-23. IEEE Comput. Soc, Los Alamitos, CA, USA, pp. 192-202. 1998.
8. Jørgensen, M. & Sjøberg, D. Impact of Software Effort Estimation on Software Work. Accepted for publication in Journal of Information and Software Technology. 2001.
9. Karner, G. Metrics for Objectory. Diploma thesis, University of Linköping, Sweden. No. LiTH-IDA-Ex-9344:21. December 1993.
10. Kitchenham, B. *Software metrics: measurement for software process improvement*. Blackwell Publishers. 1996.
11. Marchesi, M. OOA Metrics for the Unified Modeling Language. In Proc. of the Second Euromicro Conference on Software Maintenance and Reengineering, IEEE Comput. Soc, Los Alamitos, CA, USA; pp. 67-73. 1998.
12. Ribu, K. Estimating Object-Oriented Software Projects with Use Cases. Masters Thesis, University of Oslo. November 2001.
13. Schneider, G. & Winters, J. *Applying Use Cases – A Practical Guide*. Addison-Wesley. 1998.
14. Smith, J. The Estimation of Effort Based on Use Cases. Rational Software, White paper. 1999.
15. Stensrud, E. Empirical Studies in Software Engineering Management of Package-Enabled Reengineering Project. Ph.D. thesis, University of Oslo. 2000.
16. The Object Factory. Estimating Software Projects using ObjectMetrix, White paper. April 2000.
17. Vidger, M.R, & Kark, A.W. Software Cost Estimation and Control. National Research Council Canada, Institute for Information Technology. NRC No. 37116. 1994.
18. www.theobjectfactory.com

Appendix A

Use Case id and name	UC 001: Register new customer
Scope & Level	Primary use case for system for shopping through the internet
Goal in context	Register a new customer for access to the system for shopping through the internet.
Preconditions	None
Primary actor	Customer or call-centre operator
Secondary actors	None
Main scenario	1. System prompts for customer post-code 2. Customer supplies post-code 3. System determines that customer is within the delivery area and prompts for customer details 4. Customer supplies name, postal address, telephone number and e-mail address. 5. System stores the details, issues a unique reference and confirms successful registration. 6. Use case ends successfully.
Alternatives	3a. Customer post-code is outside delivery area 3a1. System informs customer that they are outside the eligible area. 3a2. Use case ends in failure. 5a. Customer already registered 5a1. System informs customer that name and address are already registered. 5a2. Use Case ends in failure. 5b. Customer blacklisted 5b1. System informs customer that registration has been rejected. 5b2. Use case ends in failure.
Variations	If e-mail address is not supplied by the customer, a generic call-centre e-mail address is inserted instead as a destination for order exception reports.