Using Web Objects for Estimating Software Development Effort for Web Applications

Melanie Ruhe  
SIEMENS AG  
Corporate Technology SE3  
81730 Munich, Germany  
melanie.ruhe@siemens.com

Ross Jeffery  
University of New South Wales  
School of Computer Science and Engineering and CAESER  
Sydney, 2052, Australia  
rossj@cse.unsw.edu.au

Isabella Wieczorek  
Federal Ministry of Education and Research  
53175 Bonn, Germany  
Isabella.Wieczorek@web.de

ABSTRACT
Web development projects are certainly different from traditional software development projects and, hence, require differently tailored measures for accurate effort estimation. In this paper, we investigate the suitability of a newly proposed size measure for web development projects: Web Objects. Web Objects have been specifically developed for sizing web applications and used for estimating effort in a COCOMO II-like estimation model called WEBMO. However, no empirical validation has yet been published. In this study, for the first time, we apply and validate the proposed Web Object approach in the context of a small Australian web development company. Besides Web Objects, we apply traditional Function Points as an effort predictor for web applications. Effort estimation models based on Web Objects are compared with models based on traditional Function Points using ordinary least squares regression (OLS). Tested on data from twelve web applications, the estimates derived from estimation models using Web Objects significantly outperformed models using Function Points, with a Mean Magnitude of Relative Error of 0.24 versus 0.33, respectively. Based on the results of this study, it seems that Web Objects are more suitable for effort estimation purposes of web applications than traditional Function Points.

Keywords
Software Size Measurement, Web Development, Cost Estimation, Web Objects

1. Introduction
Accurate cost estimates are an essential element for providing competitive bids and remaining successful in the market. Over- and underestimates of the expected costs have a significant influence on a company’s reputation and competitiveness. It is widely accepted that among a variety of factors that potentially affect the cost of software projects, software size is one of the key cost drivers [8, 14, 19, 20]. Unfortunately, there is no common way of measuring software size. Software development is an evolving process with often-changing technologies, development methods and tools, as well as often changing requirements.

A new type of software, so-called web applications, has been established in the business world in recent years. With the growing importance of web applications in general business matters, many practitioners see the measurement of web applications as a particularly valuable area of ongoing commercially relevant research. Thus, web development has become the focus of research interest as well. When comparing web development with traditional software development we can identify differences [5]. These differences comprise the software development methods and technologies, as well as the development team and the project conditions. Usually high time pressure and very volatile software requirements play a central role in a web development project. Because of the identified differences, it is necessary to revise and adapt the commonly used software engineering methods and measures before applying them to the application domain of web development.

This study focuses on the size measurement of web applications for the purpose of effort estimation. Appropriate size measures are an important prerequisite for accurate effort estimates. The literature provides examples of traditional size measures such as lines of code or Function Points, as well new size measures, specifically designed for web applications, such as Web Objects or Object-Oriented Function Points [3, 7, 17].

The purpose of this paper is to compare the traditional size measure Function Points with a recently proposed size measure Web Objects. These two measures are applied in the context of a typical web development organisation based in Australia. We use both size measures as effort predictors in ordinary least squares regression models (OLS). In addition, the predictions of the OLS models are compared to the predictions obtained from an informal expert-based estimation method used within the web development organisation. These objectives are reflected in the following research questions:

1. Do Web Objects significantly contribute to more accurate effort predictions for web applications than traditional Function Points?

2. Which of the prediction methods (OLS and expert-based informal method) employed in this study provides more accurate effort predictions?

These issues are investigated using data on twelve web applications collected in a study performed at Allette Systems, based in Sydney, Australia.

In Section 2 the paper starts with a brief discussion of related work in the area of project size measurement in general and sizing of web applications. Section 3 presents the research method including a detailed description of the
data set and the validation method. The results of the field study are presented in Section 4. The summary of the study as well as the discussion of result is contained in Section 5.

2. Related Work

Size measurement has been identified as a key issue for accurate effort estimation models [8, 14, 19, 20]. The following paragraphs discuss the related work with regard to the following three aspects: (1) What are general trends in size measurement? (2) Which size measures have been applied to web applications so far? (3) What research has been performed in the area of software development effort estimation for web applications?

2.1 Measuring Size

Common size measures can be distinguished by the inputs available at the time of measurement. Therefore, we differentiate between problem-oriented and solution-oriented size measures [8]. Solution-oriented measures represent the size of the software development outcome whereas problem-oriented measures represent the size of the problem that the future software product will solve, i.e. the target functionality of the system.

Solution-oriented size measures may reflect the specification, the design, or the implementation of a software system. A very common measure for the size of the implementation is lines of code (LOC). Jones suggests eleven major variations of LOC, e.g. counting executable lines only or executable lines plus data definitions [19]. Furthermore, much discussion concerns the validity of using LOC to measure the size of a system, because of its high dependency on the programming language and style [8].

Problem-oriented measures define elements that can be counted from early lifecycle documents, e.g. state charts, data flow diagrams, or entity-relationship models. The most common approach is to count the so-called Function Points, which is a method that was developed by Albrecht and Gaffney at IBM [17]. They capture the size of a software product in terms of its functional characteristics. A major limitation of Function Points is their restricted applicability being largely applied to Management Information Systems (MIS). To address this shortcoming additional types of functional size measure have been developed: Object-oriented Function Points were proposed for Integrated CASE environments [15]; Feature Points were designed to work equally well with MIS applications, and other types of software such as real-time software or embedded software [19]; Full Function Points (FFP) were proposed as a method for measuring the size of real-time systems [16]. Further development of this work appears in the COSMIC FFP manual [22].

2.2 Measuring Size for Web Applications

The term web application is not unambiguously defined. Web sites and hypermedia systems are usually smaller software development projects and technically different from web applications. Therefore, we firstly have to distinguish web applications from web sites and hypermedia systems [13].

Within this study, a web application is defined as a software application that uses web sites as a front end for broad and remote access. The back end provides full user functionality; the user can affect the status of the business logic on the web server. Web applications consist of different components when compared with traditional software applications, e.g. shopping carts, scripts, and COM components. Typical web applications are financial/trading, business-to-business, and intranet business management applications.

The literature provides examples for both problem- and solution-oriented size measures that can be used for measuring the size of web applications. In a case study on framework-based software development, three different size measures have been applied on five web development projects [7]. In particular these measures are Function Points, Object-oriented Function Points (OOFP), and LOC. Furthermore, the three different measures have been used for building a cost estimation model and the prediction accuracy of the three different estimation models has been compared. The data analysis revealed that Function Points are inappropriate for productivity estimation of framework-based web development because the method does not take into account reuse matters. OOFP and LOC address this matter and are equally suitable for size measurement and cost estimation purposes. OOFP, however, has the advantage of being available earlier in the development cycle. On the other hand, this approach assumes object-oriented development methods, which does not always apply to web development.

Rollo introduced a different sizing approach [18]. In his paper he considered the difficulties of sizing web applications when using the International Function Points User Group (IFPUG) rules. He also applied COSMIC Full Function Points (FFP) for sizing web applications. Rollo claims in his study that FFP’s are the most flexible approach for counting the functional size of web applications. However, he did not provide any empirical result to support his statement in this paper [18].

The COSMIC approach is also applied in a case study comparing size measures and their applicability for web development prediction models [10]. The COSMIC counting rules were slightly changed according to the needs of the study, where no functional requirements documents were available. The counting was based on the final implementation. Another limitation of this study is the type of web system, being characterized as hypermedia systems, which are different from typical web applications. Thus,
there is little empirical evidence in the suitability of the COSMIC FFP measure for web applications as defined in the current study.

Finally, the study from Reifer introduced a size measure specifically designed for web applications called Web Objects [3]. Web Objects extend the well-known Function Point approach and take into account web-specific components. Reifer has applied Web Objects as a size measure and used them in effort prediction models for 64 completed web projects. The analysis found that Web Objects have a better prediction accuracy than traditional Function Points [4]. The main weakness in this study is that Reifer did not publish any detailed empirical results that are needed to prove his claim.

### 2.3 Effort Estimation Models for Web Applications based on Software Size

There are relatively few examples in the literature that address the assessment of cost estimation models for web applications.

Reifer introduced a cost estimation model that adapts the COCOMO II model to the needs of web development, called WebMo [3]. WebMo uses data from 64 web applications on nine cost factors and software size. The COCOMO II cost factors are replaced by new cost factors as well as revised cost factors from COCOMO II [21]. Regression analysis is applied to determine the cost estimate. The model has been calibrated and predicts “within 20% of the actual experience at least 68% of the time”. A competitive accuracy of an estimation model must lie within 20% of the actual experience at least 80% of the time according to Reifer [5]. The prediction accuracy of the remaining 32% of the estimates is not reported at all. Thus, it is not possible to evaluate the prediction accuracy of the model. Furthermore, the proposed model WebMo forces extensive data collection on certain cost factors that might not specifically apply for a certain context.

In a previous study, we used the composite estimation method COBRA™ (Cost Estimation, Benchmarking, and Risk Assessment) to build an effort estimation model for web applications [6]. COBRA combines expert knowledge with data on a small number of projects to develop cost estimation models, which can also be used for risk analysis and benchmarking purposes [1]. The method has been modified and applied for its application in the area of web development. The analysis was based on data on twelve web applications developed at a small Australian company, specialising in web development. The estimates derived from the so-called Web-COBRA model showed a MMRE of 0.17. This result significantly outperformed expert estimates from Allette Systems (MMRE 0.37). In this paper, however, we did not report on the size measure.

Further research has been done in the area of hypermedia systems and web sites, but as mentioned earlier, these projects are different from web applications and therefore not of particular interest of this study [10, 11].

### 3. Research Method

This research is designed as a field study in a single organization. The estimation of web applications is a very new research area with only a few experience so far. Thus, we firstly need to gain a better understanding about the development methods in web development organizations. Field studies are an accepted ideal method for achieving inside information about a new area of interest.

#### 3.1 Data Set

Allette Systems is a small-sized Australian software development company with between twenty and thirty employees. The company’s main business is to develop web-based systems. Due to the diversity of projects within the organization, we first categorised them in order to find a subset of projects that conforms to our definition of a web application.

Project managers of Allette Systems were asked to characterise their software applications in terms of functionality, and other given categorisation criteria. Based on the analysis of the survey responses, we used the projects categorised as web applications for our study. The project managers of this type of projects were briefly interviewed to ensure that the projects followed our definition of a web application (see Section 2.2). Finally, twelve typical web applications were determined as suitable projects.

The considered web applications were developed between 1998 and 2002. The web applications were very typical, e.g. an application for analysing and managing stock or an transactions and several web-based content management systems. Nine applications were new developments, one was an enhancement, and two projects were re-developments.

Table 2 presents the descriptive statistics for the general project information for the twelve web applications.

<table>
<thead>
<tr>
<th>Measure</th>
<th>Min</th>
<th>Max</th>
<th>Mean</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effort - Development effort in person hours (ph)</td>
<td>267</td>
<td>2504</td>
<td>883</td>
<td>710</td>
</tr>
<tr>
<td>Size - System size in Web Objects</td>
<td>67</td>
<td>792</td>
<td>284</td>
<td>227</td>
</tr>
<tr>
<td>Size - System size in Function Points</td>
<td>30</td>
<td>672</td>
<td>242</td>
<td>195</td>
</tr>
<tr>
<td>Peak Staff - Maximum size of the project team at any time</td>
<td>2</td>
<td>6</td>
<td>3</td>
<td>1.5</td>
</tr>
</tbody>
</table>

The average development effort spent on a web application at Allette Systems seems significantly higher than the published effort results on web hypermedia systems.
(between 58 and 154 ph) [10]. Furthermore, the data collected at Allette Systems shows a much higher data dispersion (710 ph) compared to the data on web hypermedia systems (26 ph). The comparison of development effort required confirms the difference in complexity between web applications and web hypermedia systems.

We measured system size using both Function Points and Web Objects [3, 17]. Web Objects are an extension of the traditional Function Points addressing the needs of web development. So far, to our knowledge, no other alternative size measure for web applications is published or initially validated. In general, the number of Function Points is lower than the number of Web Objects. But when comparing the number of Function Points and Web Objects project by project, the size difference between the two measures is up to 55%, i.e. the number of Web Objects is twice as high as the number of Function Points. This sizing difference may have a large effect on the effort estimation accuracy.

The peak staff numbers at Allette Systems are in line with the published numbers for team size in the area of web development, which are usually small [4].

### 3.2 Sizing of Web Applications

#### 3.2.1 Function Points

Function Points were introduced by Albrecht and Gafney as a measure of software functionality that is independent of the programming language [17]. In this study we have used the counting conventions of the IFPUG standard. They define five function types that represent the functional size of a software system. The function types are internal logical files, external interface files, external inputs, external outputs, and external inquiries. The function type counts are weighted depending on their complexity to derive the final FP counts. Function Points can be counted from early lifecycle documents and represent the functionality of the software system from the user’s point of view. In our study the Function Point counts are based on the final product.

#### 3.2.2 Web Objects

Web Objects were proposed as a suitable measure of size for web applications [3]. Reifer adds four new web-related components to the five function types of the Function Point approach: multimedia files, web building blocks, scripts, and links. Figure 1 illustrates the five function types as well as the added web-related components.

The size of a web application is determined by evaluating the nine components of a web system based upon user requirements and page designs. The definitions of the components are attached in the Appendix to this paper. Each instance of the above components needs to be counted and categorized in terms of its complexity, being low, average, or high. A weighting table assigns the corresponding weight to each counted instance of a Web Object component. The sum of these weights represents the functional size of the web application in Web Objects.

Within our study we elaborated Reifer’s work by developing a counting manual to support the systematic and consistent measurement of Web Objects. The manual was completed by company-specific counting examples to facilitate future applications. We performed an interview-based counting procedure based on the final product. For each design element, the corresponding developer counted the total number of Web Objects answering questions with the help of the counting manual.

Figure 2 illustrates the relationship between size in terms of Web Objects and Function Points and effort in person hours. You can see that for larger projects (large in terms of effort) the difference between size measured with Web Objects versus Function Points increases.
3.3 Validation Method

The validation of the size measures’ prediction accuracy is performed as follows:
Each of the two size measures is used as an independent variable to build an effort estimation model using OLS regression.
A cross validation is performed that is a common way to avoid biased results. The effort predictions of the regression models are compared with the expert-based predictions of Allette Systems, which are based on an informal method. The two estimation methods as well as cross validation are explained in the following paragraphs.

3.3.1 Ordinary Least Squares Regression

OLS regression is the most common modelling technique applied to software cost estimation [8]. Regression modelling is one of the most widely used statistical modelling techniques for fitting a response (dependent) variable as a function of (a) predictor (independent) variable(s). With least-squares regression, one has first to specify a model (form of relationship between dependent and independent variables). The least squares regression method then fits the data to the specified model trying to minimize the overall sum of squared errors. This is different from other techniques such as machine algorithms where no model needs to be specified beforehand. In general, a linear regression equation has the following form:

\[
DepVar = a + (b_1 \times IndepVar_1) + ... + (b_n \times IndepVar_n)
\]

Where \(a, b_1, ..., b_n\) are unknown parameters, \(DepVar\) stands for dependent variable and the \(IndepVar's\) are the independent variables. If the relationship is exponential, the natural logarithm is applied to the variables involved and then a linear regression equation can still be used [9].

In our study we used development effort as a dependent variable and size measured in Web Objects as well as in Function Points as an independent variable. As the data was normally distributed we used a linear model specification [9]. It is worth noting that no other cost factors beside size were modelled when applying OLS regression. Using additional factors might have resulted in more accurate effort predictions, but this was not the focus of our study.

3.3.2 Allette Systems Informal Method

Allette Systems applied an informal method to predict development effort for web applications. The method involves the structural decomposition of a system into basic components, which are familiar in terms of estimating effort. The project managers of Allette Systems then compare the basic components with similar work that has been done and estimate the component using personal memory. The final estimate is derived by aggregating the components.

The effort estimate of Allette Systems is documented in their timesheet system. It does not necessarily represent the initial estimate, because additional tasks might be added later on in the project and effort is adjusted accordingly. The system does not mark additional tasks. Thus, the “real” initial estimates are not known for all projects.

3.3.3 Cross Validation

If one constructs a cost estimation model using a particular data set, and then computes the accuracy of the model using the same data set, the accuracy evaluation will be optimistic [8]. To avoid this, we applied a procedure known as cross-validation. Cross validation is a common way of validating a cost estimation model. The basic idea is to use different subsets for model building (training sets) and model evaluation (test sets).

In our case, there are twelve projects in the database and each cross validation step uses eleven projects as a training set to build a new model and apply it to the remaining project (test set). Thus, a separate model is built for each of the twelve training-test-set combinations.

3.4 Evaluation Criteria

The estimates are compared with the actual effort by calculating the magnitude of relative error (MRE). This is a commonly used measure for the evaluation of estimation models [2]. The magnitude of relative error as a percentage of the actual effort for a project, is defined as:

\[
MRE = \frac{Estimate - Actual}{Actual} \times 100
\]

The MRE is calculated for each project in the data sets. Either the mean MRE (MMRE) or median MRE aggregates the multiple observations.

In addition, we used the prediction level Pred. This measure is often used in the literature and is a proportion of observations for a given level of accuracy:

\[
Pred(l) = \frac{k}{N}
\]

Where, \(N\) is the total number of observations, and \(k\) the number of observations with an MRE less than or equal to \(l\). A common value for \(l\) is 0.25, which is used for this study as well. The Pred(0.25) gives the percentage of projects that were predicted with an MRE equal or less than 0.25. Conte et al. [2] suggest an acceptable threshold value for the mean MRE to be less than 0.25 and for Pred (0.25) greater or equal than 0.75. In general, the accuracy of an estimation technique is proportional to the Pred (0.25) and inversely proportional to the MRE and the mean MRE.

We also used box-plots of the MRE to graphically illustrate under- or overestimations for each estimation method.

The estimates obtained from the different methods were compared and tested whether the prediction accuracy of
one method was significantly different from another method using a t-test [8].

4. Results and Discussion

This section discusses the results of the comparison. Firstly, we provide the descriptive statistics of the Function Point as well as Web Object counting. Secondly, the results of the effort prediction using OLS regression are compared.

Table 2. Counting Results - Descriptive Statistics

<table>
<thead>
<tr>
<th>Component</th>
<th>Min</th>
<th>Max</th>
<th>Mean</th>
<th>Std. Dev.</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Input</td>
<td>9</td>
<td>206</td>
<td>51</td>
<td>53</td>
</tr>
<tr>
<td>External Output</td>
<td>16</td>
<td>163</td>
<td>60</td>
<td>44</td>
</tr>
<tr>
<td>External Inquiry</td>
<td>3</td>
<td>191</td>
<td>50</td>
<td>54</td>
</tr>
<tr>
<td>Internal Logic Files</td>
<td>7</td>
<td>12</td>
<td>49</td>
<td>33</td>
</tr>
<tr>
<td>External Interface Files</td>
<td>0</td>
<td>220</td>
<td>33</td>
<td>73</td>
</tr>
<tr>
<td>Function Points (FP)</td>
<td>30</td>
<td>672</td>
<td>242</td>
<td>195</td>
</tr>
<tr>
<td>Multimedia Files</td>
<td>0</td>
<td>70</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>Web Building Blocks</td>
<td>0</td>
<td>72</td>
<td>13</td>
<td>23</td>
</tr>
<tr>
<td>Scripts</td>
<td>0</td>
<td>18</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Links</td>
<td>0</td>
<td>28</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>Web Objects (WO)</td>
<td>67</td>
<td>792</td>
<td>284</td>
<td>227</td>
</tr>
</tbody>
</table>

The Function Point counts show similar numbers for the transactions (external input, output, and inquiry). External Interface Files were used less than Internal Logical Files (mean of 33 versus 49). The size ranges from 30 to 672 Function Points.

In our study we found relatively few scripts and links compared to multimedia files and web building blocks. Overall, the additional Web Object component counts seem to be relatively low with mean numbers between 6 and 15 Web Objects. However, for some projects the percentage of the additional Web Object components out of the total size is more than 50%, e.g. size measured with FP is 30 FP compared to 67 Web Objects.

The Function Points as well as Web Object Counts were then used as independent variables for OLS regression-based effort prediction models. The results from the 12-fold cross validation are presented in Table 3. Besides OLS regression, results of the expert-based estimates are presented. In parenthesis you can see which size measure is used for the prediction.

Table 3. Results summary

<table>
<thead>
<tr>
<th>Estimation Method</th>
<th>Min MRE</th>
<th>Max MRE</th>
<th>Mean MRE</th>
<th>Median MRE</th>
<th>Pred (0.25)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OLS regression (FP)</td>
<td>0.02</td>
<td>0.84</td>
<td>0.33</td>
<td>0.33</td>
<td>0.42</td>
</tr>
<tr>
<td>OLS regression (WO)</td>
<td>0.00</td>
<td>0.60</td>
<td>0.24</td>
<td>0.23</td>
<td>0.67</td>
</tr>
<tr>
<td>Allette’s Expert Opinion</td>
<td>0.12</td>
<td>0.68</td>
<td>0.37</td>
<td>0.36</td>
<td>0.25</td>
</tr>
</tbody>
</table>

The OLS regression results show that using Web Objects generally produced more accurate effort predictions than using Function Points. The Pred(0.25) values as well as the MRE values are in general lower when using Web Objects. Two thirds of the effort predictions are within the 0.25 level. When comparing the OLS regression results with the expert-based estimates, we can see a difference in mean MRE when using Web Objects as independent variable. However, using Function Points as independent variable in the regression model achieved almost no difference in accuracy. Therefore, we can conclude that the most accurate effort predictions are obtained from OLS regression model using Web Objects as independent variable.

Figure 3 illustrates the prediction results measured by the mean MRE using boxplots.

Figure 3. Box plot MRE values

The height of the box is the interquartile range and contains 50% of the observations. It is largest for OLS regression using FP. This indicates a high dispersion of the MRE values. Furthermore, the upper and lower tail indicate the distribution of the observation. Again, OLS regression using FP shows the largest distribution of the MRE values.

When comparing Allette’s estimates based on expert opinion with OLS regression using Web Objects only, there is almost no difference in terms of the boxplots.

Table 4 presents the results of the tests on significance using the t-test. We applied a simple t-test, because the data was normally distributed. For the test of OLS regression using Web Objects versus OLS regression using Function Points the hypothesis was that OLS using Web Objects predicts effort more accurate than OLS using Function Points. Therefore, we applied a one-tailed test. For the comparison of OLS regression with Allette’s estimates we applied a two-tailed test to see whether there is any difference in prediction accuracy.

Table 4. t-test results

<table>
<thead>
<tr>
<th>Compared Methods</th>
<th>Mean MRE_1</th>
<th>Mean MRE_2</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) OLS regression (WO)</td>
<td>0.24</td>
<td>0.37</td>
<td>0.16</td>
</tr>
<tr>
<td>(2) Allette’s Expert Opinion</td>
<td>0.24</td>
<td>0.37</td>
<td></td>
</tr>
<tr>
<td>(1) OLS regression (FP)</td>
<td>0.33</td>
<td>0.37</td>
<td>0.74</td>
</tr>
<tr>
<td>(2) Allette’s Expert Opinion</td>
<td>0.33</td>
<td>0.37</td>
<td></td>
</tr>
</tbody>
</table>
The two-tailed test did not yield significant differences in accuracy when comparing expert-based estimates from Allette Systems and OLS regression. The test results show that using bivariate regression with size as the only effort predictor does not significantly outperform expert-based predictions. Thus, we can conclude that system size alone does not contribute to more accurate effort predictions for the web applications in our study. Including additional cost factors in the OLS regression modeling may have a positive influence on the prediction accuracy. This has already been showed in our previous study, where we used the COBRA method and compared the prediction results [6].

However, the t-test yields significant differences in accuracy when comparing the results obtained from OLS regression using Function Points on one hand, and Web Objects on the other hand. Therefore, we can conclude that Web Objects significantly contribute to more accurate effort predictions than Function Points. This result is in compliance with Reifer’s results who also states, that Web Objects have better predictive accuracy than traditional Function Points when using these measures in cost estimation models [4]. Our study supports this statement with empirical evidence.

The main limitation of this study is that it is a very tailored model for Allette Systems with low external validity. The construct validity is restricted in terms of the size measurement of a web applications. Two of twelve web applications were measured together with two developers of Allette Systems. Furthermore, Don Reifer provided support via email in case of any difficulties in terms of counting Web Objects. Multiple operationalizations have not been performed due to resource restrictions. Nevertheless, the size measurement was not biased by the knowledge of the effort spent on the project. The internal validity of the OLS regression model is slightly restricted by maturation effects while counting Web Objects respectively Function Points.

5. Conclusions

The results indicate how valuable and important expert opinions are for effort estimation. However, the informal expert-based method purely relies on the expert, whereas the standard method used simply relies on the appropriateness of the sizing measure and its correct application. In practice, the advantage of the latter method should be carefully considered when trying to estimate the effort of a project. Thus, in cases where no experts are available or accessible when needed, a more formal method such as OLS regression using Web Objects might be preferred.

With any new technologies comes the requirement to re-assess the applicability of methods used in the past. Software development cost estimation depends on having a suitable size measure as this has consistently been shown to be the most significant cost driver. In this study we have begun the process of empirically testing the available size measures for web application effort estimation. More research will be necessary to confirm these results and to test other possible size measures for this domain.

In the study we have investigated two research questions:

- Do Web Objects contribute to significantly more accurate effort predictions for web applications than traditional Function Points?
- Which of the prediction methods (OLS and expert-based informal method) employed in this study provides more accurate effort predictions?

As far as we can confirm, this is the first published empirical industrial result on the application of Reifer’s Web Objects in effort estimation. We found that the results provide empirical evidence for the superior performance of the tailored size measure, Web Objects, over standard Function Points. Although this might have been anticipated, it is extremely important that validation such as this occurs and that the results are available to industry. It is also important that further validation of the measure occurs in other projects and organizations so that confidence in the results can be obtained. The sample used in this study is relatively small but sufficient for an initial investigation. It is also likely that additional application of the approach will suggest refinements to the Web Object measure as additional experience is gained.

It is also of some interest in the organization itself that Web Object based estimates are more accurate than the current informal estimates. Although we do not find statistical significance in this result, it suggests that further study could likely reveal improvement opportunities here.

6. ACKNOWLEDGMENTS

We would like to thank Allette Systems for supporting the research project with their project data, personnel, and financial support. We are thankful to all employees of Allette Systems who have made this work possible. This work was funded by an Australia Research Council Linkage Grant and Allette Systems.

7. REFERENCES


**Appendix**

The definitions of the Function Point as well as Web Object counting components are summarized below:

**External Inputs** are elementary processes in which data or control information crosses the boundary with the primary intent to either maintain Internal Logical File(s) or alter the systems behavior (e.g. user subscription form).

**External Outputs** are elementary processes that send data or control information outside the boundary with the primary intent to present information that is derived through processing logic as well as maintain ILF(s) or alter the systems behavior (e.g. reports, screens).

**External Inquiries** are elementary processes that consists of a data "trigger" followed by a retrieval of data that leaves the application boundary (e.g., browsing of data). No internal file is maintained by the transaction.

**Internal Logical Files** are user-identifiable groups of logically related data or control information that resides entirely within the application and is maintained by the web application.

**External Interface Files** are user-identifiable groups of logically related data or control information that is referenced from the application but maintained outside the application boundary (e.g. product or information databases).

**Multi-Media Files** are physical, persistent entities used by the web application to generate output in multi-media format (e.g. audio, video, animation, graphics, etc.).

**Web Building Blocks** are logical, persistent entities used to build the web applications and automate their functionality. Additional active (ActiveX, applets, agents, guards, cookies, etc.), fine-grained static (COM, DCOM, OLE, etc.) and course-grain reusable building blocks (shopping carts, buttons, logos, etc.).

**Scripts** are logical, persistent entities used by the web application to link internal files and building blocks together in predefined patterns (macro, distiller, API's, etc.).

**Links** are logical, persistent entities maintained by the web application to find links of interest to external applications (xml, html and query language lines).