A REVISED WEB OBJECTS METHOD TO ESTIMATE WEB APPLICATION DEVELOPMENT EFFORT

R. FOLGIERI       G. BARABINO       G. CONCAS       E. CORONA
R. DE LORENZI      M. MARCHESI       A. SEGNI

LUGLIO 2011
A Revised Web Objects method to estimate Web application development effort

R. Folgieri (1,2), G. Barabino (2), G. Concas (3), E. Corona (3), R. De Lorenzi (4), M. Marchesi (3), A. Segni (4)
(1) DEAS – Department of Economics, Business and Statistics, University of Milan, Italy
(2) DIBE – Department of Biophysical and Electronic Engineering, University of Genova, Italy
(3) DIEE – Department of Electrical and Electronic Engineering, University of Cagliari, Italy
(4) Datasiel spa – Genova, Italy
Contact E-mail: raffaella.folgieri@unimi.it

ABSTRACT

We present a study of the effectiveness of estimating web application development effort using Function Points and Web Objects methods, and a method we propose – the Revised Web Objects (RWO). RWO is an upgrading of WO method, aimed to account for new web development styles and technologies. It also introduces an up-front classification of web applications according to their size, scope and technology, to further refine their effort estimation. These methods were applied to a data-set of 24 projects obtained by Datasiel spa, a mid-sized Italian company, focused on web application projects, showing that RWO performs statistically better than WO, and roughly in the same way as FP.

Categories and Subject Descriptors
D.2 Software Engineering; D.2.8 Metrics/Measurement

General Terms

Keywords
Software metrics, Web development, Function Points, Web Objects.

1. INTRODUCTION

In a previous work [1] we analysed two methodologies aimed to estimate software development effort [2] [3]. Focusing on Web application development, we compared the effectiveness of Function Points [4] and Web Objects [5] analysis to estimate the development effort. We started from original requirements and estimates, obtained using FP analysis, and added Web Objects analysis of the same requirements.

We then compared the results of the two approaches, considering also the a posteriori data about the actual effort taken to develop the studied systems.

In the context of this first work it appeared evident that effort estimation depends not only on functional size measures, that represents the main input strongly influencing the final results, but we had to consider also other factors, such as the model accuracy and other challenges specific of web applications. So, we deem it is necessary to define new, specific estimation models, designed for the mentioned purpose.

The scope of this paper, after an overview of the state-of-the-art of main cost models, is to propose a possible new approach to web application effort estimation, able to address the specific issues related to this context.

The paper consist in a revisitation of Web Objects methodology, called Revised Web Objects (RWO) to estimate the effort in developing web projects. We analyzed the data of twenty-four real projects, divided into four categories identified on the basis of technical characteristics and project dimensions. Revised Web Objects is a mixed model, conciliating both the characteristics of empirical methods (i.e. the use of previous experiences in effort estimation), and algorithmic and statistical measurements. Our approach considers different weights, specifically tailored for web applications. It starts with a preliminary categorization of the considered project, according to a web application taxonomy designed on the basis of interaction characteristics, scope of the application, dimension of the project and tools to be used to develop the solution.

The comparison among classical Function Points methods, Web Objects (WO) and RWO demonstrates the best performance of RWO in web oriented applications.

The paper is organized as follows: Section 2 presents an overview of main current cost models, according to a taxonomy based on our experience; in Section 3 we describe our approach, consisting in an evolution of Reifer’s Web Objects [5]. Section 4 discusses the results of the experiments performed applying our method to obtain its validation. Finally, Section 5 presents the conclusions and plans for future work.
2. COST MODEL OVERVIEW

Project effort estimation metrics are based on cost models which consider as input a set of parameters – named cost drivers – among which the dimension is the main one [2][3], and give as output an effort measure.

As a first step, we considered current cost models and propose a taxonomy that groups them into three categories:

- empirical or expert-based
- algorithmic
- mixed

The proposed taxonomy does not correspond to a standard classification, but it reflects a model categorization on the basis of considerations concerning specific characteristics. In particular, we consider:

- the codification (or not) of specific steps to compute the effort, starting from well-defined parameters;
- the involvement of human factors;
- the capability to establish a relationship between the system dimension, expressed using a dimensional metric, and the effort measure given as output.

The empirical/expert-based models refer to models not based on codified computation steps, but based instead on previous experiences. These kinds of models are, in fact, informal and comparable to a supervised approach. These techniques are strongly experience-driven, and are based on the human knowledge of the process, of the development team and of all the influencing factors. Consequently, they are hardly repeatable, and rely heavily on the ability and experience of the actual human estimators.

The algorithmic models are those based on specific rules, translated into automatic procedures, aiming to obtain a calculation of the effort estimation. This category is constituted by all models built abstracting from a set of existing projects that, after the input and the calibration of the various context-specific cost drivers, produce an estimation. These models, like COCOMO [2] and COCOMO II [6], have a limited calibration cost, but give an estimation that is often scarcely significant, due to the errors introduced (some works refer to errors of even 600% [7]).

There are also models designed for Web systems, like WebMO, the cost model associated to Web Objects[8], or COBRA [9]. Some of these methods could be made more efficient using a larger knowledge base and more significant projects, and designing a specific model for the data set (for instance, a regression model).

Finally, in the mixed model category, we group all the other cases, in which there is not a clear boundary between the influence of the “human factor” and of the algorithmic component. An example of this kind of models are analogy-based techniques, using data from existing ended projects, stored in a database. Similar projects are used to estimate the effort on new projects, following predefined rules, but it is necessary a human intervention to define what are the similarity characteristics on which to operate. Another example could be a completely algorithmic model, taking as input a dimensional metrics purely subjective, that needs to be computed manually.

Among all, the mixed approach presents the advantage to merge two aspects equally important in the evaluation of complex software – objective measures and subjective factors linked to the experience and the ability to follow the quick evolution of the programming technologies, for instance characterizing the development of Web applications.

3. THE PROPOSED APPROACH: RWO

The aim of our study is to propose a new model, the Revised Web Objects, consisting in an extension of Reifer’s Web Objects method. In our previous research, we considered the real data of a set of Web projects in the context of Datasiel spa, a mid-sized Italian software company. For each project, the company performs a detailed Function Point estimation, computed following the “traditional” approach: five function types are defined, grouped in data (interface logical files, external interface files) and transactions (external inputs, external outputs, external inquiries). In the early estimation of the project, the company uses UAF (Unadjusted Function Points) measure, neglecting VAF factor (Value Adjustment Factor), as it is better described in the next paragraph. We found that WO showed an estimation error similar to that existing using FP, with a slight advantage for FP [1]. This is probably due to the company expertise in estimating many previous similar projects using FP, while WO was first used in the reported study. In fact, the company used Unadjusted Function Points that, despite their simplification, yield good results.

On the basis of this previous work, we devised the new RWO method, that takes into account the classical parameters of WO but recomputing the original indicators and, when we deem they have become obsolete due to new advances in the technology, incorporates our practical experience in effort estimation.

Of course, it is usually necessary to tune the proposed RWO method with respect to a productivity coefficient that depends on the adopted technology and, consequently, on the experience of the company performing specific projects. In this way, the proposed approach does not exclude the human factor, which is obviously unpredictable, but is based on the developers' experience and skills, and thus becomes a mixed approach.

Following the original WO indications, the elements we considered in RWO are divided in operands and operators, defined as following:

- operands: the elements themselves
- operators: the operations we can perform on the operands

Actually, in various counting examples (particularly in the White Paper describing the official counting conventions [8]), Reifer himself does not use this equation, but he just sums operands and operators, each weighted by a number depending on the complexity of the considered item. We use the same approach for the four kinds of operands introduced by Web Objects, in the followings described with related operators and complexity weights for “Low, Average, High” grades, reported inside the parenthesis after the name of the element, in the same order.

In the original definition, Multimedia Files (complexity low or average, depending on kind of multimedia files) are dimension predictors developed to evaluate the effort required to integrate audio, video and images in applications. They are used to evaluate the effort related to the multimedia side of a web page.
In this category we can include: images, audio, video, texts. In this case, the image considered are those related to the content of a website (for example the photos or thumbnails in a photo-gallery), not the images present in the interface (icons). Audio and video are multimedia files that can be downloaded or interactively played by the users. Also in this case, audio or video files present in the interface are not considered as multimedia files. The text eligible to be considered as multimedia file is not the text present in a web page, but text files, for instance in .pdf, .doc, .odt, and other formats. Also, texts or files generated by a script (for example a form that, when compiled, generates a .pdf file as a result) are not to be considered in this category.

We redefined the original metric guidelines, in some cases yet obsolete, to better fit the actual characteristics of current web applications. We upgrade the considered elements as follows:

- **images:**
  - generic, static format: Low
  - animated images (for example, animated GIF): Low or Average
- **audio/video:**
  - common A/V formats (for example MP3, AVI, Flash): Average
  - streaming A/V: High
- **text:**
  - for all formats: Low

Concerning typical operators for multimedia files, we considered the following categories and weights:

- **start/stop/forward for A/V files:** Low or negligible
- **operations on interactive elements (for example, a search on a map):** Low or Average

Web Building Blocks (complexity generally low or average in some cases, depending on kind of blocks) are dimension predictors used to evaluate the effort required in the development of all the components of a page of the application in the original WO. Standard libraries (such as Windows components or graphical libraries in Java) are not considered since they are part of their own environment. Our definition considers, instead, active elements such as ActiveX, applets, agents and so on, static elements like COM, DCOM, OLE, etc., and reusable elements such as shopping carts, buttons, logos and so on. All the elements recurring on more than one page are counted just once (an example is given by the buttons performing the same operation).

We consider:

- Buttons and icons, both customized widget and static images, with the activation as the only associated operator (Low)
- Pop-up menus and tree-buttons have to be considered twice: the first time as buttons (Web Building Blocks); the second as operators (counting them as many times as the number of their functions). All these operators have a Low complexity.
- Logos, headers and footers are all static elements present in the website interface. This kind of elements are often unknown in the early stage of a project. So, their count depends on the details of the requirement document available. Concerning the complexity, we can typically consider:
  - Buttons, logos, icons, etc: Low
  - Applet, widget, etc: Average or High

Scripts (complexity low with different levels, depending on kind of scripts) are dimension predictors developed to evaluate the effort required to create the code needed to link data and to execute queries internal to the application; to automatically generate reports; to integrate and execute applications and dynamic content like streaming video, real-time 3D, graphical effects, guided work-flow, batch capture, etc., both for clients and for servers. It is important to clarify the difference between a script and a multimedia file: a script is the code that activates, possibly, a multimedia file.

In our model, this category also includes:

- breadcrumb: information generally present in the top of the page, allowing a quick navigation. For this element we consider a complexity Low – Average
- pop-ups
- Internal DB queries: queries internal to the application, with complexity depending on the adopted technology.

In fact, Reifer uses the conventions defined by the Software Engineering Institute:

- html: Low
- query line: medium
- xml: high

In the projects we analyzed, we used a Low weight for DB query when a persistent framework, like Hibernate, was used. In fact, once defined the mapping of the objects in xml language, the query becomes an access to the fields of the objects, highly reducing complexity.

Usually, the complexity of these elements is considered Low-Average.

Links (complexity low or average, depending on kind of links) are dimension predictors developed to evaluate the effort required to link external applications, to dynamically integrate them, or to permanently bind the application to a database.

Links are always present when the application performs queries on databases external to the application itself. Consequently, the code to access data is considered a link. In the analyzed projects, the login is always considered as an external link, because the database holding the users' data is external in every case.

Concerning the complexity, Reifer counts the logical, and not the physical, lines of code. In our model, we follow the same approach used for the scripts, considering the complexity depending on the persistence technology adopted.

When evaluating the effort estimation for a web application project, the reported characteristics to be taken into account are typically not enough. In fact, web applications may have very different scopes objectives and interactivity level – from a simple collection of Web pages to a full client-server complex data processing application – and may be developed with very different technologies, characterized by very different productivities. These "environmental" and "basic genre" features must be taken into account for a realistic effort estimation. So, to incorporate this essential element influencing effort evaluation, in the early stage of the design of a web application, we also need to identify the kind of application to be developed. To this purpose, we incorporated in RWO method also a preliminary evaluation of the kind of the project. In this way, the guidelines for calculating the development effort can account for different parameters
resulting from the technologies used for the development of the web application, and from the development language chosen. Thus, the additional information we consider is the classification of each project. One of the aims of this experimentation is to confirm the general validity of the methods for different kinds of projects. Our categorization is made on the basis of three features:

- size (in terms of FP/RWO);
- level of reuse of other applications;
- productivity of the tools used.

The size is the estimation performed in terms of basic RWO measures, allowing to have a first, rough indication of the effort needed to complete the project.

The level of reuse is used to evaluate how many software component can be taken from previous projects, minimizing the development effort.

Concerning the productivity, this is a fundamental element completing the taxonomy and adopted by the company after accurate validation. Summarizing, projects are classified following the indications showed in Table 1.

Once a project is classified, specific weights are used to obtain the estimated effort from the computed basic RWO measures.

### Table 1. Project classification

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
<th>Features (programming language, tipology, architecture)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSP</td>
<td>Standard Software Project</td>
<td>Java, No framework, No RAD</td>
</tr>
<tr>
<td>SRP</td>
<td>Standard RAD Project</td>
<td>The skeleton of the application is developed using a RAD tool, while its detailed interface and business code are coded by hand. This category needs additional studies.</td>
</tr>
<tr>
<td>ERP</td>
<td>Extreme RAD Project</td>
<td>The application is developed using a tool that does not require particular programming skills, and no a priori knowledge, except for the ER model, constraints and validation rules. In some cases, a workflow model (static and dynamic) is needed. The RAD tool creates the database and all connected procedures. It is model-driven. An example of this kind of tools is Portofino1.</td>
</tr>
<tr>
<td>Portal</td>
<td>Broadvision architecture. Generally, portals are designed for content presentation, so they have a limited or absent data processing and management component.</td>
<td></td>
</tr>
</tbody>
</table>

1 [www. manydesigns.com/portofino.html](http://www.manydesigns.com/portofino.html)

To evaluate our RWO approach, we performed some experiments, described in the following section.

### 4. EXPERIMENTAL RESULTS

The previous empirical research [1], performed in the context of Datasiel, a mid-sized Italian software company, had the aim to compare measures obtained by the application of FP and WO methods. Choosing a narrow sample for our study (projects developed by only one company) might constitute a possible threat to the generality of the results. In this new experimental phase, we considered more projects, developed by the same company, chosen among different kinds, as defined above. In this way, we were able to consider both a larger sample and a variety of cases to which apply our RWO method.

#### 4.1 Dataset

The data set is built on 24 projects developed from 2003 to 2010 by Datasiel; this firm develops and maintains a fairly high number of software systems, mainly for local public administrative bodies.

The application domains are those in which the company operates: mainly Public Bodies and Health Services. Among the projects developed by the company, we chose the mentioned 24 ones, focusing our attention on the applications written using web technologies, which are now the most used by the company for developing its projects.

Each project is described by the requirement documentation, and by snapshots of the layout of their web pages. The company already performed the detailed Function Point estimate, allowing us to compare the results with the estimation done with RWO, following the rules detailed in the previous section. Before estimating, each project was first categorized according to the taxonomy described at the end of the previous section, and constituting the early step of RWO methodology.

In our experiments, the classification of each project was used to steer the subsequent phase, when weights are assigned to the required features.

The categorization of the studied projects was made on the basis of:

- the size (in terms of FP/RWO);
- the level of reuse of other applications;
- the productivity of the tools used.

The projects considered for the experiment belong to the cited groups in the same measure, with balanced dimensions and reuse levels. So, we had the same number (six) of SSP, SRP, ERP and Portal projects.

#### 4.2 Effort prediction and evaluation method

For each of the 24 projects we evaluated three different estimation metrics (FP, WO and RWO). Table 2 shows and compares the descriptive statistics related to effort estimation in person's hours.

Note that the output of the three methods are the rescaled in the same way, to get an estimation of the effort, which is then compared with the actual effort declared by the company for each project.
Table 2: Web projects descriptive statistics (pers./hours)

<table>
<thead>
<tr>
<th>Metric</th>
<th>min</th>
<th>max</th>
<th>mean</th>
<th>median</th>
<th>std dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>System effort in FP</td>
<td>60</td>
<td>777</td>
<td>312</td>
<td>240</td>
<td>236</td>
</tr>
<tr>
<td>System effort in WO</td>
<td>67</td>
<td>1342</td>
<td>446</td>
<td>355</td>
<td>347</td>
</tr>
<tr>
<td>System effort in RWO</td>
<td>42</td>
<td>851</td>
<td>282</td>
<td>225</td>
<td>220</td>
</tr>
</tbody>
</table>

The sizes in RWO are quite comparable to the sizes in FP. This result is encouraging, because our method, specialized for the evaluation of development effort in Web-base projects, yields results quite close to the more traditional FP method (whose use in Datasiel has been well established for many years), and apparently with less variability than with WO method.

To evaluate the performances of the measures, we calculated the MRE (Magnitude of Relative Error) for each project. MRE is defined as follow:

\[
MRE = \left( \frac{EFFORT_{ACTUAL} - EFFORT_{ESTIMATED}}{EFFORT_{ACTUAL}} \right)
\]

In addition, we also calculated the prediction level \([11][12]\)

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

where \(Pr ed(l)\) is the percentage of projects with an MRE less than or equal to \(l\). Analysis and tests have been performed using the R Project suite.

4.3 Results

The results obtained with the selected projects using FP, WO and RWO metrics are shown in Table 3. They show that RWO methods perform better than, or equal to, FP on many considered projects. Consequently, we can consider RWO an overall valid alternative to FP, surely more tailored to satisfy the needs of effort prediction for a web application.

Table 3: Error values

<table>
<thead>
<tr>
<th>MRE</th>
<th>min</th>
<th>max</th>
<th>mean</th>
<th>median</th>
<th>std dev</th>
<th>Pred (0.25)</th>
</tr>
</thead>
<tbody>
<tr>
<td>FP</td>
<td>0.00</td>
<td>3.13</td>
<td>0.49</td>
<td>0.19</td>
<td>0.90</td>
<td>62%</td>
</tr>
<tr>
<td>WO</td>
<td>0.07</td>
<td>5.93</td>
<td>1.23</td>
<td>0.66</td>
<td>1.64</td>
<td>40%</td>
</tr>
<tr>
<td>RWO</td>
<td>0.01</td>
<td>3.05</td>
<td>0.45</td>
<td>0.19</td>
<td>0.85</td>
<td>58%</td>
</tr>
</tbody>
</table>

Note that non-revised WO yields poor results on the considered dataset, while our revised method yields results quite reliable for effort prediction on web application projects data. Remember that having a \(Pr ed(0.25)\) greater than 75% (more than 75% of the projects have an MRE less than, or equal to 0.25) denotes an acceptable estimation [11]. If we follow this criterion, both FP and RWO do not give acceptable estimations. However, considering current estimation models, we can affirm that RWO is an acceptable estimation method for the target projects. Apparently, RWO performs similarly to FP and both seem better than WO.

Concerning the apparent similarity between RWO and FP performance, we have to consider that several projects belonging to the considered data set do not have strong, web-specific characteristics. Moreover, there is a significant data dispersion due to the presence of projects developed using RAD technology. In fact, the RWO method appears more reliable compared to FP in the case of complete web applications, being in this case more stable and predictable. We should also consider that RWO is tailored for web applications, and could be further refined following the evolution of web technology, while such a tailoring would be much harder with FP method.

Note that the number of studied projects, even if belonging to various kinds of application, is too low to be definitive about the validity of the proposed RWO method.

Figure 1 shows the estimated effort vs. the actual Lines of Code (LOC) for the considered standard Java systems. We did not consider data pertaining to the systems developed using RAD or .NET approaches, because less dependent on effort estimation. We found that there a is strong linear correlation between RWO and LOC (we obtained a correlation coefficient equal to 0.99), while FP and LOC correlation is slightly worse (in this case, the correlation coefficient is 0.88).

5. CONCLUSION AND FUTURE WORK

We presented an empirical study of software development effort estimation, performed on a set of industrial projects carried out at Datasiel spa, an Italian software company. The considered data set includes 24 projects divided in 4 categories, allowing to extend and generalized results to different kinds of web application projects. The data set is composed, in equal measure, by Standard Software Projects (SSP), Standard RAD projects (SRP), Extreme RAD Project (ERP) and Portals.
The performed experiment compared the estimation power of different methods—Function Points, Web Objects, and our Revised Web Objects method.

All the estimation was done considering a productivity coefficient formulated by the company on the basis of past development experiences. We believe that entirely empirical methods are not efficient enough, because they do not give an objective measurement of the project effort, but depend on a human estimator on the basis of her own previous experience. On the other hand, mixed models take advantage of both algorithmic and empirical methods. In the real world, in fact, the early estimation of a project effort cannot be based only on one of the two aspects. For this reason, we revisited the WO method, adding other parameters designed to provide forecasts based also on human experience, and at the same time specifically formulated for the prediction of effort in developing web applications.

In the specific context, good results were obtained both with FP and RWO methods. As previously discussed, FP method yielded good results owing to the long experience of Datasiel developers in its use. RWO, on the other hand, was able to yield comparable—and even slightly better—results since its first use. The RWO approach accounts for specific web application characteristics, and is suitable of further evolution, following web application technology changes.

Besides performing a comparative analysis of these—and possibly other—effort estimation methods on a larger sample of projects, a further step in the research will consist in developing a tool, based on the proposed RWO model, allowing to perform a predictive effort estimation. The tool will allow to customize productivity parameters, so that the model could evolve following new acquired competencies, new technologies and the different web applications considered.

6. REFERENCES


