D. Milicic and C. Wohlin, "Distribution Patterns of Effort Estimations", IEEE Conference Proceedings of Euromicro 2004, Track on Software Process and Product Improvement, pp. 422-429, Rennes, France, 2004.

Distribution Patterns of Effort Estimations

Drazen Milicic
RKS AB[*]
Amiralsgatan 17, SE-211 55
Malmö, Sweden
Drazen.Milicic@rks.se

[*] *Industrial Ph.D. student, located at RKS AB, in Software Engineering at the School of Engineering at Blekinge Institute of Technology in Sweden.*

Claes Wohlin
School of Engineering
Blekinge Institute of Technology
Box 520, SE-372 25
Ronneby, Sweden
Claes.Wohlin@bth.se

Abstract

Effort estimations within software development projects and the ability to work within these estimations are perhaps the single most important, and at the same time inadequately mastered, discipline for overall project success. This study examines some characteristics of accuracies in software development efforts and identifies patterns that can be used to increase the understanding of the effort estimation discipline as well as to improve the accuracy of effort estimations. The study complements current research by taking a more simplistic approach than usually found within mainstream research concerning effort estimations. It shows that there are useful patterns to be found as well as interesting causalities, usable to increase the understanding and effort estimation capability.

Keywords

effort estimation, estimation process, software project control, software project management

1 Introduction

Effort estimations within software development projects are still by many identified as a complex and critical, but elusive, aspect that software organizations yet have to master in an adequate manner [1]. At the same time, effort estimation is a central issue within software engineering as it represents a fundamental cornerstone for project managers to successfully manage software development projects in terms of input to project plans [2]. Much research has been performed within the effort estimation area to increase the estimation accuracy and to improve the success rate of software projects [3-5]. However, despite the research effort, the capacity of software projects to successfully work within agreed boundaries still is less than satisfactory; in 1998 only 26% of the software development projects were completed on time, on budget and with all the features/functionality originally specified [6]. Furthermore, many software professionals are still solely relying on expert judgment (i.e. experience with similar work, mostly using intuition) to estimate effort as this is the only feasible approach available to them. There are just too few effort estimation models available that are lean and easily adopted. Those models that are available, usually parametric models using historical data to formulate estimation algorithms [3-5], require a substantial amount of time, effort and money for collection, classification and categorization of historical data to tune the models to the specific organization's conditions. For many software developing organizations, the usefulness of such models has proven to be quite poor [7] and thus they do not rely on these tools [8]. As follows, parametric models are rarely used within software development projects, especially when the return on investment is so unpredictable; neither when applying historical data in very similar scenarios [9] nor with an abundance of historical data (with more than 500 programs and over 10 million lines of code available to predict the effort in new projects) [1].

The work performed within the study presented here aims to supplement the sparse work of lean and simple approaches [1;3;10;11] presented to guide software engineers in the effort estimations process and help project managers in managing cost predictions. The objective is to examine some effort estimation accuracy characteristics to gain additional understanding about the nature of effort estimations. The intention is that this should be achieved by studying effort estimation on the work package level[1], i.e. estimates for a development team rather than looking at software projects as a whole.

The remainder of this paper is structured as follows. Section 2 details the method used within the study to achieve trustworthy results and also covers important aspects such as research goals, design as well as threats to the study. Section 3 reveals the results of the study and presents some initial interpretations. Section 4 continues the analysis initially started in the previous section and provides deeper interpretations of the results. Section 5 discusses the finding and ties it all together with some final conclusions. Finally, some potential future work is presented in Section 6.

2 Method

2.1 Research questions

The overall research question to be answered within the study is whether there is a simple and recognizable distribution pattern of work package effort estimation accuracies, accumulated within a software development project, which can be used to predict software development project efforts. This overall question is further broken down into more specific research questions as follows:

1) What does the density function for the *accumulated project frequency distribution* of effort estimation accuracies look like?
2) Is there a causal relation between over- and underestimations of effort and the development *teams' average experience level*?
3) Is there a causal relation between over- and underestimations of effort and the estimated *size of a work package* [12]?
4) Is there a causal relation between over- and underestimations of effort and the *size of the development team*?

2.2 Case object

The software development project subjected to the empirical prediction case study was conducted in a commercial environment with an external customer financing the software development. The project was staffed primarily by software engineering consultants from the supplier's organization with some complementary technical experts from a subcontractor as well domain experts from the customer's organization. The project had 26 members, was conducted over a period of 12 months and used 9938 person hours. Over this time the project produced a total of 121 effort estimations of which 82 were included in the final results. All project members included in the study were professional software engineers highly motivated in doing a good job without exceeding the agreed boundaries.

The software development project used a technical environment based on the Oracle development platform. The clients were developed with help of 4GL tools (Oracle Forms Developer and Oracle Reports Developer) while the business logic primarily was written as function oriented store procedure components (code packages in Oracle PL/SQL).

Prior to the project, several of the software engineers were involved in creating a tailor made business application framework of standard functionality serving as a base platform upon which future customer specific applications would be built. As follows, the project subject to this study did not start building the application from scratch, but rather based the new application upon the pre-made application framework. Consequently, many of the software engineers were familiar with the environment (both from a technical and a domain perspective) already from start. The project, however, is still to be characterized as a software development project – and not a maintenance oriented project.

The software development process used can be characterized as an iterative approach with three overall functional increments [13-15]. 15% of the development effort was spent in phase 1, 49% in phase 2 and 36% in phase 3 (calculated on the total project effort also including work packages not used in the study results). The PPS project steering model [16] was used to manage and control the project from primarily a project management perspective.

2.3 Data collection

The software development project targeted for this empirical prediction case study was not aware of its participation in the study as all the data was collected post project termination. In fact, the researchers were never in any direct contact with the ongoing project. The data collection was a straightforward process as the data was naturally available within the project. All data was collected by the project for its own purpose and the fact that the project was submitted to a study did not affect the work. The data was made available to the researchers by the project manager.

An empirical prediction case study was designed to investigate the different aspects of effort estimations in relation to actual effort within a software development

[1] A work packages primarily specifies the work to be performed, the resources needed and the results to be expected [12].

project. The idea was to collect real life data through a non-experimental fixed design approach, identifying causalities and portraying a project effort prediction profile by tracking the following criterion variables [17]: effort estimation, actual effort (also work package size), experience level and size of teams.

Effort estimation: The estimation data was produced by means of expert judgment by either the project manager or other experts (not necessarily a participant of the software development team responsible for implementing the work package). The effort estimations were produced at an initial phase of the project, originating from just prior to the implementation phase [18], for the purpose of agreeing on costs and scope with the customer. The effort estimations were often reviewed by the customer and there was a continuous discussion regarding the reasonableness of the effort estimations - leaving little or no room for explicit buffers (neither on project level nor on a commercial level) in the made estimations. The estimations were made on a work package level and based on application requirements from the customer or, where necessary, the results from a supplementing design phase [18]. The application requirements were primarily functional requirements [18], specified on a relatively high level with few details, providing a general description of the functionality rather than a detailed specification.

Actual effort (also used as *work package size* metric): The actual effort was measured after completing the implementation phase and when the system was released for test [18]. The reporting of actual effort was a part of a natural project reporting process. As the effort was estimated on a team level, the outcome was also collected on a team level.

Experience level: Each software engineer has a documented competence profile within the company that describes the competence and experience of the software engineer. The competence profile is the source of information from where the number of relevant years within the software development business is extracted, i.e. the experience is measured in years of relevant work experience. The teams' experience level was calculated as the arithmetic average of the team members' experience. Experience data for four (out of 26) software engineers were not available through the competence profiles. For those software engineers, the project manager provided an assessment based on their displayed software engineering abilities. For all other software engineers a competence profile that described the competence and experience was available.

Size of teams: The team size was extracted measuring the number of individuals writing time on a work package. We chose a straightforward definition of the team size by including every software engineer that registered time on the work package – including experts providing temporary help.

The predictor variables [17] calculated to map the causality with the above presented criterion variables are:

- *Effort estimation accuracy* calculated by means of mean magnitude of relative error (MMRE) [19;20], according to: $(E_{Actual} - E_{Estimated})/(E_{Actual})$[2], where E is *Effort*.
- *Effort estimation accuracy frequency distribution* plotted as a histogram identifying the occurrences of MMRE within a 10% interval.

The results and the analysis in Sections 3 and 4 are focused on understanding the distribution of effort estimations, and to study the relationships between effort estimation accuracy and work package size, experience and team size.

2.4 Excluded data

Originally there were 121 samples collected but 29 of these were excluded as they originated from uncompleted work packages or work packages that in some way were dramatically changed compared to their original specification. An additional 10 samples with an inaccuracy of more than -117% were excluded later on as their MMRE indicated that the work package went through some substantial change of scope (e.g. which made them less relevant for this study due to that a simpler solution was chosen than initially intended, simplification of requirement etc.).

2.5 Threats to trustworthiness

Threats to trustworthiness address the vital issue of how trustworthy the empirical prediction case study is. The following threats have to be addressed: External validity, reliability and construct validity.

External validity or generalizability of findings [17] refers to the extent to which the findings are generally applicable outside the specifics of the study. Given that the project was performed by professional software engineers with a familiarity of the technical environment and the problem domain, the findings are probably most applicable for projects with similar settings. That is, if working in a new field of operation

---

[2] Some comments have been made about the unbalanced nature of MMRE as it penalizes overestimations more than underestimations [20]. For the purpose of this study it is not a major issue as the focus here primarily is the causality between MMRE and the criterion variables - not the MMRE value itself.

with bleeding edge technology the findings would probably be less relevant. The study is probably also more applicable to software development situations where there is an external party financing the project and where the commercial aspects are close upon all project members.

Reliability [17] addresses the stability of the study. That is, if the study was to be repeated, would the same results be obtained? The study, being an empirical prediction case study, may require some words about participant error – whether the data in the study is representative. The study object, the software development project, was chosen for its availability and information orderliness. However, even though the researchers did not select the project based on any other criteria than data availability, there is a risk that projects that have orderly data available also are the better managed projects. Hence, there is a risk that the object of the study represents the good example rather than an average project.

In line with the non intrusive approach, the study did not have any influence on which project members were selected for the project. The project composition was based solely on commercial grounds. Most of the project members have a history working within the problem domain and with the chosen technology. Hence, the project members were selected primarily for their capability of performing their jobs within the project.

All-in-all this portrays a situation that should be representative for projects composed of fairly experienced software engineers that are familiar with the work expected of them.

Construct validity (or in its simpler form: face validity) [17] deals with the assurance of measuring what is intended to be measured. That is, is it reasonable that the metrics are measuring what they aim to measure? One question for this study is whether the fact that the effort estimations were performed by one party and that the work was performed by another party has an effect on the estimations and the outcome? Even though this setup may be lacking the component of individual commitment [16], the estimations are still central to the project as the project boundaries are set based on the effort estimations. As the estimations need to be considered to meet the customer expectations of time and effort, they are central for the development teams as well as for the project as a whole.

Another related question is whether the estimations would have been different in any way if they were made by the development teams. There probably would have been some variation dependent on the team composition, but in general both the development teams and the estimators (project manager or other experts) have the same take on the estimations: they should be as accurate as possible while minimizing the risk for underestimations which may force the project to miss the budget. At the same time the customer is a counterbalance towards overestimations and even if the developers may have tried to perform more *safe* estimations, the customer would not have accepted large and expensive safety buffers.

3 Results

3.1 Distribution

The accumulated project effort estimation accuracy frequency distribution, depicted in Figure 1, indicates that the accumulated estimation accuracy is formed close to the familiar bell shaped curve of the normal distribution.

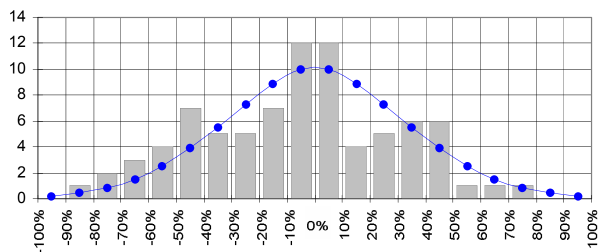

Figure 1: Accumulated project effort estimation accuracy frequency distribution. The columns represent the occurrences within the 10% MMRE interval. The dotted line shows a normally distributed frequency function (mean value = 0%) with the same area and same standard deviation as the columns.

The histogram compared to the normally distributed function indicates that the collected samples have more occurrences gathered around low MMREs as well as a slight tendency towards a non-symmetric negative skewed density function (with more overestimations than underestimations) – compared to the normally distributed density function. This is also confirmed by Table 1.

Table 1: Statistics for the effort data collected.

Mean value	-6%
Median	-2%
Standard deviation (σ)	36%

This is an interesting observation, since most studies report on finding a Rayleigh distribution[3]

[3] A Rayleigh distribution is a form of chi-squared distribution often used to approximate labor curves on software development projects. The Rayleigh distribution has the probability density function: $f(x) = \frac{x}{\beta^2} * e^{-\frac{x^2}{(2\beta)^2}}$, where $0 < x < \infty$ and $\beta > 0$.

[21;22], which means that the distribution is skewed towards underestimations. A potential explanation to the Rayleigh distribution in these studies is that engineers use the effort in some sense available to them, which means that overestimations occur more seldom than underestimations. In our case, the effort data is on a work package level and the software engineers move on to new work packages in the project, instead of gold plating the current work package.

Table 2 further illustrates the similarities between the project's effort estimating accuracies frequency distribution and a normally distributed density function by comparing the number of samples within the different standard deviation intervals.

Table 2: Occurrences of samples within 1, 2 and 3 standard deviations for the effort data collected.

# Standard deviation	# Samples	% of effort estimation occurrences	% of normally distributed occurrences
±1σ	52	63%	68.26%
±2σ	78	95%	95.44%
±3σ	82	100%	99.74%

3.2 Experience

An investigation of the relationship between experience level and effort estimation accuracy shows that there are differences associated with the teams' experience level. Studying the height of the bars shows that the bars with negative accuracy are in average higher. This indicates that more experienced engineers more often finish their work packages below the initial estimate.

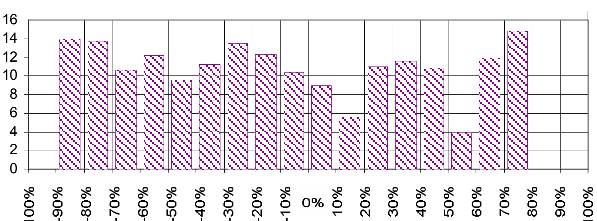

Figure 2: The average experience level of teams within each effort estimation accuracy interval.

The results are not surprising given that the estimates are based on the functional content and that it at the time of estimation wasn't finally decided who should actually implement the work package. However, it should be noted that some of the major positive deviations are made by development teams with long experience, see the rightmost bar in Figure 2.

3.3 Work package size

One question mark that might arise is to which extent the size of the work package affects the effort estimation accuracy. It is likely to find causality between the size of the work package and MMRE as large work packages also has a tendency to be specified in less detail (compared to smaller and more manageable work packages) and consequently also more roughly estimated. In line with the above reasoning, Figure 3 indicates a relationship between the effort estimation accuracy and the size of a work package.

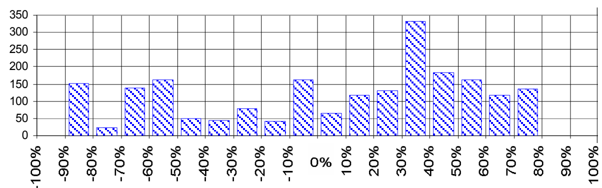

Figure 3: The average work package size within each effort estimation accuracy interval.

3.4 Team size

The size of the software development team might also be an issue that can affect the results of the study. That is, large teams might indicate that the work package is running with a higher burn rate without the corresponding increase in efficiency. Figure 4, however, does not confirm any clear and distinguishable relationship between the size of the team and effort estimation accuracy.

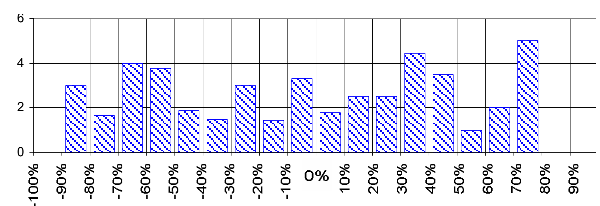

Figure 4: The average team size within each effort estimation accuracy interval.

4 Analysis and interpretations

The discussions in the previous section already indicated some answers to the research question regarding the form of the distribution of the effort estimation accuracy, and relationships between effort estimation accuracy and the other variables. However, to further strengthen the findings a statistical analysis of the results is performed.

First, it is evaluated whether the effort estimation accuracy follows a normal distribution or not. A Kolmogorov-Smirnov [22] test shows that the data is indeed normally distributed with a high significance level (ρ<0.0001). This finding is inline with the observation from Section 3.1.

In the further statistical analysis, the evaluation is done in two levels, i.e. whether the deviations are negative or positive. This means that the statistical tests investigated whether, for example, more experience results in negative or positive deviations or if no pattern can be identified.

For the three variables *experience level*, *work package size* and *size of team*, it is evaluated whether the variable is significantly related to the negative respectively positive deviations in effort estimation accuracy. The null hypotheses are that there is no relation between any of the variables and the deviations. The alternative hypotheses are that there is indeed a significant influence of these parameters. The analysis is conducted for one variable at the time and no interaction effects are studied between variables.

The three variables are evaluated to determine whether a parametric or non-parametric test should be used. The normality of the data is tested using the Kolmogorov-Smirnov test. It results in that parametric tests can be used for experience and work package size, while a non-parametric test is needed for team size. This means that t-tests [22] are used for experience and work package size and the Mann-Whitney test [22] is used for team size. A significance level of α=0.05 is chosen. The results are shown in Table 3.

Table 3: ρ-values for the statistical tests regarding experience, work package size and team size.

Variable	ρ-value	Significant
Experience	0.029	Yes
Work package size	0.056	No
Team size	0.12	No

The results show that there is a significant relation between experience and the sign of the effort estimation accuracy deviation. As observed already in the descriptive statistics, more experience in the teams results in that the teams more often come in lower than the estimated effort.

For the other two variables there are no significant results. However, the ρ-value for work package size is close to the significance level. The size of the work packages exceeding the estimated values are in average larger, although the results are not statistically significant.

In summary, the effort estimation accuracy is very close to a normal distribution. This partially contradicts studies indicating that engineers use the available effort even when they could have finished earlier [21]. This finding may be due to that the engineers in the project were involved in many work packages and they knew that finishing earlier in one work package may create a buffer in forthcoming work packages. This may explain the findings regarding the normal distribution.

In addition, the results indicate that it is more likely that a team with long experience comes in under the estimate. Moreover, the analysis almost shows that it is more likely that larger work packages exceed the initial estimate.

5 Discussion and conclusions

The study started out with a goal of answering the overall research question of whether there is a simple and recognizable distribution pattern of effort estimation accuracies accumulated within a software development project that can be used to predict software development project efforts. In essence, are there any good guiding principles that can help managers to assess the overall uncertainty and reduce the risks for running over the estimated effort?

The overall research question was detailed by studying the density function of the accumulated project frequency distribution of effort estimation accuracies. Given that effort estimations is a non-random and conscious activity where the software engineers use both intelligence and experience to get good estimates, the accuracy of such estimates can be expected to follow the normal distribution. However, the track record of software industry (with many late or failed projects) would indicate that there should be more underestimations than overestimation appearing within software development projects. As follows, the initial hypothesis of the study was to find an accumulated project effort estimation accuracy frequency distribution leaning towards a non-symmetric positively skewed density function. With better control as well as high discipline and experience, the density function would probably shift towards a more normally distributed bell shaped density function. However, thus far the capability and track record of software development projects indicate that a non-symmetric positively skewed density function was most likely to be expected.

The results of the empirical prediction case study clearly shows that accumulated over the whole project there is a normally distributed density function for the frequency distribution of effort estimation accuracies. The reason that the density function is not positively skewed is probably due to the fact that the project was situated in a mostly familiar surrounding, and that engineers could benefit themselves from finishing earlier than the initial estimate. Hence, the conclusion is that projects under these circumstances will tend to have a frequency distribution of effort estimation accuracies that are closer to the normally distributed bell shape than to the non-symmetric positive skewed shape most often seen in software development contexts. The study also indicates that the normal distribution can be used to approximate the probability for a work package's effort to end up within a wanted MMRE – depending project risk acceptance.

The hypothesis about causality between effort estimation accuracy and experience level is based on the logic that the estimates are done without knowing in detail which engineers that will perform the different work packages, and hence longer experience will help ensuring timely closure of the work package. As follows, senior software engineers will be more likely to finish below the initial estimate. This reasoning also stipulates that the task in itself has little to do with whether it is perceived as difficult or not – only the software engineer's familiarity with the task. Given that a software engineer has prior experience and/or clear and detailed information about the procedure on how to perform the task, all problems become equally difficult for that individual. Hence, a task that is perceived as difficult to estimate, is a task performed by a software engineer with no or little previous experience from such type of tasks, and/or a task where clear and detailed information about the implementation procedure is missing. As follows, the expectance was that experienced software teams would be more likely to complete their tasks under the estimated effort whereas inexperienced teams are more likely run over the estimations. The study gave significant results regarding the running over or under the initial estimate. It turned out that it is more likely that an experienced team comes in under the initial estimate.

The causality between positive or negative MMRE and the size of the work package as well as the size of the team was tracked primarily to eliminate these as influential factors within the study. The results, however, show that especially the size of the work package is potentially a factor that influences the ability to come in under the estimate. A potential explanation for this is that larger work packages are harder to survey and manage. Hence, the larger the package the more rough estimations and larger tasks will be found within the work package. Rough and large effort estimations will easily fail to notice key aspects within a work package. Consequently, the (almost significant) relation between work package size and positive/negative MMRE indicates that it is just as important to do a proper work break down of work packages so that the estimations are made on tangible and uncomplicated parts rather than larger software packages with fuzzy estimation logic.

In summary, it can be concluded that under certain circumstances the actual effort estimation accuracy follow a normal distribution. Moreover, it was found that whether a team comes in under or over the initial effort estimate depends on the experience of the team. The size of the work packages may also influence the situation, but this is an issue for further studies.

6 Future work

The analysis in Section 4 was primarily focused on whether the teams finish under or over the initial estimates. A possible extension is to evaluate whether experience level, work package size and size of team are related to the ability to finish close to the initial estimate. This would mean a stronger focus on the accuracy of the estimates. However, this is more interesting if the estimates and the actual work is conducted by the same person. In this case study, the estimates were derived by the project manager or experts in discussion with the customer. This means that the team conducting the work was given an estimate, and hence it is hard to evaluate accuracy formally since it depends heavily on the context the person making the estimates had in mind and the actual situation during development. There is a risk that an accuracy study would primarily evaluate the similarity between the context the person making the estimate had in mind and the actual development situation, rather than the true accuracy in the estimates. However, if conducting this type of case study online it is possible to discuss re-estimation of the initial estimate by the actual team and hence making an accuracy study more relevant.

An important issue in relation to effort estimation is the relation to other attributes such as precision in delivery time, quality (mainly in terms of defects) and functional content. It is always hard to evaluate whether the initial estimate really is for the results actually delivered. Unfortunately, this information was not available. To be able to fully evaluate effort estimation in relation to the other aspects, higher control over the scope and quality attributes of each work package are needed. This would enable a deeper analysis of the characteristics of each work package and allow for more detailed interpretations of the results. When investigating the accuracy of effort estimations it is important to assert that the estimations where made with the same scope in mind as the outcome is based on. Without this check there is always an uncertainty whether the accuracy metrics are relevant or not. The test phase would normally provide data in terms of faults, test effort etc. that could be used to assess some aspects of quality. However, at the time of data collection the test phase had not yet been completed and the data was not available. However, the objective should be to perform similar studies where also other aspects than effort estimations are under control.

To further complement the findings and generalizability of the paper's contents, a study based on a project not familiar with the business and technical environment would be valuable. Another valuable next step in the same direction as this study would be to make a similar study based on individual software engineers' estimates and efforts.

Acknowledgement

The study was jointly funded by RKS AB and The Knowledge Foundation in Sweden under a research grant for the project "Blekinge - Engineering Software Qualities (BESQ)" (www.bth.se/besq). Special thanks go to the project manager at RKS AB, Per-Arne Nordberg, who's forthcoming and contributions made this study possible.

References

[1] Humphrey, W. S., *Managing the software process,* Addison-Wesley, 1989.

[2] Abdel-Hamid, T. K., Sengupta, K., and Ronan, D., "Software project control: an experimental investigation of judgment with fallible information", *IEEE Transactions on Software Engineering*, vol. 19, no. 6, pp. 603-612, 1993.

[3] Boehm, B. W., Horowitz, E., Madachy, R., Reifer, D., Clark, B. K., Brown, A. W., Chulani, S., and Abts, C., *Software Cost Estimation with Cocomo II,* Prentice Hall PTR, 2000.

[4] Putnam, L. H. and Myers, W., *Measures for Excellence: Reliable Software on Time, within Budget,* Yourdon Press, 1992.

[5] Albrecht, A. J. and Gaffney, J. E., "Software Function, Line of Code and Development Effort Prediction: A Software Science Validation", *IEEE Transactions on Software Engineering*, no. SE-9 (6), pp. 639-647, 1983.

[6] The Standish Group International, "CHAOS: A Recipe for Success", The Standish Group International, Inc., 1999.

[7] van Genuchten, M. and Koolen, H., "On the use of software cost models", *Information and Management*, no. 1, pp. 37-44, Aug.1991.

[8] Wrigley, Clive and Dexter, Albert S.: "Software development estimation models: A review and critique", 1987.

[9] Ohlsson, M. C., Wohlin, C., and Regnell, B., "A project effort estimation study", *Information and Software Technology*, no. 14, pp. 831-839, Dec.1998.

[10] Höst, M. and Wohlin, C., "Subjective effort estimation experiment", *Information and Software Technology*, no. 11, pp. 755-762, 1997.

[11] Höst, Martin and Wohlin, Claes: "An Experimental study of individual subjective effort estimations and combinations of the estimates", *Proceedings - International Conference on Software Engineering,* IEEE Comp Soc, 1998.

[12] IEEE, Fairley, Richard. E., Glabas, John M., and Thayer, Richard H., "IEEE Std 1058-1998: IEEE standard for software project management plans", Inst. Electr. & Electron. Eng., 1998.

[13] Rational Software Inc., *RUP - Rational Unified Process*, www.rational.com, 2003.

[14] Kruchten, P., *The Rational Unified Process An Introduction - Second Edition,* Addison Wesley Longman, Inc., 2000.

[15] Gilb, T., *Principles Of Software Engineering Management,* Addison-Wesley Publishing Company, 1988.

[16] TietoEnator Corp., *PPS - A Guide to Professional Managment,* TietoEnator Corp., 2000.

[17] Robson, C., *Real world research: a resource for social scientists and practitioner-researchers,* Blackwell Publisher Ltd., 2002.

[18] IEEE, "IEEE Std 610.12-1990: IEEE standard glossary of software engineering terminology", Inst. Electr. & Electron. Eng., 1990.

[19] Conte, S. D., Dunsmore, H. E., and Shen, V. Y., *Software engineering metrics and models,* 1986.

[20] Shepperd, M. and Schofield, C., "Estimating software project effort using analogies", *IEEE Transactions on Software Engineering*, vol. 23, no. 11, pp. 736-743, 1997.

[21] Putnam, L. H., "Example of an early sizing, cost and schedule estimate for an application software system", *Computer Software and Applications Conference, 1978.COMPSAC '78.The IEEE Computer Society's Second International*, pp. 827-832, 1978.

[22] Everitt, B. S., *The Cambridge dictionary of statistics,* Cambridge University Press, 2002.