

# Preliminary Results of two Academic Case Studies on Cost Estimation of Changes to Requirements

Annabella Loconsole, Jürgen Börstler

## Abstract

*Requirements management measures can help us to control changing software requirements and estimate the costs of changing requirements. This paper describes two small case studies, performed in the context of a team-project based software development course. In the first study, we compared intuitive cost estimations of changes to requirements to estimations based on historical data. In the second one, we studied whether even simple tools to support impact analysis affect the accuracy of cost estimations. Although the data we collected in these studies is not suitable for statistical analysis, we can present some interesting results and lessons learned. Our results suggest that estimations based on an impact analysis checklist are better than the intuitive estimations obtained in study one. However, study two is not yet completed therefore we cannot draw further conclusions.*

## 1. Introduction

Carefully developed software requirements are a key issue for project success [21]. The cost of correcting an error after system delivery is orders of magnitude higher than the cost of correcting a similar error during the requirements analysis phase [17]. Since requirements change during software development, it is important to control those changes to be able anticipate and respond to change requests [19]. Requirements development is a learning process rather than a gathering process. Therefore, it is naïve to believe that we can specify exactly what a customer wants at the beginning of a project. The best we can do is to carefully monitor and control all requirements throughout the software life cycle.

Software measurement can help us in providing guidance to the requirements management activities by quantifying changes to requirements and in predicting the costs related to changes. Numerous software measures for the requirements management activities have been proposed in the literature (see [4], [18], [10], [20], [8]). However, few empirical studies [1], [11], [12], have been performed to demonstrate the effectiveness of these measures. In our previous work [13] we analysed the key practices defined in the Requirements Management Key Process Area (KPA) of the SW-CMM [16]. By means of the Goal Question Metrics (GQM) paradigm [2] we defined a total of 38 measures.

In this paper, we will present partial results of an ongoing case study and compare it with a previous one. The motivations to perform these studies and compare them are manifold. Our first goal is to investigate whether cost estimation of changes to requirements performed using historical data are better than intuitive cost estimations. The second goal is to investigate whether cost estimation of changes to requirements based on detailed impact analysis are better than intuitive cost estimations. Another goal is to compare the case studies and show the methodological improvements accomplished by using the lessons learned from the first case study. Furthermore, we want to contribute to the lack of empiricism in the area of requirement management measures.

Study one did not reveal sufficient data to draw statistically significant conclusions because the teams did not have changing requirements in the second iteration therefore they were not able to make predictions. At the time of writing, study two is not yet completed therefore we have only some preliminary results. Among the results, we can say that one of

the teams who collected data based on impact analysis checklist had good estimations. However, the data collected is still too little to be able to deduct some conclusions.

We roughly followed the experiment process proposed by [22]. However, the empirical work described here are case studies, because we have had little control over the variables involved in the studies.

The remainder of this paper is organised as follows: section 2 presents an overview of the studies, in section 3 we describe the context of the studies. Section 5 contains the risks of the studies. Hypothesis, plans and measures are described in section 6. Sections 7 and 8 describe the results for study one and two respectively. In section 9 we compare the two studies. Finally some conclusions and future work are presented in section 10.

## 2. Case Studies Overview

Both studies were conducted with student teams in the context of the course Object-Oriented Software Development (OOSD) held at Umeå University. In fall 2002 there were twenty students in this course divided into four teams. According to our plans, all the four teams were required to participate to the study. However, only two teams collected data on a regular basis. In fall 2003 there were twenty-six students divided into five groups. All teams are participating in the study.

The students measured their requirements every week and filled in forms with the results of the measurement activities on their projects. In both studies, the teams described their functional requirements following a specific format [14]. Requirements were measured regularly and the results submitted weekly using predefined forms. The projects had a schedule of twenty weeks. The software development process followed in the course was a two iterations process where each iteration was eight weeks long. Each team received a form with a list of measures to be collected.

In study one, only two teams performed the data collection on a regular basis (team A and team B). Team A made intuitive predictions of costs of changes to requirements during the first and second iterations. The plan for team B was to make intuitive predictions in the first iteration. In the second iteration, their predictions should be based on historical data, collected during the first iteration. However, team B did not record any changes to requirements in the second iteration and therefore did not provide the required predictions.

In study two, we decided to not have a distinction between iterations, since twenty weeks was too short a period to collect sensible historical project data. Two teams made intuitive cost estimations while the other three teams based their estimations on a checklist-based impact analysis. At the time of writing, one team is collecting data on a regular and precise way, the data collected by the other teams are not yet complete.

## 3. Definition of the Case Studies

By applying the GQM template [2] for the goals definition, we obtained the following for study one:

*Analyse:* Requirements Management process area in the OOSD course.

*With the Purpose of:* Evaluate the impact of software measurement in prediction of cost of changes to requirements.

*Quality focus:* Effectiveness of the use of historical data in predictions of cost of changes to requirements compared to intuitive predictions.

*Perspective:* Academy.

*Context:* The study is conducted in an object-oriented software development course. The context of the study will be described below in more details.

Study two differs from the first only in the purpose and quality focus:

*Analyse:* Requirements Management process area in the OOSD course.

*Purpose of:* evaluating the cost model based on detailed impact analysis of cost of changes to requirements.

*Quality focus:* accuracy of the cost model.

*Perspective:* Academy.

*Context:* The study is conducted in team-project object-oriented software development course.

#### **4. Case Studies Context and Environment**

The context of the studies was a team-project software development course (OOSD), focusing on object-oriented approaches. Methods, languages, and tools that support these approaches were discussed and applied. Projects had a schedule of 20 calendar weeks and students are expected to spend 20h per week on this course on average. Given six students on an average team this results in an effort of about 12-13 person months<sup>1</sup>. Projects span all phases of software development, from initial customer contact to the delivery of a product. Most projects have external customers. All project work has to be documented in detail by means of deliverables, presentations, prototypes, weekly reports, diaries and an on-line project workbook. Details about the course and the development process used can be found in [7].

The subjects of the case studies were 3rd to 5th year Computing Science students. The subjects were attending the OOSD course, held at Umeå University, Sweden, described above. All students had good knowledge in programming and had taken a basic software engineering course before entering OOSD. Some of the students also had some experience from software development outside the university. Therefore, the participants can be seen as semi-professional developers. In study one there were four teams of five members. The developed applications were “Inredaren” and “UmUportal”. The first one is a floor planning system that provides the user with a window-based user interface. It has two main functions, floor drawing and floor furnishing. The application was ordered by a teacher of the computing science department (internal customer). UmUportal is a personal portal for Umeå University. The application was ordered by an external customer, the IT chief of Umeå University. In study two, the students were organised into five teams of five-six members each. The developed applications were an “editor for XML metadata” and a “Course Pre-requisites Checking System”. The first application includes a textual and graphical (WYSIWYG) editor for XML-based forms, support for undo, and support for internationalised forms. The second application is an on-line, web-based system for course registration and simple curriculum management. In both cases, the customers were external to the computing science department. The first customer was a high-tech company active in the areas of digital media management, networking and high performance computing in Umeå. The customer of the second application was the faculty of teacher education in Umeå.

The students were required to use specific support analysis and design tools (Rational Rose in study one and Together ControlCenter in study two). The cost of a requirements change was calculated throughout all software development activities, i.e. from analysis to integration into the prototype, including the update of all affected documents.

---

<sup>1</sup> Approximately 20% of the total time available is spent on lectures and exercises, the learning of new methods, languages, and tools and course administration. A person month amounts to about 152 hours of work, according to COCOMO.

The studies were “specific” since they were focused on managing requirements. They both addressed a real problem, i.e. the difference between intuitive and historical based predictions in study one and the difference between intuitive predictions and predictions based on detailed impact analysis of cost of changes to requirements in study two. The objects under study were Software Requirements specifications (SRS), and the software process used.

## **5. Limitations of the Case Studies**

By performing these case studies, we identified three major risks that could become negative events:

- Classroom projects are usually stable and well defined. There was therefore the risk that the projects under study did not have changing requirements and historical data. To minimise this risk, we contacted the customers and agreed with them to change the requirements of the projects. This was especially true in study two.
- The subjects might not be motivated to participate in the case studies. The consequence of this would be a lack of data from the participants. To minimise this risk we decided to offer an inducement to the subjects in study one, while in study two we told the students that the data collection was a requirement to pass the course.
- By participating to the case studies, the subjects could become aware of the importance of managing requirements. The students might work harder to get good cost estimations. This problem is known as the Hawthorn (or observer) effect [15]. When the project personnel become aware that their work is being monitored, their behaviour will change. In both studies, this risk was lost since the course require a considerable amount of work therefore we did not expect the students to work more only to get better estimations.

Other minor risks were the possibilities that: 1) the subjects could use a formal model for cost predictions like COCOMO, affecting the results of the case study; 2) the participants could adopt requirements management tools or consult a requirements management expert; 3) the students could have misunderstood the terminology used in the forms. This risk was low because we kept e-mail contact with the students during the studies. The personnel experience was not investigated before the start of the studies and the participants decided the team’s constellation. Therefore the results of the studies could be affected by the team’s experience. We thought that these risks were minimal and were managed by interviewing the subjects and navigating in their course web site.

There were other general risks common to all empirical studies [3], for instance the danger to interpret the results without attempting to understand factors in the environment that might affect data. Underestimating the resources needed to validate and analyse the data, to associate measures with wrong scale type and consequently analyse data with the wrong statistical test [6] are other risks. During the validation process, another risk was to not know the amount of data that will never be reported. However, these risks were minimal. Unfortunately, the first two risks described above became partially true in study one resulting in too little data collected to be able to draw statistical conclusions.

## **6. Hypotheses and Plans**

In study one, we assumed that intuitive cost estimations are at least as good as estimations based on historical data (the null hypothesis  $H_0$ ). For the alternative hypothesis  $H_1$  we assumed that intuitive cost estimations are worse than estimations based on historical data. The null and alternative hypotheses in study two are similar to the first study, the difference is in the estimations based on an impact analysis checklist rather than historical data.

The independent variables were the personnel experience and the project. There was one factor, cost predictions and two treatments: intuitive cost predictions, and controlled cost predictions. The dependent variable was “precision of cost predictions”. All the students participating to the OOSD course were selected. Each team had at least one team manager and one requirements engineer. Each student in the team could assume different roles during the project. The design type chosen was one factor with two treatments.

The design principle followed was balancing i.e. to assign the treatments so that each treatment had equal number of subjects. In study one, two teams of students developed a student portal and two teams developed a Floor Planning System. We assigned the forms such that two different forms could evaluate each project. According to our plans, two teams should have performed intuitive cost predictions, and two teams should have performed cost prediction based on historical data. However, as written above, only two teams performed the data collection. In study two we had five groups therefore we assigned the forms such that at least two different forms could evaluate each project.

The kind of statistic chosen for our measures: # requirements, # changes to requirements, and time, was a parametric statistic because as suggested by [5] these measures reach at least the interval level. The instruments are usually of three types: objects, guidelines, and measurement instruments. The only instrument used during the study was forms contained in plain text e-mails. This choice was accomplished because of simplicity and flexibility in the answers.

### 6.1. Description of Measures

In both studies, the measures collected in relation to the hypothesis described above are: # requirements, # changes per requirement, and cost of changing requirements. Other measures have been collected to have a general overview of how the requirements are managed in students’ projects and to perform an internal validation of those measures [14].

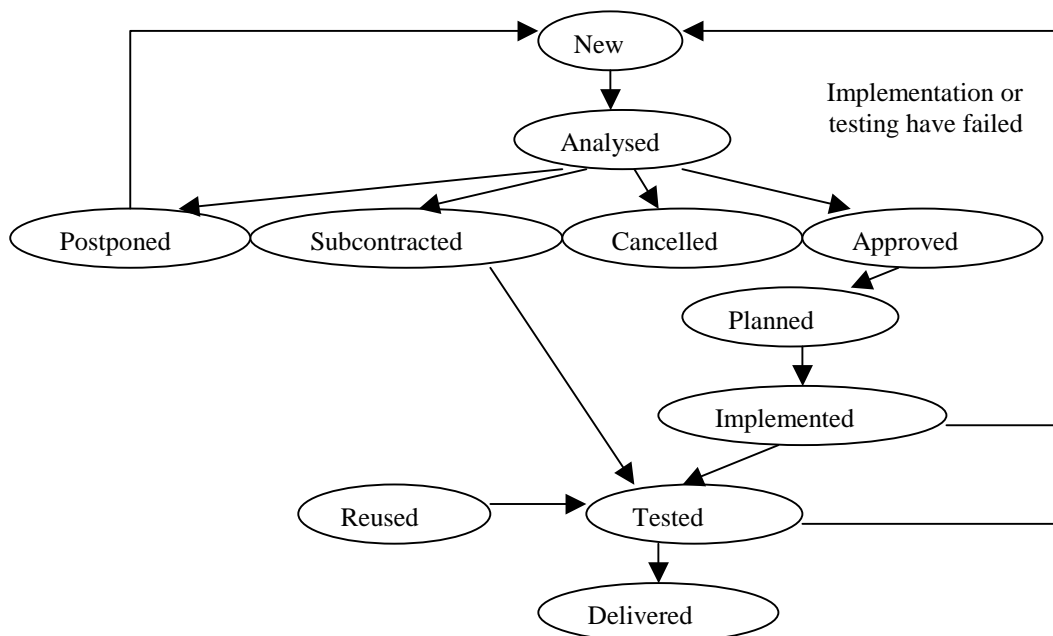


Figure 1: A simplified requirements life cycle

The # requirements is obtained by counting the functional and non-functional requirements. This counting is done disregarding the status of each requirement (a list of possible requirements states is shown in figure 1). The measure # requirements can be used in

conjunction with the # changes per requirement to assess the level of requirements volatility. The measurement rule followed in the case studies was to count all requirements that were in any of the states shown in figure 1. We excluded from the count only the deleted requirements.

The # changes made to requirements can be used to help determine requirements stability as well as to measure the impact of changes on the software process, on the budget, and on the schedule of the project. As a requirement is reviewed, all changes are recorded. The measure # changes made to requirements can be used in conjunction with other measures to chart general trends within the requirements management process. The measure # changes to requirements includes any change to a requirement that affects the development of the requirement. A requirement deletion is not considered to be a change to a requirement and readability improvements to the requirements specifications are not considered changes unless they affect the development of the requirements. The measurement rule used during the case studies was to count all changes to requirements demanding at least 15 minutes in order to perform the change. Please note that a change request can be made of many changes to requirements.

*Table 1: Requirements Management Measures used in the case studies*

<b>Entity</b>	<b>External Attribute</b>	<b>Measure</b>	<b>Domain</b>	<b>Range</b>	<b>Scale</b>
Software Requirement Specification	Stability, Change impact	# Requirements	SRS	Natural Numbers	Ratio
Requirements	Stability, volatility, change impact	# Changes per requirement	Requirements	Natural Numbers	Ratio
Change to requirement	Cost, effort of change to requirements	Time	Changes to requirements	Minutes	Interval

The cost of change is an indirect measure usually expressed as a function of variables like size of product, resources etc. The purpose of this measure is to provide information about the actual cost for changing requirements and compare this information to the estimated cost. This measure can be used to assess the overall impact of requirements change on the software project. We measured the cost in terms of resources used (the time spent on performing the change). This is because in software projects usually the staff cost dominates the overall project cost [9]. In study one, we did not define directly how to calculate the resources. Only after some discussions with the students we decided to consider the resources necessary for analysis of the change and for implementation of the change. If a change request affected many requirements, the cost of change to requirements was calculated in average for each requirement. In study two we considered the resources necessary to implement the change and not the resources for analysing the impact of change.

To test our hypothesis, each team attending the Object Oriented Software Development course collected data for the measures described above and other information useful to document the changes to requirements. Data was collected for the measures shown in table 1.

## 7. Results of Case Study One

The subjects were prepared during a lecture where we explained the background of the research, the case study goals, and we showed the forms to be filled in. During the execution of the study (from September 2002 to January 2003), the main concern was to ensure that the study was conducted according to the plans. We had e-mail contact with the students about how to collect the measures related to time and which “changes to requirements” were relevant to the study. There were some misunderstandings in how the data should be collected. For instance, one team counted the # changes to requirements while the other team counted the # change requests. One team considered a deleted requirement as a change to a requirement while the other team did not consider it in the same way.

*Table 2: Summary of data collected in study one*

Measures	First Iteration		Second iteration	
	Team A	Team B	Team A	Team B
# Requirements	29	12	31	12
# Changes	26	9	2	0
Estimated Time per change	24	17	15	--
Actual Time per change	26	40	45	--

A summary of the data collected is shown in table 2. The measurement unit for time is minutes. The estimated and actual time in the cells are sums of data collected every week divided by the # changes. In average, the time spent for each change was 40 minutes for team B and 26 minutes for team A. The reasons for changing use cases were in general for purposes of correction and improvement.

The requirements were changed mainly during the first iteration of the project. Afterwards, the requirements were very stable for both teams. Team B had some changes to use cases only during week 41. Team A changed use cases during the first two weeks of the first iteration and between the two iterations.

In table 2 we can observe the difference between the actual and expected time. Team B’s estimation was approximately less than half of the actual time necessary to perform the change. During the second iteration, team A’s estimation was one third of the actual time necessary to perform the change.

### 7.1. Lessons Learned in Study One

As pointed out earlier, the study did not succeed as expected for two main reasons. First: among the four teams performing a project in the OOSD course, only two teams participated in the case study. The other two teams decided to not collect data. Secondly: the requirements were not as volatile as expected. According to the original plan, the controlled team should use historical data to estimate the cost of change in the second iteration. We expected meetings to take place between the customer and the developers in order to validate the requirements, and as a consequence, a series of requests of change from the customer. Sadly, the controlled team had little contact with the customer to validate their requirements and this affected the results of the study.

However we have learned some lessons, which are listed below.

- An inducement is not enough to engage students to participate in a case study. More effort should be spent in committing the participants and obtaining their consensus.
- More effort should be spent in pushing students and customers to discuss their requirements. The subjects belonging to the controlled team and their customer had few

discussions about requirements. We expected validation of the requirements from the student side and requests of change from the customer side.

- When doing measurement it is very important to carefully define the measurement rules. The measure # changes per requirements was collected in different ways by the two teams. The reasons for this misunderstanding can be a non-strict definition of the mapping rules and a consequent confusion of the participants collecting the data. Strict definitions of measures and measurement rules are crucial when we perform empirical studies, otherwise it is difficult to evaluate the results of the measurement activities.

As we can see in table 2, the data available are only intuitive estimations and these are not very precise.

## 8. Partial Results of Case Study Two

As for study one, we prepared the subjects during a lecture in the beginning of the course. We decided to not show the details of the forms to be filled in during this lecture. Instead we provided the subjects with a document containing detailed instructions of how the data should be collected. The instructions contained the definitions of the measures, examples of how to collect the measures and the forms to be filled in. Two different instruction documents were designed, one for the teams whose estimations were intuitive and other for the teams making estimations based on an impact analysis checklist. Among the examples, we described how to count time. The measurement of time spent in performing the change was the sum of the single times spent by each developer in the team. This measure should include the time necessary for a change to requirement to be developed (postponed, approved, implemented etc.). At the time of writing, study two is still ongoing. The measures collected during study two are the same as for study one (shown in table 1). Data has been collected from week 40 to week 46 i.e. the first seven weeks of the project schedule.

*Table 3: Summary of data collected in study two*

Measures	Team 1	Team 2	Team 3	Team 4	Team 5
# Requirements	29	9	17	28	25
# Changes	6	4	1	8	3
Estimated time per change	165	--	4800	--	35
Actual Time per change	--	--	--	--	25

As we can observe in table 3, once again we had troubles in collecting data. The "--" in the table cells stand for "not submitted". The time is expressed in minutes. Only one team (team 5) submitted data on a regular base. The other teams missed the weekly deadlines several times, furthermore their data is incomplete. The reasons of this can be manifold. The course requires large amount of documentation and the weekly forms for the data collections could be a further burden that the subjects try to avoid. Another reason could be due to the understanding of the subjects that the data collection is not a requirement to succeed the course. However, the study is still ongoing therefore we are still in time to make adjustments to the forms and the study. Observing data for team 5 we can notice that the estimations are quite accurate. This can be due to the way team 5 is collecting data i.e. using an impact analysis checklist. However, team 5 had only three changes therefore data is not enough to draw conclusions.

## 9. Comparison of the Two Case Studies

In performing study two we accomplished some followed the lessons learned in study one. Among the improvements obtained, all the teams participated to the study while in study one only two teams participated. Although the data collection was not a requirement to succeed the course, in study two we made the students to believe that. Another lesson learned in study one was to define the measures more carefully therefore we distributed a document with the measures definition. In fact, we did not receive questions about how to collect time necessary to change requirements. Furthermore, we contacted the customers and we invited them to ask for changes. The changes to requirements in both studies were due to customer's change requests. However, we have probably done new mistakes. As we can see in table 3, the data collected was incomplete, imprecise and not delivered regularly. We suspect that the reason of this new problem was due to the large amount of data we were asking for.

A summary of the differences between the two studies can be seen in table 4.

*Table 4: Summary of the differences between the two studies*

	<b>New study</b>	<b>Old study</b>
<i>Intuitive data compared with:</i>	Impact analysis	Historical data
<i>Measures definition</i>	Yes	Partly
<i>Instructions provided</i>	Yes	Partly
<i># teams</i>	5	4
<i># students</i>	26	20
<i># teams who collected data</i>	5	2
<i>Both forms shown to all students</i>	No	Yes
<i>Well collected data</i>	No (not yet)	Yes
<i>Applications</i>	Editor for XML metadata, Course Pre-requisites Checking System	Inredaren, UmUportal

*Table 5: Comparison of the data collected in the two studies*

Measures	<b>New study</b>	<b>Old study</b>		
	Team 5	Team A	Team A iteration 2	Team B
# Changes	3	26	2	9
Estimated time per change	35	24	15	17
Actual time per change	25	26	45	40
Error	40%	10%	300%	140%

A full comparison of the results of the two studies cannot be performed because study two is ongoing. However a partial comparison can be observed in table 5. Team 5 has performed estimations following an impact analysis checklist while teams A and B have performed intuitive estimations. The estimations of team 5 are quite accurate compared to the estimations of team B in the first iteration and team A in the second iteration. This can be due to the use of the checklist. However, the estimations are not good compared to Team A in the first iteration. Furthermore the data collected is still too little to be able to draw conclusions.

## 10. Conclusion and Future Work

In the previous sections we have described two case studies performed in a university environment. The goal of the studies was to compare intuitive cost estimations of changes to requirements with estimations based on historical data (study one) and estimations based on

detailed impact analysis (study two). In performing study two we reused the lessons learned from study one as an example of what can go wrong and what risks should be taken into account.

At the time of writing, no conclusions can be drawn because study two is still ongoing and because the data collected is not complete. However as preliminary results, one team in case study two made quite accurate estimations of cost of changes to requirements performed by following an impact analysis checklist.

Currently, we are performing an empirical study in a medium-size company in Sweden. We are analysing historical projects at this company and based on this analysis we will design a suitable requirements management process for this company. After that we will compare the historical data with the actual data obtained following the new requirements management process. Finally, we will create a baseline which can be used for future projects as a reference for comparisons.

For the next year, we plan to perform another case study. We intend to compare estimations based on COCOMOII model and estimation based on impact analysis checklist. The study will be integrated into the course. Students will not know that there is a specific study. Course organisation material will be adapted to ensure that sufficient data is submitted. Another possible empirical study we would like to perform is to investigate if the size of software systems is proportional to the size of their requirements. In other words, we wonder if  $R_1$  and  $R_2$  are two requirements and  $size(R_1) < size(R_2)$ , the software systems  $S_1$  and  $S_2$  generated by these requirements are such that  $size(S_1) < size(S_2)$ .

## 11. Acknowledgement

Thanks to the students of the OOSD course Fall 2002 and Fall 2003 that have participated in the studies for their help by collecting data.

## 12. References

- [1] Anderson, S., and Felici, M., "Requirements Changes Risk/Cost Analyses: an Avionic Case Study", in Foresight and Precaution - Proceedings of ESREL 2000 - SARS and SRA-Europe Annual Conference, 2, Edinburgh, Scotland, UK, 15-17 May 2000.
- [2] Basili, V.R., and Rombach, H.D., "The TAME Project: Towards Improvement-Oriented Software Environments", IEEE Transactions on Software Engineering, 14(6), 1988, pp.758-773.
- [3] Basili, V.R., and Weiss, D.M., "A Methodology for Collecting Valid Software Engineering Data", IEEE Transactions on Software Engineering, 10(6), November 1984, pp. 728-738.
- [4] Baumert, J.H., and McWhinney, M.S., "Software Measures and the Capability Maturity Model", Software Engineering Institute Technical Report, CMU/SEI-92-TR-25, ESC-TR-92-0, 1992.
- [5] Briand, L.C., El Eman, K., and Morasca, S., "Theoretical and Empirical Validation of Software Product Measures", International Software Engineering Research Network Technical Report, #ISERN-95-03, 1995.
- [6] Briand, L.C., El Eman, K., and Morasca, S., "On the Application of Measurement Theory in Software Engineering", Journal of Empirical Software Engineering, 1(1), 1996, pp.61-88.
- [7] Börstler, J., "Experiences with Work-Product Oriented Software Development Projects", Journal of Computer Science Education, 11(2), June 2001, pp.111-133.
- [8] Costello, R.J., and Liu, D., "Metrics for Requirements Engineering", Journal of Systems and Software, 29, 1995, pp 39-63.
- [9] Fenton, N.E., and Pfleeger, S.L., "Software Metrics - A Rigorous & Practical Approach", 2nd Edition, International Thomson Publishing, Boston, MA, 1996.

- [10] Hammer, T.F., Huffman, L.L., and Rosenberg, L.H., "Doing requirements right the first time", CROSSTALK The Journal of Defence Software Engineering, December 1998, pp 20-25.
- [11] Lavazza, L., and Valletto, G., "Enhancing Requirements and Change Management through Process Modelling and Measurement", Proceeding of ICRE 2000 - Fourth International Conference on Requirements Engineering, Schaumburg, Illinois, 19-23 June 2000.
- [12] Lavazza, L., and Valletto, G., "Requirements-Based Estimations of Change Costs", Empirical Software Engineering - An International Journal, Kluwer, 5(3), November 2000.
- [13] Loconsole, A., "Measuring the Requirements Management Key Process Area", Proceedings of ESCOM - European Software Control and Metrics Conference, London, UK, April 2001.
- [14] Loconsole, A., and Börstler, J., Theoretical Validation and Case Study of Requirements Management Measures", Umeå University Internal Report, Uminf 03.02, July 2003.
- [15] Mayo, E., "The Human Problems of an Industrial Civilization", New York: MacMillan, 1933.
- [16] Paulk, M.C., Weber, C.V., Garcia, S., Chrissis, M.B., and Bush, M., "Key Practices of the Capability Maturity Model Version 1.1", Software Engineering Institute Technical Report, CMU/SEI-93-TR-25 ESC-TR-93-178, Pittsburgh, PA, 15213-3890, USA, February 1993.
- [17] Pfleger, S.L., "Software Engineering Theory and Practice", Prentice Hall, Upper Saddle River, New Jersey, 1998.
- [18] Raynus, J., "Software Process Improvement with CMM", Artech House Publishers, Boston, 1999.
- [19] Reifer, D.J., "Requirements Management: The Search for Nirvana", IEEE Software, May/June 2000, pp. 45-47.
- [20] Rosenberg, L.H., and Hyatt, L., "Developing an Effective Metrics Program", European Space Agency Software Assurance Symposium, the Netherlands, March 1996.
- [21] The Standish Group, "The CHAOS Report", Dennis, MA: The Standish Group, 1994.
- [22] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., and Wesslen, A., "Experimentation in Software Engineering An Introduction", Kluwer Academic Publishers, Boston/Dordrecht/London, 2000.