Theoretical and Applied Informatics
ISSN 1896–5334
Vol. 25 (2013), no. 1
pp. 3–15
DOI: 10.2478/thai-2013-0001

SAMEE – the nonlinear adaptive method for predicting work effort of information systems development

Dariusz Rafal Augustyn, Lukasz Warchal

Silesian University of Technology, Institute of Informatics,
16 Akademicka St., 44-100 Gliwice, Poland
{draugustyn,lukasz.warchal}@polsl.pl

Received 13 October2012, Revised 20 March 2013, Accepted 27 March 2013.

Abstract: At the early stage of information system analysis and design one of the challenge is to estimate total work effort needed, when only small number of analysis artifacts is available. As a solution we propose new method called SAMEE – Simple Adaptive Method for Effort Estimation. It is based on the idea of polynomial regression and uses selected UML artifacts like use cases, actors, domain classes and references between them. In this paper we describe implementation of this method in Enterprise Architect CASE tool and show simple example how to use it in real information system analysis.

Keywords: adaptive method for work effort prediction, polynomial estimation model, software analysis and design, Enterprise Architect add-in

1. Introduction

Estimation of the cost and work effort of information system development has been considered form years. The taxonomy of software estimation techniques is presented in [1]. Many methods of estimation, mostly based on Functional Points [3] and Use Case Points [2], are already implemented in different software tools. Very often they are integrated with CASE tools used for information system analysis and design. Popular CASE Enterprise Architect [7] – implementing UCP method [10] – can be an example of such software.

In this paper we propose a new method for work effort estimation based on selected UML artifacts like use cases, actors, business classes, and class references, all created at the very early stage of system analysis. The method is called SAMEE – Simple Adaptive Method of Effort Estimation. The article also presents implementation of SAMEE method as an add-in for Enterprise Architect.

The existing estimation methods require setting environmental or technical parameters [2, 5, 10] (e.g. till 13 technical complexity factors and 8 environmental factors for UCP method). The meaning of the parameters may be ambiguous in particular information system or their values may be difficult to determine. SAMEE can adopt to a particular software development process, which is specific to concrete software company. It also does not need to set any technical or environmental parameters.

SAMEE is an adaptive method based on the concept of polynomial regression. Using such elementary approach SAMEE allows to obtain a nonlinear model of effort estimation.

In SAMEE we assume that two stages should be performed: the learning phase and the predicting one. The learning phase bases on historical data (i.e. known work effort of already developed software). The method does not obtain a system size as a base for computing a work effort, but it calculates work effort directly. In the predicting phase a user can obtain estimations of unknown work effort values using the learned SAMEE model.

The nonlinear model applied in SAMEE may be more flexible comparing to other methods where some linear dependency is assumed. For example in UCP method, the work effort depends linearly on unadjusted use case points (although UCP method supports simple nonlinear model which results from weights for use cases and actors with different complexities).

SAMEE is capable to obtain an explicit expression describing a dependency between the estimated work effort and the number of artifacts. This information allows to point out elements of designed system that have supreme impact on the total work effort.

SAMEE may be classified as a learning-oriented method (but the applied technique lets classify it as a regression-based method, too) [1]. Taking into account the types of the input artifacts there is some similarity between this method and the Karner's one.

2. Theoretical Background

Let us assume that some information system functionality can be described by set of features, diagrams of use cases and class diagrams.

Let us assume that the system functionality at the highest level of abstraction can be described by a set of features (high level requirements). A feature can be realized by one or more use cases. Use cases may be associated with collaborations which contain class diagrams describing their static aspects.

SAMEE method allows to estimate the unknown work effort (denoted by y) for selected feature basing on some selected 4 types of early analysis artifacts like:

- number of use cases (denoted by x_1),
- number of actors (denoted by x_2) associated with the use case,
- number of classes (denoted by x_3) in collaboration which realize the particular use case,
- number of references (denoted by x_4) between the mentioned-above classes.

In SAMEE we assume a nonlinear model of dependency between work effort of a feature and number of use cases, actors, classes and references – we propose to use a polynomial dependency as follows:

$$y(x_1, \ldots, x_4) = \sum_{k=1}^{4} a_{0k} x_k + \sum_{j=1}^{4} \sum_{k=j}^{4} a_{jk} x_j x_k, \tag{1}$$

were a_{jk} are model parameters.

This seems to be a more adaptive approach than UCP method because it supports a nonliner model and it takes into account a dependency between pairs of variables ($x_i x_j$ terms in eq. (1)).

Let us assume that there are some historical data about the past development of some information system, i.e. we know values in Y vector and X matrix:

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_i \\ \vdots \\ y_N \end{bmatrix}, X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ \vdots & \vdots & \vdots & \vdots \\ x_{i1} & x_{i2} & x_{i3} & x_{i4} \\ \vdots & \vdots & \vdots & \vdots \\ x_{N1} & x_{N2} & x_{N3} & x_{N4} \end{bmatrix}, \tag{2}$$

where $i = 1 \ldots N$, and N is a number of features realized in the information system. X and Y are training set values.

There are many method of finding values of parameters a_{ij} using data from training set. Here we propose to adopt the well-known nonlinear regression method.

Let Q denotes a score function given as follows:

$$\begin{aligned} Q(a_{01}, \ldots, a_{04}, a_{11}, \ldots, a_{14}, a_{22}, \ldots, a_{24}, a_{33}, a_{34}, a_{44}) = \\ = \sum_{i=1}^{N} (y_i - \sum_{k=1}^{4} a_{0k} x_{ik} - \sum_{j=1}^{4} \sum_{k=j}^{4} a_{jk} x_{ij} x_{ik})^2. \end{aligned} \tag{3}$$

Finding optimal values of $a_{01}, \ldots, a_{04}$, $a_{11}, \ldots, a_{14}$, $a_{22}, \ldots, a_{24}$, a_{33}, a_{34}, a_{44} is equivalent to finding the minimum of the multivariate function Q given by eq. (3). The necessary condition of existence an extremum of Q function allows to formulate the following set of 14 equations:

$$\frac{\partial Q}{\partial a_{jk}} = 0, \tag{4}$$

for ($j = 0$ and $k = 1, \ldots, 4$) or ($j = 1, \ldots, 4$ and $k = 1, \ldots, 4$ and $j \leq k$). The equations given by (4) are equivalent to the equations as follows:

$$\begin{gathered}
\frac{\partial Q}{\partial a_{01}} = 2 \sum_{i=1}^{N} (y_i - \sum_{k=1}^{4} a_{0k} x_{ik} - \sum_{j=1}^{4} \sum_{k=j}^{4} a_{jk} x_{ij} x_{ik}) x_{i1} = 0 \\
\vdots \\
\frac{\partial Q}{\partial a_{23}} = 2 \sum_{i=1}^{N} (y_i - \sum_{k=1}^{4} a_{0k} x_{ik} - \sum_{j=1}^{4} \sum_{k=j}^{4} a_{jk} x_{ij} x_{ik}) x_{i2} x_{i3} = 0, \\
\vdots \\
\frac{\partial Q}{\partial a_{44}} = 2 \sum_{i=1}^{N} (y_i - \sum_{k=1}^{4} a_{0k} x_{ik} - \sum_{j=1}^{4} \sum_{k=j}^{4} a_{jk} x_{ij} x_{ik}) x_{i4}^2 = 0.
\end{gathered} \tag{5}$$

Finally we can find a_{jk} from the linear equations system:

$$\begin{gathered}
\sum_{k=1}^{4} a_{0k} \left[\sum_{i=1}^{N} x_{ik} x_{i1} \right] + \sum_{j=1}^{4} \sum_{k=j}^{4} a_{jk} \left[\sum_{i=1}^{N} x_{ij} x_{ik} x_{i1} \right] = \sum_{i=1}^{N} y_i x_{i1} \\
\vdots \\
\sum_{k=1}^{4} a_{0k} \left[\sum_{i=1}^{N} x_{ik} x_{i2} x_{i3} \right] + \sum_{j=1}^{4} \sum_{k=j}^{4} a_{jk} \left[\sum_{i=1}^{N} x_{ij} x_{ik} x_{i2} x_{i3} \right] = \sum_{i=1}^{N} y_i x_{i2} x_{i3}, \\
\vdots \\
\sum_{k=1}^{4} a_{0k} \left[\sum_{i=1}^{N} x_{ik} x_{i4}^2 \right] + \sum_{j=1}^{4} \sum_{k=j}^{4} a_{jk} \left[\sum_{i=1}^{N} x_{ij} x_{ik} x_{i4}^2 \right] = \sum_{i=1}^{N} y_i x_{i4}^2.
\end{gathered} \tag{6}$$

At the beginning of the method we solve the equation system given by (6). After finding values of a_{jk} some of them are rejected. This applies to those ones that have an insignificant influence on the final effort values. For every parameter a_{jk} of a sum in (1) a parameter weight is defined as follows:

$$w_{a_{jk}} = \frac{1}{N} \sum_{i=1}^{N} \frac{|y(x_{i1}, x_{i2}, x_{i3}, x_{i4})|_{a_{jk}=0} - y_i|}{y_i} \tag{7}$$

which denotes a mean relative error of effort estimation when the component with a_{jk} does not occur in eq. (1). We take into account only those a_{jk} for which w_{ajk} are big relative to the others, e.g. greater than 5% of mean value. 5% is some value threshold parameter which is called *RejectThreshold*. The remaining a_{jk} are rejected.

In the second stage of the method we formulate a new simpler score function and respectively a new equation system only for those parameters a_{jk} which were selected at the previous stage as significant coefficients. This allows to use a low dimensional approximate model and calculate final work effort values using only a subset of parameters a_{jk}.

3. SAMEE Method Evaluation

SAMEE method was evaluated on real information system designed and developed in a company from the healthcare sector. The main goal of this software was to provide patients an on-line functions that they can use to book medical services.

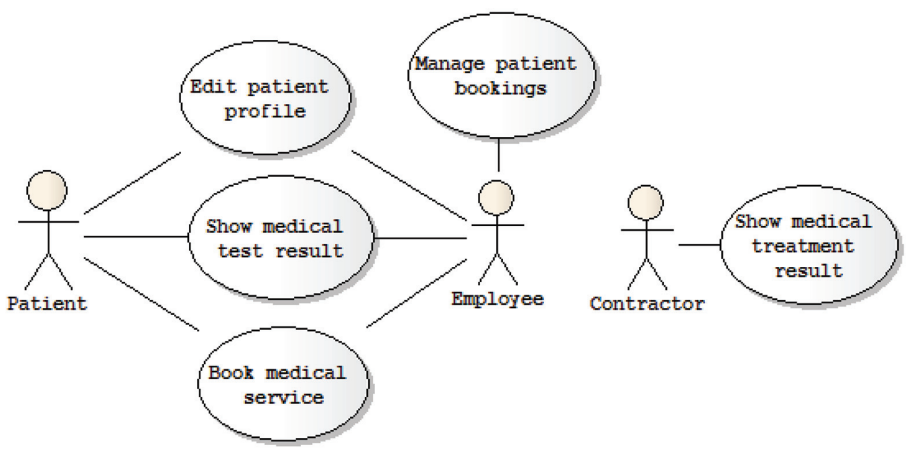


Fig. 1. Use case diagram for small part of analyzed information system

3.1. Training Data Acquisition

In this software company the development process is use case driven. It is based on the idea of Unified Process Framework (UP) and partially similar to RUP [4] and OpenUP [11].

At the early stage of the project, analysts created set of features, which outline system functionality. Then those features were decomposed to use cases and actors.

Fig. 1 shows a fragment of an use case diagram for the system. To easily trace which use cases are related to which features, feature realization diagrams were drawn. Fig. 2 presents such diagram. During system design many domain classes were also created. Some of them were placed on class diagrams describing static part of collaborations, which were used to show use case realizations (see Fig. 5).

When a part of the system was developed, work efforts of several features were exactly known. With this information and knowledge from analysis artifacts mentioned above the training dataset was build (see eq. 2).

3.2. Estimation Model

Using SAMEE we obtained estimation model over the set of 27 features ($N = 27$) which describes the medical service booking system. The result is as follows:

$$y(x_1, \ldots, x_4) = a_{01}x_1 + a_{02}x_2 + a_{03}x_3 + a_{34}x_3x_4 \tag{8}$$

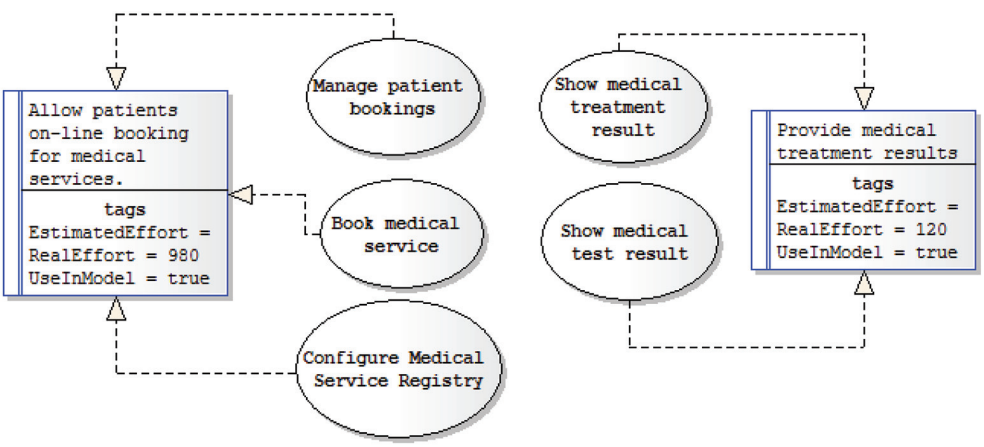


Fig. 2. Feature realization diagram for two sample features

were y is the work effort expressed in person-days and $a_{01} = 102.6$, $a_{02} = 81.7$, $a_{03} = 21.6$, $a_{34} = 0.54$.

The interesting observation is that values of a_{01} and a_{02} are rather similar and big (comparing to small values of a_{03} and a_{34}). This fact corresponds to the assumption made in G. Karner's method that UUCPs are based only on Unadjusted Use Cases and Unadjusted Actor.

The contribution of variable x_i in formula (1) may be find as follows:

$$c(x_i) = \frac{1}{N}\sum_{j=1}^{N} \left.\frac{\partial y(x_1,\ldots,x_4)}{\partial x_i}\right|_{x_1=x_{j1},x_2=x_{j2},x_3=x_{j3},x_4=x_{j4}}. \quad (9)$$

For example, for formula (8) the contribution of x_1 and x_3 may be obtained from:

$$c(x_1) = a_{01} = 102.6, c(x_3) = a_{03} + \frac{a_{34}}{N}\sum_{j=1}^{N} x_{j4} = 21.6 + 9.36 \approx 31. \quad (10)$$

Found all $c(x_i)$ for the analyzed system allow to state that use cases (x_1) had the greatest impact on the final work effort estimation.

Finding the explicit expression (8) for y was not the main goal of our work. We suppose that for a different software company (with a different development process, staff, software tools, programming language, frameworks etc.) the expression will differ too. We assume that our estimation method may adopt to local environment with every use.

3.3. Estimation Validation

To validate our method we used it to estimate work effort for 12 completely new features. Calculated effort values were similar to those provided by 3 independent specialists from the software company, who based only on their experience and knowledge about very similar systems developed in the past. The SAMEE estimated effort values differs maximally about 14% from values given by those experts. We may say that SAMEE estimations of feature work effort agree with expert's predictions for this given validation set of artifacts.

4. SAMEE Method Implementation in Enterprise Architect

The SAMEE method was implemented in widely known CASE tool – Enterprise Architect. Authors of this software provide a library that allows to browse and manipulate artifacts repository and implement add-ins that can extend basic capabilities of this tool.

As mentioned in section 3 the proposed method uses information from UML diagrams. The method is based on features and it allows feature nesting. During learning phase we use feature realization diagrams. SAMEE method requires features to have some additional attributes so, as shown in Fig. 2, they have special tagged values. *UseInModel* tagged value allows to explicit denote features that may be used duiring learning model. *RealEffort* contains the real effort that was necessary to

implement given feature (based on historical training data). *EstimatedEffort* contains estimated work effort (based on SAMEE calculation).

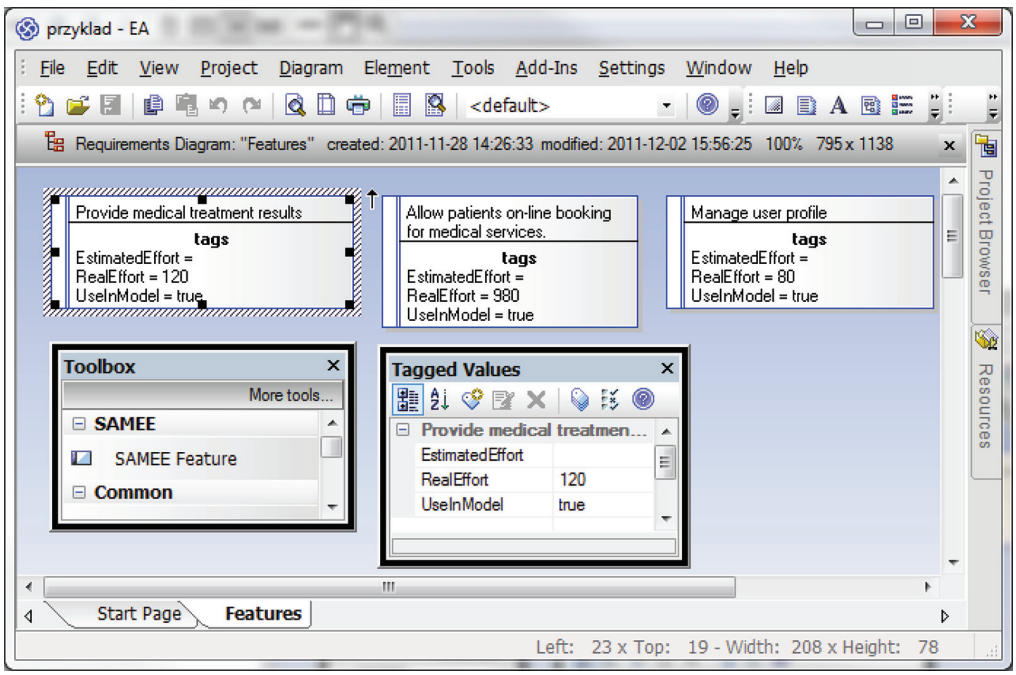


Fig. 3. Enterpise Architect with sample feature diagram

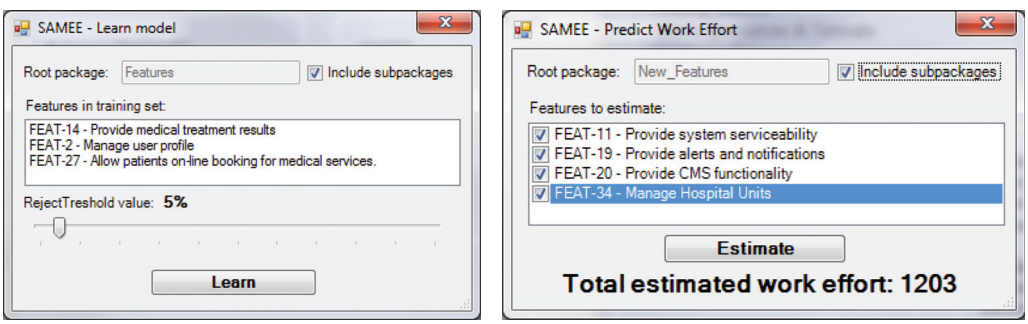
To simplify SAMEE usage, we provide UML profile [9] with custom feature element with mentioned tagged values. Such feature can be easily dragged from a toolbox and dropped on particular diagram. Fig. 3 presents Enterprise Architect main window with:

- sample feature diagram,
- toolbox with custom SAMEE feature element,
- tagged values editor opedned for a selected feature.

4.1. Learning

Learning phase starts when user selects a particular package in *Project Browser* window and runs SAMEE add-in (see Fig. 4a). At first the learning algorithm searches for all feature elements in the package that have *UseInModel* set to *true*

and obtains *RealEffort* values (y_i in eq. (1)). Then the algorithm, based on feature realization diagrams, finds the numbers of all use cases that realize each of those features (x_{i1} values). Next, for every of those use cases the numbers of associated actors is obtained (x_{i2} values). Then we need to know which of designed classes are involved in accomplishing use case requirements, therefore we analyze use case realization diagrams. In our case those diagrams describe which collaborations realize which use cases (see Fig. 5). Each collaboration can contain one or more class diagrams (describing its static aspect) and some activity diagrams (presenting its behavior), respectively. Analyzing class diagrams we finally obtain the number of classes (x_{i3} values) and relations between them (x_{i4} values).



(a) (b)

Fig. 4. SAMEE add-in sample screens: (a) launching model learning on *Features* package, (b) launching work effort estimation on *New Features package*

With this knowledge we are ready to formulate equations mentioned in section 2 and calculate unknown parameters a_{jk}. Then, for every a_{jk} we calculate parameter weight defined in eq. (7) and reject those below *RejectThreshold* value defined by user at the beginning of learning process (see Fig. 4a). Finally we obtain our estimation model like this one described in eq. (8).

4.2. Prediction

With already learned model we can use it to estimate work effort for some new features. Prediction process starts after choosing a proper package from *Project Browser* window in Enterprise Architect and launching add-in. In the window presented in Fig. 4b user can choose features to estimate. During estimation, for each of selected features, work effort is calculated individually and stored in its tagged value *EstimatedEffort*. At the end, all those partial efforts are summed and total work effort value is presented to the user (see Fig. 4b).

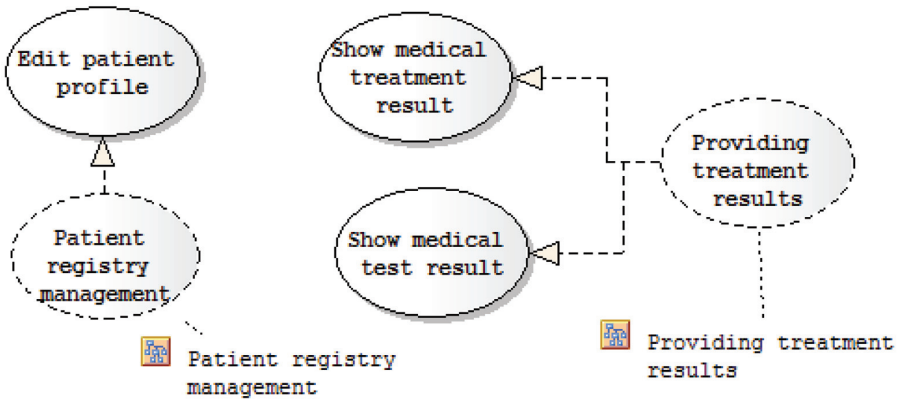


Fig. 5. Sample use case realization diagram

5. Conclusions and Future Plans

SAMEE is based on nonlinear regression (described in sections 2, 3) but there are a lot of more advanced approaches to the nonlinear system identification problem (eg. [12]). We consider to use the method based on Group Method of Data Handling Algorithm (GMDH [6]) in future. Using this method we may also find a polynomial model. Although this method is based on a hierarchy of quadratic regression polynomials, the order of the result polynomial (see eq. (11)) may be greater then 2 (now, in SAMEE we have maximal order equals 2 in the polynomial given by eq. (1)). This could make SAMEE more adaptive approach.

$$y(x_1,\ldots,x_4) = a_{0000} + \sum_{j=1}^{4} a_{j000}x_j + \sum_{j=1}^{4}\sum_{k=1}^{4} a_{jk00}x_j x_k + \ldots + \\ + \sum_{j=1}^{4}\sum_{k=1}^{4}\sum_{l=1}^{4}\sum_{m=1}^{4} a_{jklm}x_j x_k x_l x_m \tag{11}$$

We assume that SAMEE should be useful at the beginning of a complex information system developing process. The current version of SAMEE supports estimation based only on 4 types of analysis artifact (use cases, actors, classes, class references). On the basis of our practical experience, we found that more detailed model of analyzed information system is unavailable in most cases. However, there is possibility to introduce more detailed description of analysis artifacts like complexity of a use case (measured by number of scenarios), complexity of a class (measured by number of fields and/or methods).

Moreover, we can introduce levels of complexity (low, middle, high), similar to those ones well known from Function Point Estimation method [3] or UCP one [2]. Thresholds for such levels may be defined by a user and then SAMEE might automatically assign a proper complexity level value to an use case (considering number of scenarios) or a class (counting actual number of fields or methods). This approach will introduce new x variables (e.g. use case may be represented by one of 3 variables depending on selected complexity level). Such high dimensional model of approximation will rather require applying more advanced method like mentioned-above GMDH (which supports more input variables).

In this paper we presented a new method for work effort estimation called SAMEE. On the basis of selected UML artifacts like use cases, actors, classes and references between them it builds estimation model using nonlinear polynomial regression. The advantage of our method is that it gives explicit expression describing dependency between work effort and number of analysis artifacts. It also easier adapt to particular development process than other methods because it does not require any specific technical or environmental parameters. Future work will concentrate on deeper validation of the proposed method. The method of work effort estimation will be verified for other information systems. Especially reduction of the model by setting proper *RejectThreshold* parameter value will be deeply considered.

We successfully implemented SAMEE as an extension to Enterprise Architect CASE tool and we used it for the real information system work effort estimation with satisfying results. We also outlined some possible improvements and modifications that can be done to obtain probably better results.

Acknowledgments

This work was supported by the European Union from the European Social Fund (grant agreement number: UDA-POKL.04.01.01-00-106/09).

References

1. B. Boehm, Ch. Abts, S. Chulan, *Software development cost estimation approaches – A survey*, Annals of Software Engineering 10 (2000), pp. 177-205.
2. G. Karner, *Use Case Points – Resource Estimation for Objectory Projects*, Objective Systems SF AB (copyright owned by Rational Software), 1993.
3. P.R. Symons, *Software Sizing and Estimating MK II FPA (Function Point Analysis)*, John Wiley & Sons, 1991.

4. P. Kruchten, *Rational Unified Process – An Introduction*, Addison–Wesley, 1999.
5. P. Mohagheghi, B. Anda, R. Conradi, *Effort Estimation of Use Cases for Incremental Large–Scale Software Development*, Software Engineering, 2005, pp. 303-311.
6. S. J. Farlow, *Self–organizing Methods in Modeling*, Statistics: A Series of Textbooks and Monographs, Dallas Texas, 1984.
7. UML tools for software development and modeling – Enterprise Architect UML modeling tool, http://www.sparxsystems.com.
8. Enterprise Architect Automation Interface Examples: Introduction, http://www. sparxsystems.com/resources/developers/autint.html.
9. Sparx Systems – Resources – Developers – UML Profiles, http://www. sparxsystems. com/resources/developers/uml profiles.html.
10. Enterprise Architect – Resources – Project Estimation using Use Case Metrics, http:// www.sparxsystems.com/resources/ucmetrics.html.
11. OpenUp, http://epf.eclipse.org/wikis/openup.
12. R. Haber, L. Keviczky, *Nonlinear System Identification – Input-Output Modeling Approach*, Springer 1999.

SAMEE – nieliniowa adaptacyjna metoda predykcji pracochłonności wytworzenia systemów informatycznych

Streszczenie

We wczesnych etapach tworzenia systemu informatycznego szczególnie trudnym zadaniem jest wstępne oszacowanie pracochłonności wytworzenia całego systemu, gdy pierwszy, wstępny, analityczny opis systemu jest znikomy, tzn. gdy dostępne są jednie uproszczone artefakty analityczne (w sensie dostępności tylko niektórych rodzajów artefaktów i niskiej szczegółowości ich opisu). W niniejszym artykule zaproponowano metodę SAMEE (ang. *Simple Adaptive Method for Effort Estimation*), tzn. prostą adaptacyjną metodę szacowania pracochłonności, opartą na UML-owych artefaktach analitycznych takich jak: cechy systemu, przypadki użycia aktorzy, klasy dziedzinowe, referencje pomiędzy klasami. Na podstawie znajomości danych historycznych dotyczących pracochłonności realizacji cech oraz związanych z tymi cechami artefaktami (przypadki użycia, aktorzy, klasy, referencje) metoda określa nieliniowy model estymacji pracochłonności wytworzenia cech. Wykorzystując regresję wielomianową, metoda pozwala na znalezienie jawnej zależności pomiędzy pracochłonnością wytworzenia cech systemu, a innymi, wymienionymi wcześniej

artefaktami. W szczególności metoda pozwala na wskazanie, który typ artefaktu ma najistotniejszy wpływ na estymowane pracochłonności cech. Metoda ma charakter adaptacyjny i lokalny, tzn. uzyskany model wynikowy zależny jest ściśle od organizacji procesu wytwórczego, zespołu produkcyjnego, przyjętych narzędzi wytwórczych.

Artykuł opisuje również prototypową implementację metody SAMEE w postaci tzw. programowej wtyczki (ang.plug-in) do popularnego narzędzia CASE – Enterprise Architect – wspomagającego analizę i projektowanie systemów informatycznych. Dodatkowo, dzięki mechanizmowy profili UML, odpowiednie artefakty (np. cechy) zostały wzbogacone o atrybuty wymagane do obsługi zaproponowanej metody estymacji. Taka integracja z narzędziem CASE pozwala myśleć o praktycznym zastosowaniu narzędzia do szacowania pracochłonności metodą SAMEE. Przykład konkretnego użycia SAMEE w narzędziu Enterprise Architect został zamieszczony w opracowaniu.

W artykule opisano również potencjalne kierunki rozwoju metody (np. zastosowanie szerszej bazy typów artefaktów analitycznych, czy wykorzystanie innego, bardziej złożonego, nieliniowego modelu estymacji).