2011 International Symposium on Empirical Software Engineering and Measurement

Predicting Development Effort from User Stories

Pekka Abrahamsson, Ilenia Fronza,
Raimund Moser, Jelena Vlasenko
Free University of Bolzano-Bozen
Piazza Domenicani – Domenikanerplatz, 3
I-39100 Bolzano – Bozen, Italy
{Pekka.Abrahamsson, Ilenia.Fronza,
Raimund.Moser}@unibz.it, Jelena.Vlasenko@stud-inf.unibz.it

Witold Pedrycz
University of Alberta
9107 - 116 Street, Edmonton
Alberta, Canada T6G 2V4
Witold.Pedrycz@ee.ualberta.ca

***Abstract*—In this paper, we propose a method for predicting development effort based on user stories. Such approach is well suited for Agile software projects where requirements are developed along with the project and only sketched in a rough manner. We apply the proposed method to two industrial Agile software projects of very different size and structure. We show that such effort estimation works reasonably well if user stories are written in a structured way.**

Keywords- agile methods; effort prediction; user stories.

I. Introduction

Underestimating software development effort can have detrimental effects on business reputation, competitiveness, and performance. Overestimations can result in poor resource allocation and missed opportunities [8, 24]. Therefore, a large number of data driven techniques for effort estimation have been reported in the literature [7, 15, 17]; an overview of these techniques, their applications in industry, and their drawbacks regarding accuracy and applicability can be found in [18]. For Agile methods, effort estimation becomes harder, as these methods usually do not come with detailed requirements specification, analysis, and design documents before the start of development [1, 9, 14, 16, 21, 23].

To address these issues we propose a new method for Agile effort prediction, which is based only on predictors automatically extracted from user stories, i.e., one of the most prominent ways to specify requirements in Agile software projects [11]. Given a set of user stories, first, we define a set of predictors that can be extracted automatically from a user story. We do not require any particular structure of the format or content of a user story. Second, we build models and use them to predict the implementation effort of a user story. We compare the results both with our previous work [23], i.e., estimation based on design metrics, and with developers' subjective estimates.

The strongest point of our approach is that on one hand it addresses the problem of collecting predictor variables in Agile development, and on the other hand it fits naturally into Agile development practices as it predicts implementation effort of a single iteration, defined by a set of user stories. Moreover, data collection does not require much effort: data are available once user stories have been written, i.e., after a first planning phase, or at least before each single development iteration.

The proposed method can be applied for prioritizing requirements – knowing implementation effort per requirement is essential for planning and prioritization. Moreover, our approach does not depend on subjective knowledge; therefore, it might be useful for novice developers as an additional, independent estimation tool to reduce the risk of estimation.

The remainder of the paper is organized as follows. In Section 2, we present related work. Section 3 describes our method. In Section 4, we report on the results for effort prediction for two case studies. In Section 5, we discuss conclusions and limitations of this research.

II. Related Work

[2] examines the impact of object-oriented design metrics on development effort in two industrial projects using an Agile approach and in one industrial project using a traditional, long-cycled framework evolution process. Object-oriented design metrics are shown to be correlated with effort for refactoring and error-fix only in the Agile, short-cycled process.

The most common approach for effort prediction in Agile environments is subjective estimation [23]. Although such method is easy to apply, estimates can be highly biased [16]. Ideas for iterative project planning along with iterative monitoring and prediction of project resources have been proposed [6, 12, 20]. However, these works do not focus on specific issues proposals for their realization in the industry.

Some Agile projects use UML diagrams (in particular the "use case" diagram) for describing the functionality of a software system. For such cases, [21] demonstrates that it is possible to apply such methods in large industrial projects and for incremental development. [3] has reported similar results for use cases; well structured and detailed use cases are shown to provide higher confidence and accuracy for effort prediction models. [9] proposes to use more detailed UML diagrams, such as activity and class diagrams. While such methods might provide more accurate estimates, they are of limited use when dealing with Agile processes, as usually such detailed information is not available.

978-0-7695-4604-9/11 $26.00 © 2011 IEEE
DOI 10.1109/ESEM.2011.58

In our previous work [23] we proposed an incremental effort prediction model targeted to iterative development: it is based on the idea of refining a prediction model after each development iteration, and predicting the development effort for the next iteration. Such model outperforms traditional models as it is able to adapt to changes during project evolution and uses actual project data. However, one challenge of our previous proposal is grounded in the fact that it is based on the set of object-oriented design metrics defined by [10]. Thus, it is applicable only in those Agile projects where such information is available at the beginning of a new development iteration. Starting from our previous work, we continued our research and came up with the idea of using only information that is available for most Agile projects, namely user stories or any kind of informal textual description of the functionality of the software system. We use predictors, which are extracted from such short text documents, in order to overcome the limitations of traditional metrics in Agile environments.

III. Modelling Effort Prediction

Figure 1 presents a schematic view of the proposed effort estimation model. Starting from a set of completed user stories, first, we extract the predictors and we use them to train the model. Then, we use the model to predict the effort for new user stories.

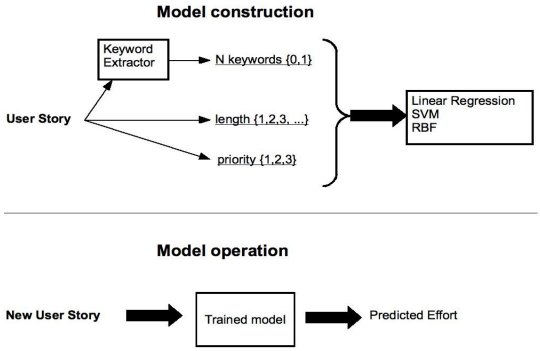

Figure 1. Effort prediction model.

We extract the following predictors from the user stories:

- *Number of characters.* This value may reflect the development effort, as developers may need to use longer texts to describe complex functionality.
- *Presence of keywords.* We selected the 15 most frequent terms together with their frequencies [19, 25]; we chose the value of 15 by testing models with different numbers of keywords and selecting the best value. Afterwards, we checked the presence of the keywords in the user stories and stored such findings as binary variable.
- *Priority.* The team works on the user stories in priority order (i.e., importance to the customer). An ordinal scale (usually, from 1 to 3) is used to represent priority, 1 indicating the highest priority. Priority may be related with development effort.

We consider only algorithmic models that can be constructed automatically from quantitative data: regression models [18], neural networks, and Support Vector Machines (SVM) for regression. We use the so-called leave-one-out (LOOV) cross validation procedure [5], and common criteria for evaluation of cost estimation models in Software Engineering: the Magnitude of Relative Error (MRE), the prediction at level k (PRED(k)), the Magnitude of Error Relative to the estimate (MER), and the Standard Deviation (SD) [15]. Moreover, we compare the prediction results with those achieved in our previous works [23] and with the developers' subjective estimates.

IV. Case Studies

In *case study A* we analyzed 1325 user stories of a large Italian company (that prefers to remain anonymous). The team was an Agile team composed of 17 developers, using a customized version of Extreme Programming (XP). They use weekly iterations, pair programming [13], user stories, and the test-first approach. The team has been using XP for more than two years previously to our study. The members of the team were not aware of the actual purpose of our study.

The single projects are very similar, as they use the same technology, have the same application domain and developers, and are strongly related to each other. Therefore, we have treated them as one big project in our analysis.

Table 1 shows the descriptive statistics of the data set.

TABLE I. Descriptive statistics of case study A.

Case study A - 1325 User stories			
Priority	***1***	***2***	***3***
	12%	87.7%	0.3%
Continuous variables	***Mean***	***Std-Dev***	***Min/Max***
Effective effort*	17	24	1/312
Estimated effort*	16	21	1/230
Size**	62	34	3/254
Keywords (English translation)	***Present***	***Absent***	***% Present***
gestion (management)	76	1249	5.7
file	76	1249	5.7
test	88	1237	6.6
creazion (creation)	49	1276	3.7
report	64	1261	4.8
tool	74	1251	5.6
possibilita (possibility)	37	1288	2.8
dati (data)	48	1277	3.6
modifica (modification)	30	1295	2.3
visualizzazion (visualization)	31	1294	2.3
configurazion (configuration)	37	1288	2.8
build	53	1272	4.0
visualizzar (to visualize)	26	1299	1.9
time	67	1258	5.1
canali (channels)	27	1298	2.0
* In story points: one point corresponds to about 25 minutes of uninterrupted work; ** In text characters.			

Developers' subjective estimation (obtained either by the subjective rating or by a structured group process similar to the Planning Poker [11]) is surprisingly accurate. We explain this by the fact that all developers are very experienced and have worked for a long time in industry on similar problems.

The text of most user stories is very short, usually containing only a few words.

Case study B concerns a commercial software project developed at a non-profit-making research organization. The aim of the project was to create a production monitoring application for mobile devices. The development process followed a tailored version of the XP practices [4, 22, 26]: two pairs of programmers worked for a total of eight weeks. The practices of refactoring and test-driven development have been also adopted. Table 2 reports the descriptive statistics of case study B. 13 user stories were developed for the Agile project under scrutiny. User stories have been written in a structured way and with a higher level of detail than in case study A.

The subjective estimates of developers are not as accurate as in case study A. We explain this by the fact that developers are less experienced and use for the first time the XP method. Since we do not have information about the priority of a user story we cannot include it into our set of predictors. Thus, for case study B the predictors are: presence/absence of keywords and number of characters.

TABLE II. DESCRIPTIVE STATISTICS OF CASE STUDY B.

Case study B - 13 User stories			
Continuous variables	***Mean***	***Std-Dev***	***Min/Max***
Effective effort*	12.7	13.5	3/47
Estimated effort*	16	9.8	3/35
Size**	316	327.7	16/1246
Keywords	***Present***	***Absent***	***% Present***
screen	8	5	61.5
user	6	7	46.1
data	9	4	69.2
item	4	9	30.7
capac	2	11	15.3
show	8	5	61.5
block	2	11	15.3
sale	3	10	23.0
graph	2	11	15.3
menu	4	9	30.7
set	3	10	23.0
order	5	8	38.4
level	4	9	30.7
chang	3	10	23.0
select	2	11	15.3
* In hours; ** In text characters.			

Table 3 shows the results of the models in the two case studies. The training cross validation error in case study A is very high, indicating that none of the 4 models will be able to predict development effort of new user stories with the required accuracy (a relative error of less than 30%). Also the error does not vary too much across different models. Thus, we conclude that the user stories do not contain enough information for training any useful prediction models. Case study B provides better results: both the Support Vector regression and a simple, stepwise linear model provide similar cross validation errors, a Relative Root Squared error of about 44% and a correlation coefficient of about 0.88.

Table 4 reports the evaluation of the predictions. Since stepwise regression performs as good as more sophisticated machine learners, and being the simplest model, we use it for prediction for both case studies. For case study A we train the model using 66% of all data and predict the remaining 34% of user stories. For case study B we have only 13 data points; thus we perform leave-one-out prediction, i.e., we train the model on 12 data points, predict the remaining one, and repeat this for each single data point.

TABLE III. RESULTS OF MODEL TRAINING.

Model	**LOOV-RMS**	**LOOV-RRS**	**Correlation**
Case study A			
LR	24.2	100.3%	0.07
RLR	26.0	108.0%	0.03
RBF	24.1	100.2%	-0.01
SVR	25.2	104.7%	0.10
Case study B			
LR	6.2	44.1%	0.88
RLR	14.2	101.1%	-0.11
RBF	10.1	72.4%	0.62
SVR	6.1	43.6%	0.88

TABLE IV. PREDICTION RESULTS.

Study	**mMRE**	**PRED(25%)**	**mMER**
Outcome of the linear model			
A	90%	17%	67%
B	66%	31%	47%
Subjective estimates made by developers			
A	28%	51%	25%
B	38%	23%	42%
*In story points; ** In hours.			

In case study A developers' subjective estimates work better than our approach. We explain this by the fact that user stories have been written in a very informal way and are very short. Moreover, developers are experienced and able to estimate implementation effort of user stories very well. For case study B the accuracy of the model lies in the range of subjective estimates of developers. Thus, it can help inexperienced developers to validate their opinion or propose a first estimate. Overall, it seems that well-written user stories can be used effectively for effort estimation.

The developers of case study B are used to work in a structured way within a research setting. Thus, they write user stories in a concise way, which provide enough information to train and apply successfully a prediction model using only keywords and length as predictors. This is not the case for study A where developers write user stories in a sloppy way. The difference could be explained by the fact that the industrial developers of study A are under much more pressure and have to meet strict deadlines; therefore, they neglect activities, which do not directly contribute to the final product, such as writing good user stories. It would be interesting to analyze whether it would pay for them to invest more time for writing well-structured user stories both to save implementation effort and to lower the risk of wrong effort planning.

In our previous work [23] we proposed iterative models based on design metrics as predictors. We found that such model provides an mMRE of less than 30% after a few development iterations. This is clearly better than we have achieved in this work. However, [23] is based on the assumption that design metrics can be estimated accurately from informal design documents such as user stories or CRC cards. We have seen that without this assumption the

performance of models based on design metrics might be very poor. In light of this, the approach discussed in this work offers many advantages: 1) it does not require any estimation of design or size metrics at project start; 2) assuming well written user stories the accuracy is as good as and probably better (with a large dataset) than subjective estimates; 3) it can be applied easily as it fits into traditional planning activities before starting a new iteration in Agile processes.

V. Limitations and Future Work

This work offers a first step towards understanding how to develop an Agile effort prediction model based on user stories. It is needless to say that a significant number of replications is needed to consolidate and generalize the findings of this study. As regards the internal validity of this work we have to be aware that the datasets are of limited size (in particular in case study B): any statistical conclusions from regression analysis have to be considered carefully. Another threat to internal validity arises from the fact that in case study A developers wrote user stories both in Italian and English. The performances of the tool for keyword extraction could depend on the language. We plan to analyze the impact of different languages on the accuracy of our models. A possible threat to the conclusion validity is our particular choice of predictor variables; more sophisticated data, as semantic text analysis, might provide better results. In this work we did not include such information as we aim at proposing a very simple and easy to build/apply model, inline with Agile principles. Each author's own style in writing user stories and the use of particular words related to different projects could be a limitation of our approach. Anyway, this study addresses this issue considering different developers working on different projects.

It could be interesting to build models in an iterative way, adding the effort of similar, already implemented user stories as predictor. This could provide more precise predictions as the project evolves and more information becomes available.

Acknowledgments

We acknowledge gratefully the support of the Free University of Bolzano/Bozen, of the Province of South Tyrol, of the Italian Minister of Education, University, and Research via the Italian Fund for Basic Research (Project Art-Deco). We also thank the anonymous company for supporting our study.

References

[1] P. Abrahamsson, R. Moser, W. Pedrycz, A. Sillitti, and G. Succi, "Effort Prediction in Iterative Software Development Processes – Incremental Versus Global Prediction Models," Proc. 1st International Symposium on Empirical Software Engineering and Measurement, 2007.

[2] M. Alshayeb and W. Li, "An Empirical Validation of Object-Oriented Metrics in Two Different Iterative Software Processes," Transactions on Software Engineering, 29, 11, pp. 1043-1048, 2003.

[3] B.C.D. Anda, H. Dreiem, D.I.K. Sjøberg, and M. Jørgensen, "Estimating Software Development Effort Based on Use Cases - Experiences from Industry," Proc. 4th International Conference on the Unified Modeling Language, 2001, pp. 487-502.

[4] K. Beck, Extreme Programming Explained: Embrace Change. Addison-Wesley, 2000.

[5] C.M. Bishop, Neural Networks for Pattern Recognition. Oxford University Press, 1994.

[6] K. Bittner and I. Spence, Managing Iterative Software Development Projects. Addison-Wesley Professional, 2006.

[7] B.W. Boehm, C. Abts and S. Chulani, "Software development cost estimation approaches - A survey," Annals of Software Engineering, 10, pp. 177-205, 2000

[8] L.C. Briand and I. Wieczorek, "Resource Modeling in Software Engineering," In Second edition of the Encyclopedia of Software Engineering, Wiley, 2002.

[9] M. Carbone and G. Santucci, "Fast&Serious: a UML based metric for effort estimation," Proc. 6th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering, 2002, pp. 12-23.

[10] S. Chidamber and C.F. Kemerer, "A metrics suite for object-oriented design," Transactions on Software Engineering, 20, 6, pp. 476-493, 1994.

[11] M. Cohn, User stories applied: for Agile software development. Addison-Wesley, 2004.

[12] M. Cohn, Agile Estimating and Planning. Prentice Hall PTR, 2005.

[13] I. Coman, A. Sillitti, and G. Succi, "Investigating the Usefulness of Pair-Programming in a Mature Agile Team," Proc. 9th International Conference on eXtreme Programming and Agile Processes in Software Engineering, 2008.

[14] I. Coman and A. Sillitti, "Automated Identification of Tasks in Development Sessions," Proc. 16th IEEE International Conference on Program Comprehension, 2008.

[15] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit, "A Simulation Study of the Model Evaluation Criterion MMRE," Transactions on Software Engineering, 29, 11, pp. 985-995, 2003.

[16] N.C. Haugen, "An Empirical Study of Using Planning Poker for User Story Estimation," Proc. AGILE 2006 Conference, 2006, pp. 23-34.

[17] F. Heemstra, "Software Cost Estimation," Information and Software Technology, 34, 10, pp. 627-639, 1992.

[18] M. Jørgensen and M. Shepperd, "A Systematic Review of Software Development Cost Estimation Studies Document Actions," Transactions on Software Engineering, 33, 1, pp. 33-53, 2007.

[19] Y. Matsuo and M. Ishizuka, "Keyword extraction from a single document using word co-occurrence statistical information," International Journal on Artificial Intelligence Tools, 13, 1, pp. 157–169, 2004.

[20] A. McDonald and R. Welland, Agile Web Engineering Process. Technical Report TR-2001-98, University of Glasgow, 2001.

[21] P. Mohagheghi, B. Anda, and R. Conradi, "Effort estimation of use cases for incremental large-scale software development," Proc. 27th International Conference on Software Engineering, 2005, pp. 303-311.

[22] R. Moser, P. Abrahamsson, W. Pedrycz, A. Sillitti, and G. Succi, "A case study on the impact of refactoring on quality and productivity in an agile team," Proc. 2nd IFIP Central and East European Conference on Software Engineering Techniques, 2007.

[23] R. Moser, W. Pedrycz, and G. Succi, "Incremental Effort Prediction Models in Agile Development using Radial Basis Functions," Proc. International Conference on Software Engineering & Knowledge Engineering, 2007, pp. 519-522.

[24] J.W. Paulson, G. Succi, and A. Eberlein, "An empirical study of open-source and closed-source software products," Transactions on Software Engineering, 30, 4, pp. 246-256, 2004.

[25] M.F. Porter, "An algorithm for suffix stripping," *Program*, 14, 3, pp. 130–137, 1980.

[26] A. Sillitti and G. Succi, "Requirements Engineering for Agile Methods," In Engineering and Managing Software Requirements, Springer, 2005.