C. Wohlin and A. Andrews, "Evaluation of Three Methods to Predict Project Success: A Case Study", Proceedings of International Conference on Product Focused Software Process Improvement (PROFES05), LNCS-series, Springer Verlag, Oulu, Finland, 2005.

Evaluation of Three Methods to Predict Project Success: A Case Study

Claes Wohlin

Dept. of Systems and Software Engineering, School of Engineering, Blekinge Institute of Technology, Box 520
SE-372 25 Ronneby, Sweden
+46 457 38 58 20
claes.wohlin@bth.se

Anneliese Amschler Andrews

School of Electrical Engineering and Computer Science
Washington State University
Pullman, WA 99164-2752 USA
+1 509 335 8656
aandrews@eecs.wsu.edu

Abstract. To increase the likelihood for software project success, it is important to be able to identify the drivers of success. This paper compares three methods to identify similar projects with the objective to predict project success. The hypothesis is that projects with similar characteristics are likely to have the same outcome in terms of success. Two of the methods are based on identifying similar projects using all available information. The first method of these aims at identifying the most similar project. The second method identifies a group of projects as most similar. Finally, the third method pinpoints some key characteristics to identify project similarity. Our measure of success for these identifications is whether project success for these projects identified as similar is the same. The comparison between methods is done in a case study with 46 projects with varying characteristics. The paper evaluates the performance of each method with regards to its ability to predict project success. The method using key drivers of project success is superior to the others in the case study. Thus, it is concluded that it is important for software developing organizations to identify its key project characteristics to improve its control over project success.

1. Introduction

There are many project characteristics that influence project success. If it is true that certain project characteristics lead to certain project outcomes, it becomes important to be able to identify similar projects, since they would likely lead to similar project outcomes. Nowadays when many software systems evolve over time and come in several releases, it becomes even more crucial to learn from past projects to increase the likelihood for success in future projects. Software organizations must understand which project characteristics are most important for them to ensure successful projects. This paper contributes with a comparison of three potential methods to use to predict project success.

Some examples of project characteristics include stability of requirements, knowledge of developers and management, inherent difficulty of the project,

techniques and tools used (and how appropriate they are), tightness of schedule and type of application, required levels of reliability, fault tolerance, security, use of development techniques including use of object-oriented programming, design patterns, extreme programming, etc. Success indicators include timeliness of delivery, quality of software, as well as more long term properties such as maintainability and evolvability.

Some of these project characteristics can be measured objectively, while others have to be measured subjectively. The same is true for project success variables. We focus here on subjective measurement of both project characteristics and project success as also is the case in for example [1]. The focus here on subjective measures is primarily due to availability in the case study presented below.

In order to predict project success reliably, it is first necessary to find a method to identify similar projects. Projects are similar, if they show similar project characteristics and/or similar project outcomes. This paper investigates three methods to identify similar projects:

1. Nearest neighbor. This method takes a set of projects, the base set, and computes for a new project the distance to each project in the base set. The new project is most similar to the project with the smallest distance to the new project.
2. Friends. This approach also uses a base set of projects as well as the new project and groups them based on Principal Components Analysis (PCA) [2]. The new project is most similar to the projects with which it has been grouped.
3. Key success driver analysis. Not all project characteristics are equally important in predicting project success. This analysis method reduces the full set of project characteristics to those that behave similarly to project success. This reduced set of project characteristics is then used to rank projects by success and failure. New projects are identified as potentially successful or not based on this classification.

We selected these three approaches, because they represent three very different approaches to similarity identification. The first method is focused on identification of the most similar project using a distance measure [3], which is a simple measure used in case-based reasoning. The second method identifies a set of similar projects using all available information. Finally, the third method is based on identification of key success drivers. Based on these key success drivers, the most similar projects are identified.

Other methods would have been possible to use as for example neural networks, clustering and so forth. However, the main objective is not to make an exhaustive comparison of methods. The objective is primarily to investigate whether is makes more sense to focus on a few key variables (in particular the key success drivers in the context of this paper) rather than all data available for different projects.

All three methods are evaluated with respect to their ability to predict project outcome (success or not). Since the impact of project characteristics on project outcomes varies over time (for example, use of the software factory concept [4] will

influence the effect of project characteristics over time), we need to take this into account. Our approach is to use a sliding window of projects for the base set. This means, that the oldest projects are removed from the base set as new ones are added to it. Given the high degree of innovation and change in software development, it is not likely that the base set will ever be very large. This makes it important that methods used for prediction are able to function with small amounts of data.

The paper is organized as follows. Section 2 describes related work with respect to identifying similar projects, predicting various aspects of project success based on measurement of objective and subjective variables, and their use for various types of predictions like effort estimation, risk estimation, in addition to estimating various success factors. Section 3 describes the case study data used to illustrate the approach. Section 4 specifies the approach and illustrates the approach on the case study data. Section 5 draws conclusions and points out further work.

2. Related Work

Most prediction work has been directed towards using quantitative data, for example, measures of size, person-hours spent, number of defects and so forth. However, subjective measures have only recently been analyzed more extensively in empirical software engineering, specifically in software project assessment and evaluation. Part of the problem stems from issues related to collecting trustworthy data [5]. On the other hand, subjective measures have been used successfully for effort estimation [6, 7], and risk management [8]. Expert judgment in estimation tasks has been discussed in [9].

More recently, subjective variables have been used to map project characteristics to project success. First in [1], subjective factors were evaluated alongside objective quantitative ones to evaluate both efficiency and success of software production for projects from the NASA-SEL database [10]. The paper identifies which successful projects were also efficient and determines primary (subjective) drivers of project success. In [11], the method for analyzing subjective project characteristics and success indicators is refined and discussed in-depth, and two case studies are presented. The primary success drivers amongst the project characteristics are identified and an agreement index is established that quantifies to which degree the project characteristics that were identified as primarily connected to project success are able to predict project success. The results identified that about one third of the successful projects could be predicted accurately. We believe that this is due to the limitations of the approach, specifically that projects are classified into two categories: upper half and lower half (both based on project characteristics). The

halves are denoted “good” and “bad” respectively although it is really up to each individual organization to judge where the limit between “good” projects and “bad” projects is. It is reasonable to assume that projects around the border between “good” and “bad” exhibit more uncertainty with respect to project outcome (success or failure). This could account for some of the misclassification. To circumvent this problem, an extension was proposed in [12], where a third class was introduced to try to avoid classifying projects close to the border between “good” and “bad”.

However, the work so far has been focused on development of a method for identification of key success drivers in software development. Thus, the research has been focused on developing one model and then extending and improving the model. Here, the main focus is on using the developed model for prediction and to evaluate the model in comparison with two other approaches using industrial data from 46 development projects.

Prediction models in software engineering have been directed towards several different areas. Researchers have also used a large variety of statistical methods and approaches. This includes regression analysis (linear, multiple linear regression, stepwise regression) [13,14], multivariate statistics [15], machine learning [16] including neural networks [14, 17], analogies [13,18] and so forth. Some of these methods, including for example neural networks, require a substantial amount of data for model building. To address this problem, researchers have looked at methods for prediction when having few data points. This includes, for example, the use of different decision making methods such as the analytic hierarchy process [19] which is applied to effort estimation in [20]. Case-based reasoning has also been used as a means for prediction, for example, in [14, 20].

3. Case Study Data

The data comes from the NASA-SEL database [21]. Further information about NASA-SEL can be found in [10]. In total, the database available to us contains data from more than 150 projects spanning five years. Of these, we selected 46 for analysis, based on completeness of project data recorded. These 46 projects represented a variety of types of systems, languages, and approaches for software development. The actual data is not intended to play any major role. The main objective is to compare three methods for prediction purposes and in particular the importance of using key characteristics in the prediction rather than as much information as is available.

The selection criterion of projects in the database was completeness of data for 27 project characteristics. The characteristics are measured on an ordinal scale (1-5)

using subjective judgment. A higher value denotes more of the quality ranked. The subjective evaluations rank the projects in terms of problem complexity, schedule constraints, nature of requirements, team ability, management performance, discipline, software quality, etc. Six success factors were measured per project. These six variables are aggregated to one variable in the evaluation here, since it does not affect the actual comparison. From an analysis point of view, it is unfortunate that the data is on an ordinal scale as seen below. However, this is the type of data that we as researchers many times face when collecting data from industrial environments. The actual variables are not crucial for the comparison of methods and hence the reader is referred to, for example, [11].

4. Analysis Method

The analysis method consists of three phases:

1. Identify a base set of projects to use in assessing new ones. This base set should describe the current spectrum of project capability of the organization. While one way is to simply take all projects finished by an organization to date, this might include old projects whose relationships between project characteristics and success indicators no longer reflects current project behavior. Reasons for changes include: changes in capability, environment, process, successful use of experience factory concepts, etc. Thus we propose to use a sliding window instead, since it is most likely that older projects no longer reflect new ones.

   For the case study, the first 20 projects were viewed as the initial base set, the initial experience base. This first base set was used to identify which projects in the base set were similar to which of the following 5 new projects (for the purpose of predicting project success). Then the base set was updated by removing the five oldest projects from the base set. Again, the new base set (projects 6-25) was used to identify similar projects for projects 26-30. Projects 11-30 were used to identify similar projects for projects 31-35; projects 16-35 were used to identify similar projects for projects 36-40, projects 21-40 were used to identify similar projects for projects 41-45 and finally projects 26-45 were used to identify similar projects for project 46.
2. Identification of similar projects. This paper investigates and compares three ways to identify similar projects. For each window and set of new projects identified in step one, Nearest Neighbor, Friends, and Key Success Driver are used to identify projects in the experience base that are similar to the new projects. The detailed analysis is described in the subsections below.

3. Evaluate prediction quality. New projects identified as similar (based on characteristics) with projects in the base set are assumed to show similar outcomes to those projects with respect to level of success. Since we know the level of success through the data in the database, it is possible to evaluate prediction accuracy. We use a diffusion matrix to illustrate correct predictions versus false positives and false negatives. In addition, we also use the kappa statistic as an agreement index. The agreement index is described in more detail in a software engineering context in [22] and briefly outlined below.

We selected to work with the average rather than the median, because it takes into consideration the effect of extreme or unusual values better and accentuates differences in data that the median would not show (if most project success indicators are quite similar, very little can be said about their differences). Projects are then ranked based on this average score. Similar to the original model [11], we consider a project to be successful, if it is in the top half of the projects ranked. It should be noted that all projects may be successful, but the upper half is more successful than the lower half. This is a somewhat simplistic view on success to illustrate the prediction methods. In a real case, it would be important to actually define success properly, for example the software was developed on time within budget with the correct functionality and the quality expected. Anyhow, here it is primarily a matter of relative comparison between the projects rather than distinguishing between successful project and failed projects. Projects in the top half are ranked green, in the bottom half, red. We have chosen not to use the extended model with three classes [12], since the main objective here is to compare three competing methods for prediction and the extended method tries to circumvent the prediction problem. Here, we would like to address the prediction difficulties.

New projects identified as being similar with green projects are predicted as green, while new projects identified as being similar with red projects are considered red. A diffusion matrix reports agreement between prediction using the experience base and the actual outcome (average score of success variable).

The following subsections describe the three methods for identification of similar project(s) and evaluate the ability of the methods to predict success.

4.1. Nearest Neighbor

In this method a distance measure is calculated for each new project from the projects in the experience base (the case study has 20 projects in the experience base for each iteration). The measure is a measure of dissimilarity (DS), since the most similar

project in the experience base is the project with the highest value of the measure below. The measure is for ordinal data computed as:

$$DS = \frac{1}{\sqrt{\sum_{i=1}^{n} (C_{1i} - C_{2i})^2}}$$

In the formulae, C_{ji} is the value of variable i for project j, where only two projects are considered. One project is the new project and then it is compared with all other 20 projects one at a time. This means that the nearest neighbor is the project with the largest value of *DS*. More information about the measure in a software engineering context can be obtained from [3].

The project in the experience base with the largest value for the distance measure is selected as the nearest neighbor. The nearest neighbors are listed in Table 1 for projects 21-46. The table shows that some projects end up as neighbors more often than others. Project 16 is one of those.

TABLE 1. Identification of Nearest Neighbour.

Project	21	22	23	24	25	26	27	28	29	30	31	32	33	34
Neighbor	2	16	16	16	16	16	12	25	23	15	16	27	26	27
Project	35	36	37	38	39	40	41	42	43	44	45	46		
Neighbor	20	21	35	30	26	24	30	35	35	35	26	39		

The outcome of the new project is predicted as the outcome of the nearest neighbor. The prediction is only concerned with whether the new project will be a success (green) or not (red). One could look more closely into the prediction of the actual value for the different success variables. However, we believe that at the stage we are making this prediction, it is sufficient to provide an indication of where the project is heading. When two or more projects have the same value for the distance measure, the newest project is selected as the nearest neighbor.

The results from using this method as a prediction method for all successive experience bases are shown in the Table 3. The method is not very accurate. Only eight projects out of 26 (about 31%) are actually predicted correctly (green-green or red-red). There are 5 false positives (19%) and 13 false negative (50%). An agreement

index is also calculated, which often is referred to as kappa statistic [23]. In software engineering, the kappa statistic has been applied to inter-rater agreement of process assessments [22]. For an introduction to the kappa statistics, please refer to these references. An interpretation of the kappa statistics can be found in Table 2.

TABLE 2. The Altman kappa scale.

Kappa statistics	< 0.20	0.21-0.40	0.41-0.60	0.61-0.80	0.81-1.00
Strength of agreement	Poor	Fair	Moderate	Good	Very good

The agreement index requires comparable scales for the two classifications, for example, as is the case when comparing the agreement between two raters using the same scale. The problem addressed here is different since we compare one prediction model with the actual outcome. This problem is inherently harder and hence the mapping in Table 2 is most likely too ambitious. Informally with respect to the scale, it is still reasonable that values below 0.20 are poor and values between 0.21-0.40 are fair. However, given that we compare predictions with outcome, we believe that 0.41-0.60 is good, 0.61-0.80 is very good and above 0.80 is utopia.

A random assignment of projects to the cells in Table 3 should on average result in an agreement index (kappa) of zero. Values below zero means that the prediction method performs worse than random assignment. This is the case for this particular method for our data set. In other words, this method fails quite dramatically in identifying an existing project to make a prediction of project success.

TABLE 3. Diffusion matrix for Nearest Neighbour prediction

Kappa = -0.26		**Outcome**	
		Green	**Red**
Prediction	**Green**	27, 32, 34, 38	22, 23, 24, 25, 26, 30, 31, 35, 36, 41, 42, 43, 44
	Red	21, 28, 29, 33, 37	39, 40, 45, 46

4.2 Friends

The second method is based on using Principal Component Analysis [2] to "package" similar projects (friends), i.e. projects whose characteristics seem to vary in the same way. The statistical package used for Principal Component Analysis (PCA) is StatView version 5.0.1. Basically, PCA tries to abstract variables (or in this case projects) that covariate into a higher level factor. The analysis method creates factors with loadings for each variable in each factor. A variable is here viewed to belong to the factor where the variable has the highest positive loading.

In the case study, all 27 project characteristics are measured on an ordinal 5 point Likert scale. While this type of analysis usually requires at least interval data, it is fairly robust with regards to using ordinal data. In addition, we only use it to identify projects that behave in a similar fashion with respect to the project characteristics. No other conclusions are drawn from the analysis.

The PCA is conducted for the 27 project characteristics to identify similar projects. Table 4 shows an example of the results of this analysis for the first experience base (projects 1-20) and one of the projects (project 21) in the first set of projects to be classified (projects 21-25). From Table 4, we see that project 21 has its highest loading in Factor 1. The other projects having this are projects 2, 3, 5, 7, 10, 12, 13 and 20. Thus, these eight projects are regarded as being the Friends of project 21.

The Friends approach extends the Nearest Neighbor approach from a single project to a set of close projects (hence the name Friends). Given that a new project is now considered to have the same level of success as its Friends, the average success value of the new project is defined as the average of the success values of its Friends. When the Friends analysis is unable to identify any similar project, all 20 projects are used to compute the success value of the new project. Given this average success value, the projects are ranked and divided into an upper (green) and lower (red) half. This provides the prediction for the new projects in terms of either being a green or red project.

The Friends for projects 21-46 are determined as illustrated in Table 4. Comparing Friends with Nearest Neighbors, in 18 of 26 cases the Nearest Neighbor is also a Friend. In three cases there is no Friend. This method is able to identify that there are no friends, while the previous method still selects the nearest neighbor although it may not be very close.

The results for all experience bases are shown Table 5. Table 5 shows that now 11 or 42% of the project outcomes are predicted correctly. There are 3 or 12% false positives and 12 or 46% false negatives. While the correct predictions have increased,

they are still no better than random, as indicated by a kappa statistic that is close to zero.

TABLE 4. PCA with loadings for projects 1-20 and project 21 as a new project.

Orthogonal Solution

	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5	Factor 6
P1	,172	,116	-,031	,300	,277	,666
P2	,845	,016	,068	,205	,233	,290
P3	,858	,273	-,014	,054	,124	,024
P4	-,156	,071	,067	,139	-,839	-,059
P5	,746	,117	-,222	,025	,249	,375
P6	,240	,925	,020	,017	,098	-,032
P7	,766	,381	,091	-,099	,024	-,168
P8	,215	,895	,033	,200	,074	,162
P9	,157	,463	,289	,100	,533	,376
P10	,936	,138	-,043	,098	-,033	-,035
P11	,001	,442	,295	-,063	,703	,069
P12	,761	-,314	,262	,210	,125	,306
P13	,849	,261	-,019	,044	,062	-,308
P14	-,079	-,165	,340	,244	,105	-,678
P15	,293	,196	-,284	,596	,464	-,194
P16	,562	,104	,607	,057	,058	-,028
P17	-,074	,085	,851	-,045	,043	-,196
P18	-,059	,058	-,056	,863	-,185	,080
P19	,484	,078	,384	,512	-,087	-,078
P20	,766	,007	,065	-,513	-,063	,076
P21	,749	-,174	,151	-,119	,279	,381

TABLE 5. Diffusion matrix for Friends prediction

Kappa = -0.03		**Outcome**	
		Green	**Red**
Prediction	**Green**	27, 32, 33, 34, 37, 38	22, 23, 25, 26, 30, 35, 36, 39, 42, 43, 44, 45
	Red	21, 28, 29	24, 31, 40, 41, 46

4.3 Key Success Drivers

The third method is based on identifying key project characteristics. These are characteristics that covariate with the success variable the most. This analysis is done using PCA with the project characteristics and the success variable.

In cases when the number of project variables exceeds the number of projects in the experience base, the original method developed in [11] has to be adapted. Otherwise the principal component analysis would lead to singularities. Our approach is to reduce the project variables to those having the highest correlations with the success variable. This can be done by setting either a threshold for the number of project variables to be included to a value *n* and taking the *n* project variables with the highest correlations, or by setting a threshold *c* for the correlation value.

In the case study we had 27 project variables, but only 20 projects in each experience base. We decided to set the threshold at c=0.4. The threshold is not crucial, since the main aim was to reduce the number of variables so that an analysis could be conducted. Moreover, the variables with the lowest correlation do not vary together with the success variable anyway. This resulted in different sets of variables to be investigated for the different sets of 20 projects, i.e. the experience bases obtained through the sliding window approach.

With this reduction of project variables, it was possible to perform a principal component analysis [2] and identify project characteristics grouped in the same factor as the success variable.

Initially nine characteristics are identified as being important for projects 1-20 and 6-25. This then changes to only two variables being identified as the key success drivers for the organization (projects 11-30, 16-35 and 21-40) The drivers identified are “requirements stability” and “application experience” of the development team. In the last experience base there is a change (projects 26-45). Four characteristics are included in the identified key success drivers. This shows that the success drivers may change over time, as the result of change or evolution in an organization.

Given that the key success drivers have been identified, it is possible to also store the average of the identified project characteristics, and hence rank them based on the average. For new projects, the average of the key success drivers was calculated and the outcome (in terms of red or green) was predicted after comparison with the experience base. The outcome of the prediction is presented in the diffusion matrix in Table 6.

The third method is best for this data set. With this method, 19 or 73% of the projects are predicted correctly. There are no more false positives, but 7 or 27% false negatives. The kappa statistics has increased to 0.51, which in Altman’s model would map to moderate agreement, see Table 2. However, we would like to regard it as

fairly good given the difficulty in creating accurate prediction models in software engineering.

TABLE 6. Diffusion matrix for Key Success Driver prediction

Kappa = 0.51		**Outcome**	
		Green	**Red**
Prediction	**Green**	21, 27, 28, 29, 32, 33, 34, 37, 38	26, 30, 31, 36, 41, 42, 45
	Red		22, 23, 24, 25, 35, 39, 40, 43, 44, 46

4.4 Discussion

Regardless of the method, this data set tends to have a relatively high number of projects classified as red, both in terms of correct classifications and terms of false negatives (red projects classified as green). This can be explained by the looking closer into the data [24], which unfortunately is impossible to fit into this paper. Anyhow, the high number of red projects among projects 21-46 can be explained when looking at the whole data set, both project characteristics and success variables. From the success variables, it is clear that in the beginning we have fairly low values and then comes a set of projects with very high scores (projects 10-20). The sliding window used in the analysis also means that the decreasing trend that we see from project 27 until project 46 results in that a majority of the new projects will turn out to be viewed as less successful (red), i.e. better projects were conducted in the past. A majority of projects is however predicted to be green since the new projects are viewed as being similar to older projects that are better and hence the level of success is overestimated. It is clear from looking closer into the data that the data behaves differently in terms of how it changes over time for project characteristics and success variables respectively. Ideally, the project characteristics and success variables should vary together as much as possible. The trends identified in the success variable are however not visible for the project characteristics, and hence we have a challenge. This challenge is partially addressed by the third method, where the project characteristics that vary together with the success variable are identified. A simple analysis of the data using descriptive statistics provides quite a lot of information that can be valuable to use in projects to come.

Due to the idiosyncrasies of this data set, it is impossible to generalize from it, except that it seems to make sense to have a method that identifies the key success drivers in terms of project characteristics. This is clearly an advantage with the third method. It provides support to identify key success drivers for organizations and use it for prediction purposes.

All three methods are independent of the actual measures defined by a specific organization. This makes them suitable for a wide variety of possible applications. Each organization can determine what they believe to be a suitable set of project characteristics to measure.

The third method also has the advantage that identifying key success drivers can help managers to plan and control success. The first two methods are aimed at prediction only. They cannot really be used for other purposes. Thus, the third method has two main advantages:

- It is the best predictor of project outcome.
- It provides management support to plan and control for success.

In addition, it should be noted that since we classify the data into two halves, some of the projects are viewed as being green although the actual average values of the success variables are decreasing. This may have been avoided if a threshold of success was set based on, for example, the first 20 projects. This is an area for future research, i.e. to look at absolute values for success rather than relative values as we have done here.

In other words, this study evaluates projects relative to other projects. When considering the data in more detail, it clearly indicates that a major improvement in terms of success was achieved after project 9. However, there is a tendency that the level of success is declining after that again and the level of success is closing in on what was seen in projects 1-9. This may be explained either with an actual decline in success or as a changed view on what is successful, i.e. our yardstick has changed.

5. Conclusions

This paper evaluated three approaches to identify similar projects. Projects are similar, if they (a) have similar project characteristics, and (b) have similar project outcomes related to levels of project success. We showed that grouping projects by key success drivers is superior to nearest neighbor and friends evaluation.

The approach also dealt with aging of projects for prediction purposes by using a sliding window of projects. An interesting question to investigate in the future is more sophisticated methods to determine the base set over time. For example, this paper assumed a sliding window of 20 projects. Is there a method to determine the “best”

size for such a window of base projects? Are there ways to identify projects that should be removed from the base set (such as “atypical” projects)? How would one identify those? Essentially this is a question of the effect of both changing project environments as well as the effect of using the experience factory to determine the base set of projects that represents the key descriptors for projects (both good and bad) for a development organization.

Although the identification of key success drivers was superior to the other two methods, it is still a challenge to identify similar projects to predict success of new projects. The scales (i.e. the perception of a certain score for a variable) may change over time and hence older projects may have to be re-evaluated to ensure that we use the same scale over time. Another challenge is to understand and have methods that are able to cope with trends and to help address decreasing trends in the data (as observed here). The trends in the data may be due to changes in the context of software development. In this case it is important to be able to use information like the one in this paper or similar information to manage software organizations.

References

[1] von Mayrhauser, A., Wohlin, C., Ohlsson, M. C.: Assessing and Understanding Efficiency and Success in Software Production. Empirical Software Engineering: An International Journal, Vol. 5, No. 2, pp. 125-154, 2000.

[2] Kachigan S. K.: Statistical Analysis – An Interdisciplinary Introduction to Univariate & Multivariate Methods. Radius Press, 1986.

[3] Shepperd, M.: Case-based Reasoning in Software Engineering. In A. Aurum, R. Jeffrey, C. Wohlin and M. Handzic (eds.): Managing Software Engineering Knowledge. Springer-Verlag, Heidelberg, Germany, 2003.

[4] Basili, V. R., Caldiera, G., Rombach, H. D.: Experience Factory. In J. J. Marciniak: Encyclopedia of Software Engineering. John Wiley & Sons, Inc., Hoboken, N.J., USA, 2002.

[5] Valett J. D.: The (Mis)use of Subjective Process Measures in Software Engineering. Proc. Software Engineering Workshop, NASA/Goddard Space Flight Center, Greenbelt, Maryland, USA, pp. 161-165, 1993.

[6] Gray A. R., MacDonell S. G., Shepperd M. J.: Factors Systematically Associated with Errors in Subjective Estimates of Software Development Effort: The Stability of Expert Judgement. Proc. of the Sixth Int. Software Metrics Symposium, Boca Raton, Florida, USA, pp. 216-227, 1999.

[7] Höst M., Wohlin C.: An Experimental Study of Individual Subjective Effort Estimations and Combinations of the Estimates. Proc. IEEE Int. Conf. on Software Engineering, Kyoto, Japan, pp. 332-339, 1998.

[8] Ropponen J., Lyytinen K.: Components of Software Development Risk: How to Address Them? A Project Manager Survey. IEEE Trans. in Software Engineering, Vol. 26, No. 2: 98-112, 2000.

[9] Hughes R.: Expert Judgement as an Estimation Method. Information and Software Technology, Vol. 38, pp. 67-75, 1996.

[10] Basili V., Zelkowitz M., McGarry F., Page J., Waligora S., Pajerski R.: SEL's Software Process-Improvement Program. IEEE Software, November pp. 83-87, 1995.
[11] Wohlin, C., Amschler Andrews, A.: Assessing Project Success using Subjective Evaluation Factors. Software Quality Journal, Vol. 9, No. 1, pp. 43-70, 2000.
[12] Wohlin C., von Mayrhauser A., Höst M., Regnell B.: Subjective Evaluation as a Tool for Learning from Software Project Success. Information and Software Technology, Vol. 42, No. 14: 983-992, 2000.
[13] Myrtveit, I., Stensrud E.: A Controlled Experiment to Assess the Benefits of Estimating with Analogy and Regression Models. IEEE Transactions on Software Engineering 25(4): pp. 510-525, 1999.
[14] Shepperd, M., Kadoda, G.: Comparing Software Prediction Techniques Using Simulation. IEEE Transactions on Software Engineering", Vol. 27, No. 11, pp. 1014-1022, 2001.
[15] Ohlsson, N., Zhao, M., Helander, M.: Application of Multivariate Analysis for Software Fault Prediction. Software Quality Journal, Vol. 7, No. 1, pp. 51-66, 1998.
[16] Mair, C., Kadoda, G., Lefley, M., Phalp, K., Schofield, C., Shepperd, M., Webster, S.: An Investigation of Machine Learning Based Prediction Systems. Journal of Systems and Software, Vol. 53, No. 1, pp. 23-29, 2000.
[17] Khoshgoftaar, T.M., Szabo, R.M.: Using Neural Networks to Predict Software Faults during Testing. IEEE Transaction on Reliability, Vol. 45 No. 3, pp. 456-462, 1996.
[18] Shepperd, M.J., Schofield C.: Estimating Software Project Effort Using Analogies. IEEE Transactions on Software Engineering 23(11): pp. 736-743, 1997.
[19] Saaty, T. L., Vargas, L. G.: Models, Methods, Concepts & Applications of the Analytic Hierarchy Process. Kluwer Academic Publishers, Dordrecht, the Netherlands, 2001.
[20] Shepperd, M., Cartwright, M.: Predicting with Sparse Data. IEEE Transactions on Software Engineering, Vol. 27, No. 11, pp. 987-998, 2001.
[21] NASA-SEL: Software Engineering Laboratory Database Organization and Users Guide, Revision 2. Goddard Space Flight Center, Greenbelt, MD, USA, NASA-SEL Series, SEL-89-201, 1992.
[22] El Emam, K., Wieczorek, I.: The Repeatability of Code Defect Classifications. Proc. the Ninth International Symposium on Software Reliability Engineering, pp. 322-333, 1998.
[23] Altman D.:Practical Statistics for Medical Research. Chapman-Hall, 1991.
[24] Wohlin, C., Amschler Andrews, A.: A Case Study Approach to Evaluation of Three Methods to Predict Project Success. Technical Report, Blekinge Institute of Technology, 2004, http://www.ipd.bth.se/cwo/TR-case.pdf.