2008 15th Asia-Pacific Software Engineering Conference

Automated Support for Software Cost Estimation using Web-CoBRA

Jacky Keung and Ross Jeffery
NICTA Ltd., Sydney, Australia
CSE UNSW, Sydney, Australia
{Jacky.Keung,Ross.Jeffery}@nicta.com.au

Abstract

Software cost estimation is a crucial yet very difficult task for a project manager at the very beginning of a new project. Since software projects are always different in nature, past projects may not necessarily cover all aspects of a new project when used as a basis for cost estimation. The CoBRA hybrid cost estimation technique uses expert knowledge to build a causal model of context-specific cost factors and past project data to predict costs in terms of effort as well as to assess the risks of a project. Further practical advantages of CoBRA are its high level of interpretability and its transparency. While our previous studies have shown that a modified CoBRA called Web-CoBRA produces higher prediction accuracy, the method was not fully adopted by our industry partner because of its complex application steps when it is manually performed.

In this paper, we report on our experiences with further automating Web-CoBRA based software cost estimation for a software company. It supports group decision-making processes by utilizing a wideband Delphi technique. We identify a range of problems when applying Web-CoBRA in the context of a software company and describe the approaches we used to solve these problems in our new tool called EffortWatch.

Furthermore, we report on the evaluation and the effectiveness of EffortWatch using a technology acceptance model (TAM) questionnaire. The result is then compared with a previous study, showing EffortWatch drastically improves the use of the Web-CoBRA technique.

1. Introduction

Quick response to customers and accurate cost estimation are crucial elements in providing competitive bids and remaining competitive in the rapidly growing software market environment. The ability to provide all of these objectives to customers demands capabilities in many areas, which are not easy to achieve with conventional business models. For example, estimating software development effort required for a new project at its early development life cycle with limited information is very difficult to achieve [1]. A large amount of research has occurred to derive early software cost estimates, using methods based on expert opinion [2] and case-based reasoning [3]. Regression based approaches may not be suitable given insufficient project measurement data.

In our previous work we developed Web-CoBRA[1] based on CoBRA [4], a hybrid software cost estimation method that utilizes experts' knowledge on software cost factors and past project data to build the CoBRA cost estimation model. This method was implemented as a set of proof-of-concept tools, each of which supports a different aspect of CoBRA. Initial experiences with using Web-CoBRA in a software company were very promising. Web-CoBRA was considerably more accurate than the company's previous estimation process[1]. However, despite this improvement, the company has not fully adopted the new method due to a number of issues. A major issue found in the previous study was the lack of an integrated system to support required processes. This was resulting in significant amount of time spent on manual re-entry of data throughout the Web-CoBRA process.

In this paper, we have developed a tool called EffortWatch[1] to implement the processes of Web-CoBRA a modified version of the original CoBRA method. The focus here is to present the approaches used in EffortWatch system, and the results of acceptance evaluation assessment of EffortWatch using technology acceptance model (TAM)[28].

We first introduce this work by providing background information to software cost estimation, and discuss issues of our original Web-CoBRA supporting tools in the next section. Section 3 provides an overview of our new EffortWatch[1] tool. Section 4, 5 describes the approaches we used to build EffortWatch in order to automate the model development and application processes of Web-CoBRA. In section 6, we report on the evaluation of EffortWatch in a software company. We also provide further discussion and recommendation on the EffortWatch system. Section 7 concludes the paper.

2. Related Work and Background

Over the last three decades, many software cost estimation methods have been developed with focus on software project prediction accuracy. Extensive study of 104 attributes of 169 software projects by SDC (Scientific Data Center) in 1965 marks the beginning of empirical software cost modelling [5]. In the late 70s, more robust models such as SLIM [6] , COPMO [7] and COCOMO [8] were developed as an attempt to take into account adjustments to nominal estimates made by experts. In the 80s, widely used parametric methods were compared using datasets of various sizes and environments. The main findings showed that parametric methods performed poorly when applied without calibration [7] [9] [10]. In the late 90's, researchers were focused on the expert-based estimation techniques such as subjective effort estimation, and modelling based on expert knowledge elicitation [2], and in many cases found outperformed parametric models. Other newer modelling techniques were also introduced based on machine learning algorithms, such as OSR [12], artificial neural networks [13] [14], CART regression tress [15] and Analogy-based estimation approaches such as ANGEL[3]. In addition, many estimation methods were automated and packaged into commercial tools [11].

Despite enormous measurable improvements, only very few organizations actually use systematic estimation methods [17]. Typical pitfalls with empirical software cost estimation methods were identified as lack of sufficient organizational data, precise size measure and lack of transparency for calibration [4] [1].

[1] ***EffortWatch***™ is a trademark of NICTA Ltd.

1530-1362/08 $25.00 © 2008 IEEE
DOI 10.1109/APSEC.2008.44

2.1 CoBRA

Hybrid methods that incorporate expert knowledge into the model development process seem to be the distillation of thirty years' empirical software cost estimation studies. They have been considered to be "the key in the future developments of software resource estimation field" [18]. CoBRA is one such method, it was introduced in the late 90's by Briand [4]. As a cost estimation package that addresses all the aforementioned pitfalls with empirical software cost estimation methods. The CoBRA method allows practitioners to incorporate any functional size measure and data model into its cost estimation model. Moreover, it utilizes expert judgements to calibrate cost drivers and further models the distributions. The flexibility offered by the CoBRA method has given practitioners a new option to tailor cost estimation models to the context of an organization. The CoBRA method also offers a way to document company project data consistently, elicit expert knowledge and combine both activities to derive cost estimates.

2.2 Web-CoBRA

We have worked with a small software organization, Allette Systems Company, on an implementation of CoBRA in the domain of web application development. This required some adaptation to the specific features of this domain [19], resulting in a modified CoBRA method we call Web-CoBRA.

As reported in [1, 19], the Web-CoBRA method has significantly improved the prediction accuracy of Allette Systems company when compared with the informal prediction method formerly used in Allette Systems. The initial application of the CoBRA method achieved a prediction accuracy of 91%, which is significantly better than using other means of software cost estimation methods. [4]

The Web-CoBRA application process uses a new project specification to derive a probability distribution of effort, instead of a point estimate of effort [4]. The project manager then uses it for project management activities such as cost estimation.

To address the issue of personnel and technology changes, the Web-CoBRA model can be continually updated through model maintenance. For example, company experts may improve their technical knowledge and collaboration between team members. These changes have composite influences on the CoBRA model. As a consequence, some features of the CoBRA model should be modified to reflect these changes. Furthermore, when projects are completed, project managers need to document these projects to keep the CoBRA model up-to-date. As a result of model maintenance, the accuracy of the CoBRA model should be improved overtime.

Supporting tools to facilitate and automate the Web-CoBRA application processes were also developed initially in our Web-CoBRA project, with functionalities to collect, analyse and report results. However these functionalities were loosely coupled and not fully developed. They combined several software applications including:

- A small web application to capture new project data.
- A separate excel spread sheet template for data analysis
- A statistical package plug-in to perform probability distribution calculations using Monte-Carlo simulations.

The Web-CoBRA model also requires a maintenance step to keep it updated. It was reported that the project updating and maintenance was a very tedious job. [20]

Despite the prediction performance of the Web-CoBRA model, inadequate supporting tool has resulted in near zero usage in Allette System. [20]. An obvious drawback is the complexity of building a CoBRA model. In addition, based on the acceptance test using TAM [28] performed by Keung and Jeffery [20], the Web-CoBRA model would have had a higher perceived of usefulness if the model building and the application processes of Web-CoBRA are integrated and fully automated.

3. Overview of EffortWatch™

In response to the issues we discovered with Web-CoBRA we developed a new automated tool, EffortWatch. It is a web-based application, developed to automate the model building and the application processes of the Web-CoBRA method. EffortWatch is a proprietary system incorporating other open source packages such as JFreeChart for visualisations and Java Maths for statistics.

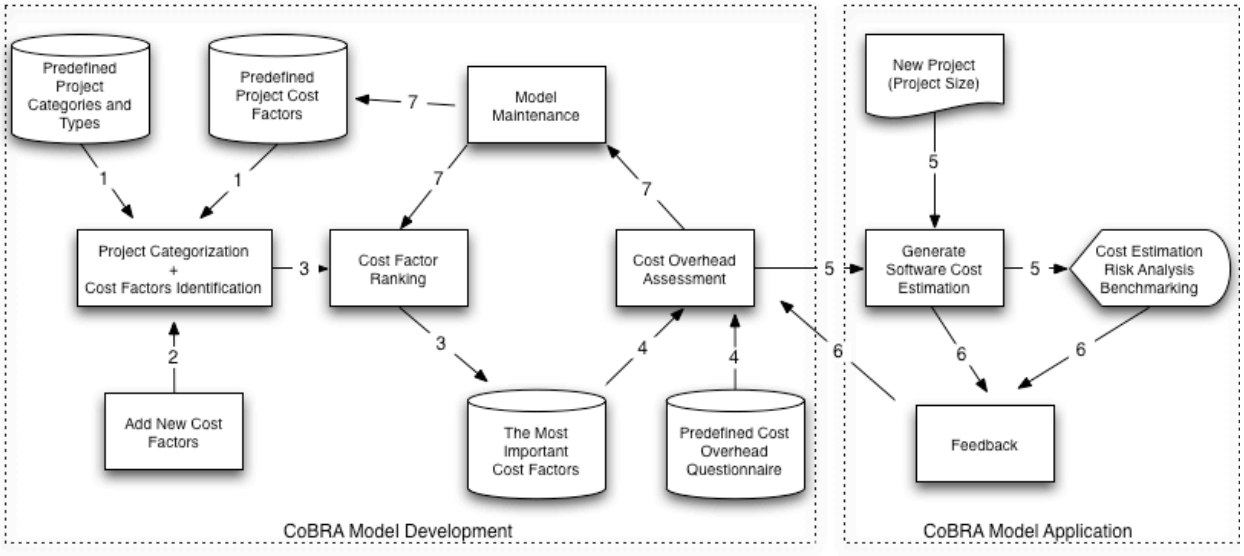

Figure 1 Overview of the EffortWatch Architecture.

Figure 1 provides a general overview of the EffortWatch architecture and its usage. To build a Web-CoBRA model, firstly predefined project categories and types together with predefined project cost factors are retrieved from the database (1). Experts identify additional new cost factors for the model based on their expertise (2). Identified cost factors are then ranked according to their importance. EffortWatch highlights the most important ones for experts. Once agreed by all experts, the most important cost factors are stored in a database (3). EffortWatch generates a project cost overhead questionnaire based on the most important cost factors and a predefined cost factor questionnaire. Experts are required to provide information on the cost overhead values for each cost factor. The Wideband Delphi technique is particularly useful in this case to ensure discrepancies are minimal (4). Once the Web-CoBRA model is generated, the Web-CoBRA model can be used to estimate required development cost, and risk analysis as well as benchmarking. This process is easily accomplished by EffortWatch's built-in automation procedures to perform Monte Carlo simulations and various other computations (5). New project data can be retained (6), and stored back to the database for model maintenance (7) and future use.

In addition, the EffortWatch system offers extra functions to give project managers full control over the system. For instance, project managers have control over the time of the model development process, which allows them to start or end any step based on the company situation. Moreover, building a Web-CoBRA model requires group effort. The EffortWatch system allows project managers to summarize the experts' task progress (i.e. it shows who has completed the current task and who has not). Furthermore, the EffortWatch system provides highly visualised reports and summaries to assist experts to build and apply Web-CoBRA models within the same environment.

4. Expert Knowledge Acquisition

User evaluation of Web-CoBRA identified several enhancements needed for group decision making support [20]. These included software developers' desire for a collaborative support environment. To achieve this we chose to extend the Delphi technique and implement it into EffortWatch.

The Delphi technique [21] is one of the most widely practised expert knowledge acquisition techniques [22]. More recently, the technique has been used as a means of guiding a group of informed individuals to a consensus of opinion on the issue of interest. It helps to eliminate bias in estimates produced by self-proclaimed experts, inexperienced estimators or influential individuals who have hidden agendas or divergent objectives [22]. The original Delphi technique avoided group discussion; the Wideband Delphi technique [8] accommodated group discussion between assessment grounds.

Wideband Delphi elicits expert knowledge through meetings and discussions within the development team. This technique works as follows [22]:

1) The facilitator shows each expert a specification and a form to record estimates.
2) Each expert completes the estimation form independently and discusses questions only with the facilitator
3) The facilitator prepares a summary of all estimates from the experts. Experts then discuss assumptions, estimation issues and questions in a group.
4) Experts converge on a shared set of assumptions and common task list. Experts repeat step 2 to 3 until the estimates are narrowed down to an acceptable range, which has been defined in advance.

The Wideband Delphi technique was widely adopted to derive useful components for software estimation models. For instance, Chulani and Boehm used the Delphi approach to specify the prior information required for the Bayesian calibration of COCOMOII [23]. In the case of Web-CoBRA, expert knowledge can be acquired using this technique to instantiate the CoBRA model.

Before an individual's expert opinion is aggregated and processed it is important to realise that expert opinion values have uncertainties due to a number of reasons. For example, expert knowledge about the parameters which describe the estimate data could be insufficient [24]. Furthermore, experts could be unfamiliar with the subject of the estimation.

In general, experts have difficulties to make a precise point estimate of the cost overhead, for example the influence of one cost factor on effort might vary between different projects. Our approach takes this uncertainty into account by modelling the values provided by experts as uncertain variables and thus requesting the minimal, most likely and maximal cost overhead in percent from each of the experts. Thus, a triangular distribution of the cost overhead is determined. Triangular distribution is the most commonly used distribution for modelling expert opinions. Equation 1 below is the density function of triangular distribution represented by two lines L_1 and L_2, which make up the triangle.

$$f(x|a,b,c) = \begin{cases} L_1 : \dfrac{2(x-a)}{(b-a)(c-a)} & where \quad a \le x \le c \\ L_2 : \dfrac{2(c-x)}{(c-a)(c-b)} & where \quad c \le x \le b \end{cases} \tag{1}$$

where these three parameters (*a, b* and *c*) are Minimum, Most Likely and Maximum values respectively. The range of the distribution indicates how uncertain the expert is about the cost overhead of a specific cost factor. The wider the distribution the more uncertain is the additional cost.

By capturing expert opinion using these three parameters (*a, b* and *c*) and then calculating its triangular distribution, enables the system to convert an expert's qualitative knowledge into quantitative data. The following sections describe how EffortWatch captures expert opinion information to build a CoBRA model for software cost estimation.

4.1 Deriving Potential Cost Factors

Identifying the most important cost factors is an important phase of the Web-CoBRA model development process. As part of a the Web-CoBRA model development process, experts are required to develop at least one cost factor question for the past project questionnaire and one cost factor question for the new project questionnaire for every most important cost factor. Cost factor questions are used to collect information about cost overhead of the most important cost factors in a project.

We use a ranking system to identify the most important cost factors in EffortWatch. Individual experts are then required to rank the importance of each cost factor based on their influence on software effort. Within the same cost factor group, experts can give equal ranking to a cost factor when they consider these cost factors as equally important. When the same rankings were assigned, the rankings of subsequent cost factors would be changed. (See Figure 2)

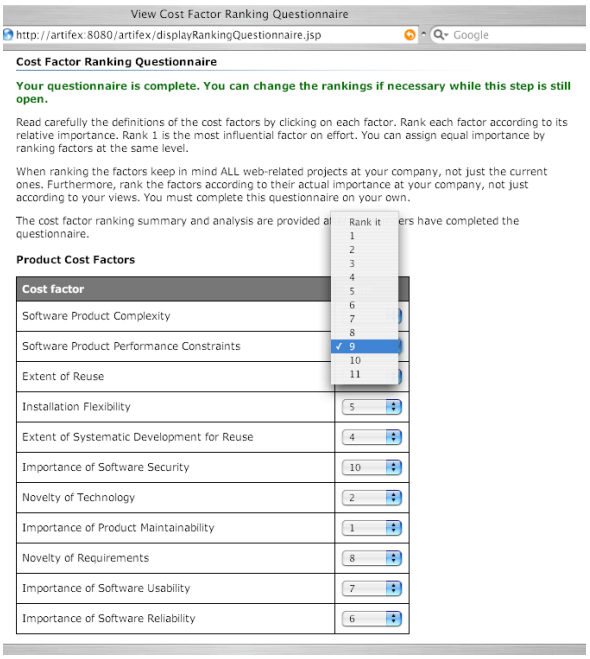

Figure 2 Individual Expert Cost Factor Ranking Collection

Ranking adjustment can be calculated using equation 2 and 3 below, where N is the number of cost factors having the same rank. Rank is the current ranking given these cost factors. For example if there are three factors on "Rank 4" and their adjusted rank equals 5.5 using equation 6, then the next ranked factor is assigned to "rank 6" using equation 3.

$$Adjusted\_Rank = Rank + {}^{(N-1)}\!/_{2} \quad (2)$$

$$Next\_Rank = Rank + N \quad (3)$$

To assist a group of experts to identify the most important cost factors, the EffortWatch system utilizes the Delphi technique to provide collaborative environment and to generate related statistics of cost factor rankings after all experts completed the cost factor ranking exercise. (see Figure 3) The report shows average rank, minimum rank, maximum rank and standard deviation of each cost factor. Cost factors are presented in order by average rank within their cost factor type. The top 25% ranking cost factors in each cost factory category will be automatically selected to form the list of the most important cost factors.

After coming to an agreement about the most important cost factors, the experts are required to identify strong dependencies between the most important cost factors, and remove cost factors irrelevant to the purpose of the estimation exercise. EffortWatch uses a correlation method to ensure cost factors identified are orthogonal. However, it may not always be able to identify irrelevant cost factors. For example, our experts excluded the cost factor "Importance of Software Reliability", because the architecture of web applications is usually highly dependent on the hardware. From the practical experience of the experts, the development effort spent on testing is only a minor part to assure software reliability. It is decisive to choose appropriate web system architecture. Thus, the software reliability is an important cost factor in terms of the customer satisfaction with the developed web application, but not in terms of development effort.

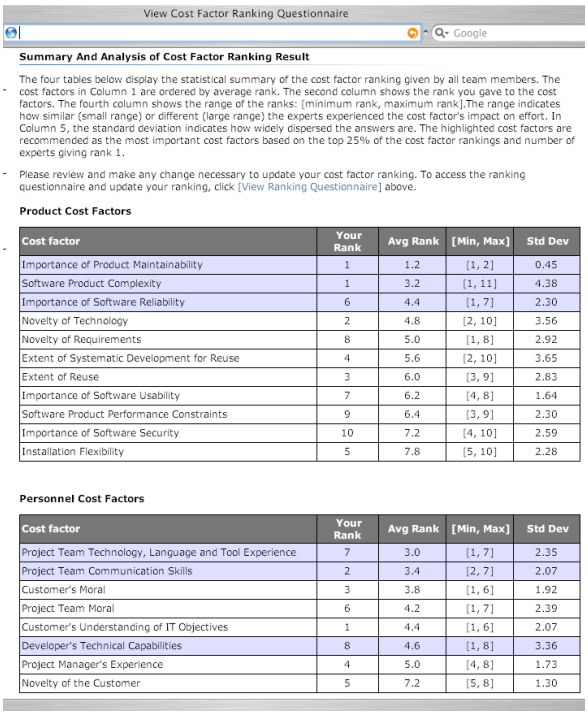

Figure 3 Software Cost Factor Ranking

Figure 3 illustrates the interface of the select cost factor page in which the ranking statistics are displayed. Cost factors considered as most important are highlighted. Experts can further expand the statistics report and select important cost factors by checking the box in the "Important" column.

4.2 Deriving Cost Overhead Values

The cost overhead component is developed to assist users to derive and manage cost overhead values for the most important cost factors. This component is responsible for creating the cost overhead questionnaire. Deriving accurate cost overhead values is essential to the prediction performance of the system in the later Web-CoBRA application process.

The cost overhead questionnaire is generated based on the list of the most important cost factors (see Figure 4). For each important cost factor, experts are required to provide the minimal, most likely and maximal cost overhead values, as inputs to the triangular distribution discussed earlier in this section.

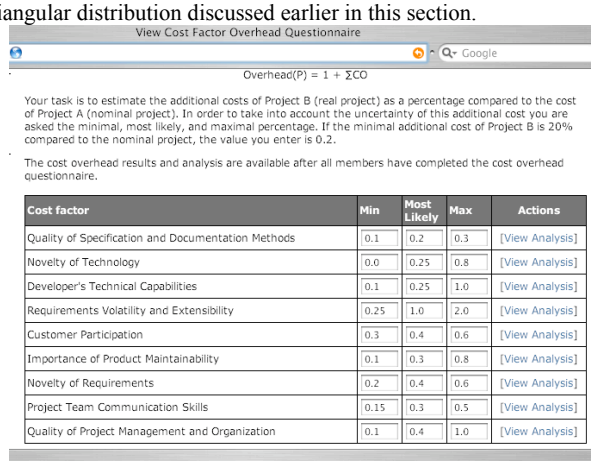

Figure 4 Software cost overhead questionnaire

After all experts completed the cost overhead questionnaire (see Figure 4), the software cost overhead analysis system (see Figure 5) displays statistics related to each cost factor's overhead value, including average, minimum, maximum and standard deviation.

Individual experts can each adjust their input for each cost overhead value until the discrepancies on the values are resolved. The system also helps experts to identify differences in cost overhead values by highlighting any standard deviation whose value is greater than 0.2. The value 0.2 is used for recommendation purposes only, which indicates the given value significantly deviates from a group of data points.

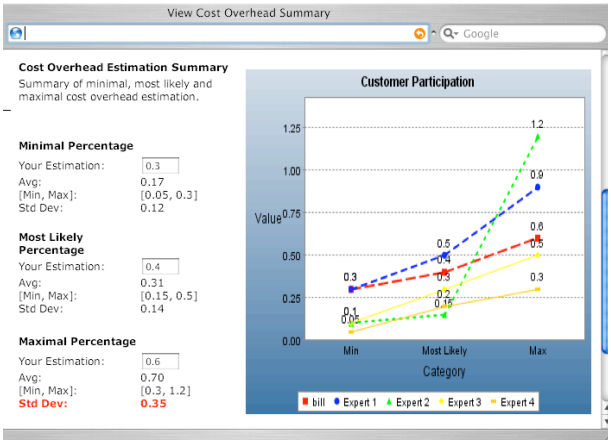

Figure 5 Software cost overhead analysis

5. Software Cost Estimates Generation and Visualisation

In the CoBRA application process, Monte Carlo simulation is used to simulate random or stochastic processes, which describe probability distribution functions [25]. In our previous Web-CoBRA implementation [1], once a Web-CoBRA model is instantiated, the data is then manually transferred to an Excel spreadsheet using a third-party software plug-in to perform Monte Carlo simulation on the dataset [20].

There are usually three major steps for generating values from a distribution function [24]. The following steps have been automated in EffortWatch.

1. Identify the probability density function $f(x)$
2. Identify the distribution function $F(x)$ from $f(x)$ or determine F(x) directly
3. Identify the generating function $G(F(x))$ from $F(x)$. $G(F(x))$ is the inverse function of $F(x)$.

To obtain a random sample for a probability distribution, a random number r is generated between 0 and 1. The r value is then input to the equation $G(r) = x$ to determine the value x. The random number r must be generated from an uniform (0,1) distribution to provide equal opportunity of an x-value being generated in any percentile range [24].

As discussed, the Web-CoBRA method uses a triangular distribution as the Monte Carlo simulation's probability density function (see Equation 1). Thus, the triangular cumulative distribution function can be expressed as (Equation 4) :

$$F(x) = \begin{cases} \dfrac{(x-a)^2}{(b-a)(c-a)} & where \quad a \le x \le c \\ 1 - \dfrac{(b-x)^2}{(b-a)(b-c)} & where \quad c \le x \le b \end{cases} \qquad (4)$$

Furthermore, the generating function $G(F(x))$ from $F(x)$ is derived in equation 5 below:

$$G(r) = \begin{cases} \sqrt{r(b-a)(c-a)} + a & where \quad a \le x \le c \\ b - \sqrt{(1-r)(b-a)(b-c)} & where \quad c \le x \le b \end{cases} \qquad (5)$$

To perform a Monte Carlo simulation, EffortWatch generates a random number r (between 0 and 1). The result obtained represents the answer of one sampling using a triangular distribution as a uniform distribution function. The Monte Carlo simulation can achieve a higher level of precision by increasing the number of permutations in a simulation. EffortWatch uses 1,000 random iterations in the simulation to achieve a realistic minimum for estimating a significant level of about 0.05.

If one constructs a cost estimation model using a particular dataset, and then computes the accuracy of the model using the same dataset, the evaluation will be optimistic [26] . To avoid this, a procedure known as cross-validation approach is applied [27]. It uses different subsets for model building (training sets) and model evaluation (test sets). The following illustrates an example of cross validation. The procedure can easily be automated using a simple loop, where the number of iterations is the number of projects available in the dataset, and it is incremented by 1 project.

The next step in the model building process is to model the relationship between cost and cost overhead values. According to the Web-CoBRA implementation [1], when modelling the relationship between cost and cost overhead in past projects, two options are possible. The first option is to model the linear relationship between productivity and cost overhead. The second option is to model the linear relationship between cost overhead*size and effort. The model that produces better goodness of fit is used to estimate new projects. The EffortWatch system automates the modelling process and outputs the result of a report to the users.

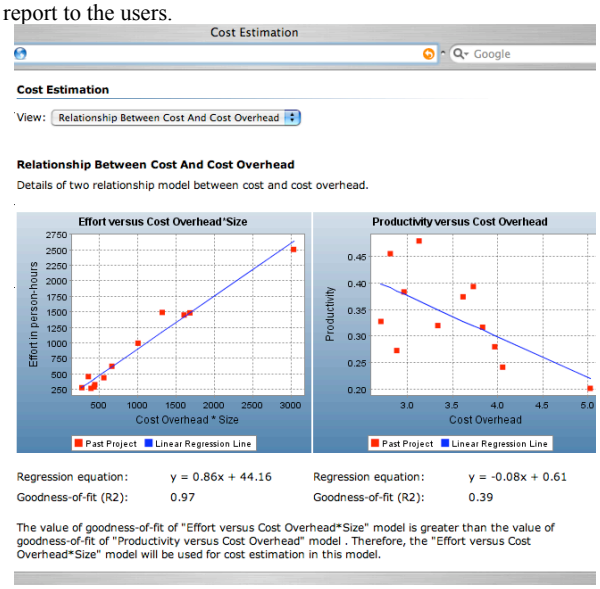

Figure 6 Relationship Between Cost and Cost Overhead

Figure 6 illustrates a report of modelling the relationship between cost and cost overhead. The external Java library JFreeChart is used to provide linear regression charts in the report.

The CoBRA method consisted of the model development process and the model application process. While the previous components discussed in the paper are mainly used to build CoBRA models, the CoBRA application component is developed to assist users to apply CoBRA models when a new project cost estimation is required. There are three major CoBRA applications that the EffortWatch system provides: they are cost estimation, benchmarking and risk analysis. Each application shows different aspects of the estimated efforts of a new project.

The CoBRA application process integrates information from previous parts of a completed CoBRA model to derive a probability distribution of effort of the new project. Therefore, this component contains data access classes to database tables. Furthermore, the application process requires Monte Carlo simulation and several statistical functions. A sub-component is developed in conjunction to provide these mathematical functions to support the CoBRA application process.

Figure 7 illustrates a sample of the effort probability distribution report. It shows that the estimated effort required to develop the target project is 587.69 person hours. The total cost overhead expected is about 243% on top of a nominal project (perfect case). In addition to the numerical presentation of the expected software cost estimates, the effort probability distribution chart is also displayed as a reference for the project manager.

Additional cost estimation information is also provided, such as the dataset used to base estimate, and estimates derived using other cost estimation methods, such as analogy [3]. All of the information are important for the project manager to decide the final cost estimate for the project under investigation.

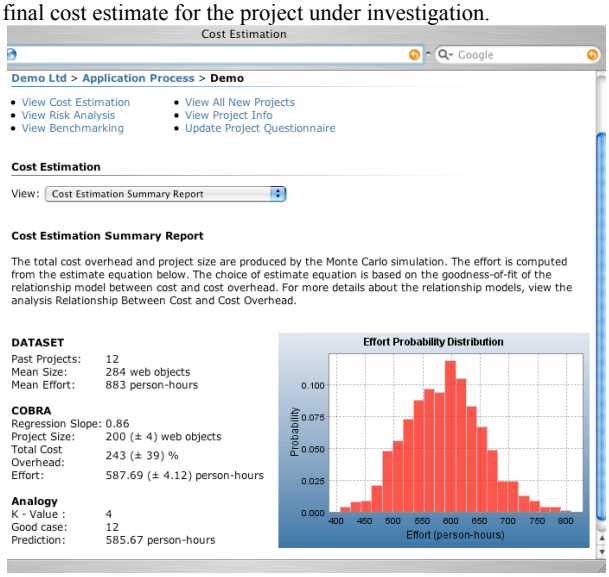

Figure 7 Software cost estimation and effort probability distribution

The benchmarking summary shown in Figure 8 displays details of three thresholds (Minority, Typical and Majority) defined to characterise the cost overhead distribution from the set of past projects. The "Minority" threshold value has 25% of past projects' cost overhead average below it; likewise the "Typical" and "Majority" threshold values, each has, respectively 50% and 75% of past projects' cost overhead average below. Based on the three thresholds and the total cost overhead of the new project, the benchmarking also shows the level of difficulty that the new project belongs to using the following simple algorithm:

```
If (total_cost_overhead <= minority_threshold) {
        the_new_project = low_difficulty_level
} else if (total cost overhead > majority_threshold) {
        the_new_project = high_difficulty_level
} else {
        the_new_project = typical_difficulty_leval
}
```

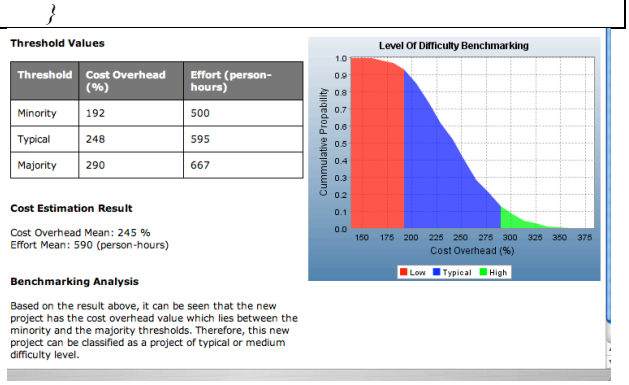

Figure 8 A benchmarking Summary

6. System Evaluation

This study has drawn together different approaches to automate the cost estimation method Web-CoBRA. The new EffortWatch system can easily be used by project managers to build a new instance of the Web-CoBRA model, without memorizing the sequence of all the model development steps involved, and cost estimates with risk analysis can easily be generated. Based on the previous experience, it is important to evaluate the impact of introducing a new technology into the organization, and decide on a course of action in the early stages of technology development. We evaluate the user experience of the new system and provide further discussion of the study.

6.1 Acceptance Evaluation

We have employed a technology acceptance model (TAM) to evaluate the EffortWatch system's user acceptance from the users' point of view. The technology acceptance model (TAM) developed by Davis [28] is a commonly used measure to evaluate user attitude to information technology. It attempts to assess whether users will accept or reject a specific technology. In our case, the TAM questionnaire is used to evaluate the usefulness, the ease of use and the predicted future usage of the EffortWatch system in the context of Allette Systems. In other words, this TAM assessment evaluates how well the new EffortWatch system provides automated support to users of the Web-CoBRA method in their organization.

Four potential users/experts from Allette Systems are selected to participate in this evaluation study. A forty-minute training/presentation was given to the participants. A replicated TAM questionnaire of a previous study carried out by Keung [20] was used to assess user acceptance of this new automated tool. This allows us to compare the new TAM results against the previous TAM results. The questionnaires of four experts were analysed, and results are presented in Table 1.

Table 1 Descriptive Statistics of TAM Survey Result

Question	Minimum	Maximum	Mean	Std.Dev.
Section A: Perceived usefulness				
Q1	6	7	6.75	0.50
Q2	5	7	6.00	0.50
Q3	6	7	6.25	0.50
Q4	6	7	6.75	0.57
Q5	6	7	6.00	0.57
Q6	5	7	5.75	0.95
Section B: Perceived ease of use				
Q7	5	7	6.25	0.50
Q8	4	6	4.75	1.32
Q9	6	7	6.75	0.57
Q10	6	7	6.25	0.50
Q11	6	7	6.25	0.50
Q12	5	7	5.75	0.57
Section C: Self-predicted future usage				
Q13	6	7	6.75	0.50

(See Appendix for the TAM questionnaire used in the study.)

All answers correspond to a value scaled from 1 to 10. The perceived usefulness of the EffortWatch system was derived by aggregating the answers of questions Q1 to Q6. The aggregation is carried out through adding up the corresponding values. Therefore, the perceived usefulness of EffortWatch has a mean value of 37.5 (out of 42), which suggests that EffortWatch is a useful automated support system to facilitate the application of the Web-CoBRA modelling framework.

Similarly, the perceived ease of use of EffortWatch can be obtained through aggregating all the corresponding values rated by participants. The responses show a positive trend (36/42). The result indicates that EffortWatch is considered to be easy to use, which confirms the result we observed previously by conducting a usability test with six fellow students. Question 8 above, which intends to evaluate whether EffortWatch allows uses to manage Web-CoBRA easily is an exception. The slightly negative trend can be attributed to the reason that participants are new to EffortWatch, although they are familiar with Web-CoBRA modelling. Mentally affected by the previous experience with the Web-CoBRA model management, participants would consider managing the Web-CoBRA a difficult task in the environment of EffortWatch. It is believed, in a longer term, participants may find it easy to use the model in the automated environment of EffortWatch.

Question 13 was used to capture self-predicted future usage, which resulted as "extremely likely". It suggests a very positive attitude of the experts towards the EffortWatch system, which has met the users' needs and expectations.

Further evidence toward this conclusion is that, in contrast to the user experience findings of an earlier study [20] where EffortWatch was not used, the ratings of perceived usefulness, perceived ease of use and self-predicted future usage have been improved substantially.

6.2 Lessons Learned and Future Research

It is essential for project managers to have sufficient related software cost estimation knowledge because they need to ensure the quality of the outputs derived from each step during the Web-CoBRA model development process. The accuracy derived by the Web-CoBRA model using EffortWatch is highly depending on whether the project managers have made informed decisions of the following:

- The selection of experts is appropriate
- The project characteristics derived are correct and complete
- All derived cost factors should be independent of each other
- The deviation of cost overhead estimates is reasonable
- Size measure is consistent

The study shows a practical automated software engineering solution to support software cost estimation using CoBRA. Based on the acceptance test results, the EffortWatch system is highly accepted by our software partner.

The first lesson learned is the importance of visualisation, which enables project experts to identify and understand important aspects of their project portfolio and how this information could be applied onto the new project under investigation. The second major lesson is the design of the graphical user interface, which can be used as the guideline throughout the Web-CoBRA model development and its application processes.

Key future enhancements to EffortWatch include:

1. The integration of functional size measure of a software project is likely to improve the overall performance of the Web-CoBRA technique.
2. To provide better interoperability support, currently the system can only import or export data and results using a plain text file or a structured XML file. How EffortWatch would interact with other project management support system within an organization is under investigation.

More case studies in small companies need to be performed to improve the understanding of their work practices. Once we achieve this, we can tailor research activities more towards the generalised needs of small companies. Thereby, we can better contribute and develop software engineering automation solutions to improve software project management in software development organizations.

7. Conclusion

In the previous study, we developed Web-CoBRA to estimate the development cost of web applications in the context of a small software organization. Despite the fact that Web-CoBRA showed a significant improvement in cost estimation accuracy for the company, it was later discovered that it had not been used in practice. Our study shows that an improved automated supporting tool for Web-CoBRA is required.

We have developed and introduced EffortWatch in this study. EffortWatch is an all-in-one system for Web-CoBRA that supports all the required steps involved in the model development and application processes. Effort required in group meetings to derive cost factors and cost overhead values has been reduced to a minimum by using the collaborative decision-making feature presented in EffortWatch, utilizing the Wideband Delphi technique to acquire experts' knowledge. Monte Carlo simulation and other mathematical calculations are handled by the EffortWatch system in parallel with the model development processes of Web-CoBRA. Together with advanced visualisation features, this has greatly reduced the complexity of building and applying a Web-CoBRA model, making this sophisticated method (Web-CoBRA) more acceptable in the software development industry.

A study has been carried out to evaluate the effectiveness of EffortWatch using the technology acceptance model (TAM). Our result indicates that our software partner intending to improve their software engineering practice has successfully adopted EffortWatch.

8. Appendix

The Technology Acceptance Model (TAM) questionnaire used in the study:

Q1. Using EffortWatch in my job would enable me to accomplish tasks more quickly.
Q2. Using EffortWatch would improve my job performance.
Q3. Using EffortWatch in my job would increase my productivity.
Q4. Using EffortWatch would enhance my effectiveness on the job.
Q5. Using EffortWatch would make it easier to do my job.
Q6. I would find EffortWatch useful in my job.
Q7. Learning to operate EffortWatch would be easy for me.
Q8. I would find it easy to get EffortWatch to do what I want it to do.
Q9. My interaction with EffortWatch would be clear and understandable.
Q10. I would find EffortWatch to be flexible to interact with.
Q11. It would be easy for me to become skillful at using EffortWatch.
Q12. I would find EffortWatch easy to use.
Q13. Assuming EffortWatch would be available on my job I predict I will use it on a regular basis in the future.

9. Acknowledgements

The author would like to thank Allette Systems Australia for providing on-going support and contributing to the context for this study, and its staff for participating in the study. Thanks also to Melanie Ruhe [1] for developing and implementing the Web-CoBRA technique, and Shirley Xu developed enhancement to the new EffortWatch.

NICTA (National ICT Australia Ltd.) is funded through the Australian Government's Backing Australia's Ability Initiative, in part through the Australian Research Council.

10. References

[1] Ruhe M., Jeffery R., and Wieczorek I., "Cost Estimation for Web Applications," in *International Conference on Software Engineering*, 2003, pp. 285-294.
[2] Jorgensen M., "A Review of Studies on Expert Estimation of Software Development Effort," *Journal of Systems and Software,* vol. 79, pp. 37-60, 2004.
[3] Shepperd M. and Schofield C., "Estimating Software Project Effort Using Analogies," *IEEE Transactions on Software Engineering,* vol. 23, pp. 736-743, 1997.
[4] Briand L. C., Emam K. E., and Bomarius F., "CoBRA: A Hybrid Method for Software Cost Estimation, Benchmarking and Risk Assessment," in *20th International Conference on Software Engineering*, 1998, pp. 390-399.
[5] Boehm B. and Abts C., "Technical Report: Software Development Cost Estimation Approaches - A Survey," CSE, University of Southern California, LA, CA 2000.
[6] Putnam L. H., "A General Empirical Solution to the Macro Software Sizing and Estimating Problem," *IEEE Transactions on Software Engineering,* vol. 4, pp. 345-361, 1978.
[7] Conte, S., H. Dunsmore and V. Shen, *Software Engineering Metrics and Models*: Benjamin/Cummings, 1986.
[8] Boehm B. W., *Software Engineering Economics*: Prentice Hall, 1981.
[9] Kitchenham B. A. and Taylor N. R., "Software Project Development Cost Estimation," *The Journal of Systems and Software,* vol. 5, pp. 267-278, 1985.
[10] Kemerer C. F., "An Empirical Validation of Software Cost Estimation Models," *Communications of the ACM,* vol. 30, pp. 417-429, 1987.
[11] Stutzke R. D., "Software Estimating Technology: A Survey, Crosstalk," *The Journal of Defence Software Engineering,* vol. 9, pp. 17-22, May 1996.
[12] Briand L. C., Basili V. R., and Thomas W. M., "A Pattern Recognition Approach for Software Engineering Data Analysis," *IEEE Transactions on Software Engineering,* vol. 18, pp. 931-942, Nov. 1992.
[13] Jorgensen M., "Experience with the Accuracy of Software Maintenance Task Effort Prediction Models," *IEEE Transactions on Software Engineering,* vol. 21, pp. 674-681, Aug. 1995.
[14] Finnie G. R., Wittig G. E., and Desharnais J. M., "A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-based Reasoning and Regression Models," *Journal of Systems and Software,* vol. 39, pp. 281-289, Dec. 1997.
[15] Srinivasan K. and Fisher D., "Machine Learning Approaches to Estimating Software Development Effort," *IEEE Transactions on Software Engineering,* vol. 21, 1995.
[16] Kitchenham B., Shepperd M., and Schofield C., "Effort Estimation Using Analogy," in *International Conference on Software Engineering* Berlin, 1996.
[17] Heemstra F. J., "Software Cost Estimation," *Information Software Technology,* vol. 34, pp. 627-639, 1992.
[18] Briand L. C. and Wieczorek I., "Software Resource Estimation," in *Encyclopedia of Software Engineering*. vol. 2 P-Z, J. J. Marciniak, Ed. NY: John Wiley and Sons, 2002, pp. 1160-1196.
[19] Ruhe M., "The early and accurate effort estimation of web applications." vol. diploma thesis Kaiserslautern, Germany: University of Kaiserslautern, 2002.
[20] Keung J. W. and Jeffery R., "The Challenge of Introducing a New Software Cost Estimation Technology into a Small Software Organisation," in *Australian Software Engineering Conference, ASWEC04* Melbourne, Australia, 2004.
[21] Helmer O., *Social Technology*. New York: Basic Books, 1966.
[22] Wiegers K. E., "Stop Promising Miracles," *Software Development,* Feburary 2000.
[23] Chulani S. and Boehm B., "Bayesian Analysis of Empirical Software Engineering Cost Models," *IEEE Transactions on Software Engineering,* vol. 25, 1999.
[24] Vose D., *Quantitative Risk Analysis: A Guide to Monte Carlo Simulation Modeling*: John Wiley & Sons, 1996.
[25] Fisherman G. S., *Monte Carlo: Concepts, Algorithms, and Applications*. New York: Springer Verlag, 1995.
[26] Walkerden F. and Jeffery R., "An Empirical Study of Analogy-based Software Effort Estimation," *Empirical Software Engineering,* vol. 4, pp. 135-158, June 1999.
[27] Hayes W., *Statistics*, Fifth ed.: Hartcout Brace College Pub., 1994.
[28] Davis F. D., "Perceived usefulness, perceived ease of use, and user acceptance of information technology," *MIS Quarterly,* vol. 13, pp. 319-339, 1989.