Calculating Software Projects on a Mockup based approach

Axel Kalenborn, Daniel Kuhn, Fabian Lorig

Universität Trier, Wirtschaftsinformatik
kalenbor@uni-trier.de, kuhnd@uni-trier.de, lorigf@uni-trier.de

Abstract. Before starting software projects proper cost estimation for accurate project planning and staffing is required. The project requirements, which are normally not well defined in the early stage of a project, need to be considered carefully for a comprehensive calculation. In order to discuss and understand the project requirements, Mock-Ups or Prototypes are often used, as they provide the possibility of analyzing the potential customer's requirements. Because they show the way the software works and how they get in touch with it, Mock-Ups are a suitable approach for visualizing needs. In this paper we propose a method using Mock-Ups not only to understand and discuss the project requirements but for proper cost estimation. For achieving this, we use the semantically enriched Mock-Up elements for cost estimation.

Keywords: Mock-Ups, Cost Estimation, Requirements Engineering

1 Motivation

Regarding the common process models of software engineering, the projects usually begin with an extensive stage of requirements engineering and then pass on to conception and realization [1], [5]. In this stages different methods of requirements engineering and cost estimation can be used to specify the requirements and calculate the software to be implemented in collaboration with the principal, in case of sufficient time and money [3].

In practice, the initial analysis of the requirements and the documentation do not take place after signing the contract. This process takes place already before the start of the software project within the bid preparation. It is primarily used to enhance the comprehension of the project and to provide a basis for the bid and/or the contract to be concluded with a prospective customer.

During the bidding stage, detailed requirements analysis are not yet possible because the analysis is not being paid and the bidders are competing with other suppliers. Nevertheless, the software needs to be presented to the potential client and a proposal with a detailed cost estimation needs to be written.

For the presentation, Mock-Ups turned out to be useful, because the visualization is the only element of a project that can be discussed by the decision makers. Technical details are so complex that only IT professionals can understand and assess them.

This phenomenon is known as IKIWISI[1] and deals with the problem that the software users do not understand the requirements until they see them [2].

Because of this, drafts, screenshots and HTML prototypes have been established in the context of bid preparation of internet based projects in these days and have to be created to keep the chance to award a contract.

IT budgets are running short and require an efficient approach when preparing a bid. The paper addresses this problem and presents a way to integrate the process of creating and calculating a project in a Mock-Up based approach. In this approach the calculation is one aspect that is integrated in the Mock-Up while modeling it. Thus, the focus of this papers lies on dialog oriented and internet based systems.

2 State of the Art

During the bidding stage there are two challenges the bidders are facing. They have to gather the customer's requirements. A suitable approach is using a Mock-Up to visualize the requirements for evaluating them in collaboration with the principal. The reason is the high importance of the usability and the presentation layer [6]. Likewise they have to prepare a precise calculation for the project.

For the Mock-Up creation the bidders often use graphic programs or HTML editors. Though, these tools only focus the visual layer. There is no chance to provide any information about the cost estimation or a technical description. This has to take place in a separate step.

The cost estimation occurs in assistance with estimation methods. Those can be distinguishing in empiric and algorithmic estimation methods. The empiric estimation methods use ascertainment and analyses provided by previously calculated projects and therefore cause minimal effort. Consequently the estimation is not as accurate as the estimation based on algorithmic methods.

Algorithmic estimations (also known as parametric estimations), such as COCOMO II, use mathematical formulas for cost estimation. To determine the parameters, the empirical results will be analyzed regarding relevant values, i.e. those who affect the project time or cost.

A study by Molokken and Jorgensen describes that empiric estimations (70-80%) were more often used than algorithmic estimations (10-20%) in practice [7].
The cost estimation quality is highly influenced by the experience of the estimator as well as the precision of the quantity structure of costs which was defined for the cost estimation [8]. Thus, in case the project is in an early state, empirical methods need to be used for cost estimation, due to a lack of information. Algorithmic methods are not easy to use because the analysis for defining suitable arguments is complex. The cost estimation with empiric methods assumes that in the past similar project have been calculated. In case there are not such projects given, the estimation will become more difficult and the use of risk markups will be required. However, in the worst case using risk markups in cost estimation will cause the offer to be no longer competitive.

[1] IKIWISI: I know it, when I see it

To avoid this dilemma, we present an approach using Mock-Ups to provide a process of cost estimation.

3 Mock-Ups for software calculation

As mentioned before, Mock-Ups are usually used for representing the presentation layer. However, the *Modeling by Example* approach describes a different strategy. Beside the visualization of the requirements, this method tends to enrich Mock-Ups with semantic information. The aim of semantic enrichment is an automatic creation of a project calculation as well as a technical and professional specification.

3.1 The modeling by example method

The idea of Modelling by Example refers to the Query by Example approach [9]. With this method the user can add entries to a database without minding different database language as for example SQL. The Modeling by Example method transfers this approach to requirements engineering of web applications. Instead of making the user learn new methods of representation or modulation, the resulting presentation layer will be used for the conception of the web application [10].

The construction of Mock-Ups follows the template concept, which is widespread in the development of web applications. The template describes the website's "Look & Feel", which will be included in all subsites. Furthermore the hypertext layer is depicted on the template.

The real content of a web application contains text, pictures, form elements and further objects. An example of a form would be a contact form on a website. A contact form contains form elements for the input of user data. Furthermore, there is a submit button which triggers an action. As a result, the user sees a success or a failure site. For representing this behavior, the Modelling by Example approach defines rules for controlling the behavior of sites.

3.2 Semantical enrichment in the MbE method

The related sites (e.g. contact form) and the defined rules are summarized into a module. A useful description for the example above would be a contact form.

Sites represent the module's visual aspect. Beside this aspect, there are yet 3 another aspects. Those provide the semantic enrichment of modules (figure 1).

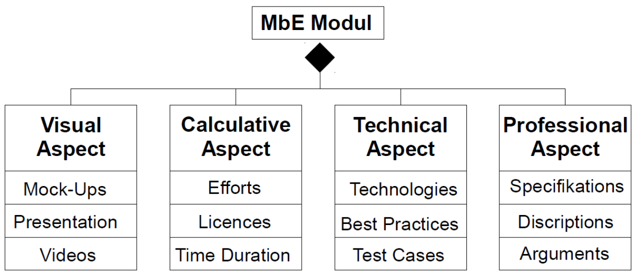

figure 1: MbE-Modul Aspects

- The professional aspect contains the description of the modules, sites and templates. It serves the annotation of the Mock-Up and its functions.
- The technical aspect contains information that can be used for creating professional and technical aspect as well as test case specifications.
- The calculative aspect contains information about the expenditure and the duration for the conversion of the project. The precise pattern of the calculation is described in chapter 4.

The three aspects descripted above can not only be allocated to the modules itself. Furthermore, also sites and those elements (text, picture etc.) that are included in the module can be enriched by these aspects.

A core aspect of the Modelling by Example method is the reuse of requirements. Each requirement like the contact form example shown above can be mapped into a module. Those modules can then be enriching with semantic information. The precalculated modules will be stored in a separate module collection.

4 Mobex: a tool for semi-automated software calculation

To prove the suitability of this approach in practice, the Mobex tool was developed, implementing the Modelling by Example method.

4.1 The Mobex Tool

The Mobex tool combines features for image editing and requirements engineering. The presentation view of the Mock-Ups is the tool’s key component according to the Modelling by Example method. An overview of all single modules and their sites will enable the user to navigate using a tree structure.

For enriching sites with content the user may choose between a set of elements given by the tool. These sets contain elements such as forms, text, pictures etc. Furthermore, Mobex provides an integrated library. In this library templates, functions and stylesheets can be defined and added to a module, site or element when required. Semantic information as described above can be added to the library elements, too.

The allocation of semantic information will be done using the properties of the objects (for example element). Semantic information will be applied to objects in a textual-based way.

In order to be able to use the semantic information, a report generator was integrated into the Mobex tool. Hereby, professional- and technical descriptions can be generated as well as an offer based on the semantic information.

4.2 The calculation schema in Mobex

To be able to make a precise cost estimation of the project, a quantity structure of the components which are going to be developed is needed. To receive this quantity structure, different variables like lines of code, the number of input and output masks or input and output data that the software produces, etc. can be used [3]. Some of these variables can be used while creating Mock-Ups.

For providing calculatory aspects of the Mock-Ups, the elements will be enriched like analogue to the technical and functional aspects. Each element of a Mock-Up can be arbitrarily assigned to many categories of expenses and provided with cost rates and times. In addition, a risk premium in the form of a percentage mark-up rate is provided at the level of modules. This risk premium enables the Mobex user to account the risk of complex application parts. The anchoring of the risk premiums on the level of modules emphasizes their characteristics as a self-contained unit. They are modeled as such and can be reused.

To increase the flexibility of the calculation, a schema combining a *cost type* (CT) with a *time dimension* (TD) a *cost rate* (CR) a *time unit* (TU) and a *value* (V) is used. As time dimensions (TD), hours, days, weeks, months and years are provided. In order to map expenses, which have no time dimension, such as license fees or hardware procurement, the time dimension (TD) can take the value "no", and the time unit (TU) will then be ignored in the calculation.

Three examples illustrate the possibilities of the calculation scheme:

- One (TU) hour (TD) Customizing (CT) costs 80 € (CR) and the customizing of a captcha module in a contact form takes two hours (V), so cost of 160 € arise.
- The license costs for a database (CT) amounts to 4.500 € (CR) and two licenses for the project (V) are required, resulting to 9.000 € costs of yield.
- For the rental of a server (CT) are each (TU) Month (TD) 250 € (CR) computes a year (V) server rent thus causes 3.000 € cost.

The total costs (TC) for the implementation of a project are then obtained from the sum of the costs of all the individual elements (e) multiplied by the respective risk premiums (RP) of the modules (m).

$$TC = \sum_{m=1}^{n} \left[\sum_{e=1}^{n} [TU_{me} * TD_{me} * CR_{me} * V_{me}] * RP_m \right]$$

In the Mobex Tool a standard set of types of expenses across projects is defined (e.g. conception, programming or testing) and can be completed in each project with individual types of expenses and priced differently to achieve maximum flexibility.

In order to facilitate the consideration of expenses directly when creating the Mock-Ups, all available elements can be provided with standard cost rates. The cost rates define the average expected expenditures for the integration of an element in the project. Causes for example, the integration and validation of a text box in a form on average 10 minutes programming effort, this can be deposited to the item. If we place 6 text boxes on the form, we need one hour programming time multiplied by the costs for programming in the specific project.

The idea behind the concept is simple, the more elements a screen mask contains, the more complex is its implementation. Although these simple rules do not necessarily apply to the whole project, as the project scope is not associated linear increase in the complexity and thus the cost, however, is the quantity structure of a valuable tool in the calculation.

5 Conclusions

In this paper an approach for enriching Mock-Ups in order to simplify the process of cost estimation has been presented. Each element which has been modeled in the Mock-Up will we part of the quantity structure. As individual cost rates can be specified for these elements, the quantity structure can be used by the sales staff for preparing their tenders.

The Mobex tool introduced in this paper combines the processes of creating a Mock-Up and simultaneously performing the calculation in the background. The resulting offer can either be used for submitting a bid directly, or as an input for parametric estimation methods, such as the function point analysis or COCOMO II. Furthermore, an expert can use these results as a basis for further calculations.

By merging the design and calculation of software projects into one tool, the time and money for submitting a competitive offer can be decreased. Hereby, resources can be saved during the usually unpaid bidding stage.

References

1. Dumke, R.: *Modernes Software Engineering*, Vieweg-Verlag, Magdeburg 1993.
2. Boehm, B.: *Requirements that Handle IKIWISI, COTS, and Rapid Change*, IEEE Computer, 33 (7): Page 99-102, IEEE Computer Society, New York 2000.
3. Sneed, H. M.: *Software Projektkalkulation*, Hanser-Verlag, Budapest 2005.
4. Thaller, G. E.: *Software-Anforderungen für Web Projekte*, Galileo Nürnberg 2002.
5. Sommerville, I.: *Software-Engineering*, 8. Aufl., St. Andrews 2007.
6. Kalenborn, A. / Fetzer, K.: *Modeling by Example: Ein Ansatz zur Unterstützung der Vorvertragsphase Internet basierter Software Projekte*, Fachtagung Modellierung 2012.
7. Molokken, K. / Jorgensen, M. 2003: *A review of software surveys on software effort estimation, ISESE*, S. 223 – 230, Lysaker 2003.
8. Hummel, O.: *Aufwandsschätzungen in der Software- und Systementwicklung kompakt (IT kompakt)*, Karlsruhe 2011.
9. Zloof, M. M.: *Query-by-Example: a data base language*, IBM Systems Journal, 16 (4): Seite 324-343, IBM Corporation, New York 1977.
10. Lechner, S.: *Web-Scheme Transformers By-Example*, Universität Linz 2004.