Extending the UCP Model by Incorporating the Prevailing Trends in Software Effort Estimation

M Usman Habib
Software Engineer
Central Depository Company of Pakistan

M Asim Ali
Assistant Professor
Dept of Computer Science
FAST – NUCES, Karachi

Nauman Atique
Assistant Professor
Dept of Computer Science
FAST – NUCES, Karachi

ABSTRACT

The research work is an extension of the original UCP Model for software estimation by incorporating the factors that are currently weighted in estimation of software by IT professionals. The rationale of the study is to make UCP Model effective, realistic and more accurate for the current IT industry. To identify the gap between the original UCP model and the current estimation considerations, a survey was conducted and the results of the survey were used to extend the model.

General Terms

Software Estimation, Use Case Point estimation.

Keywords

Software Estimation, Use Case Point, measurement, Pakistan.

1. INTRODUCTION

Use Cases have become a standard artifact to representation software requirements and based on Use Cases, Karner formulated UCP (Use Case Points) Model [2] for software estimation. Although there exists other estimation models e.g. FP and COCOMO [5] but they are based on lines of code rather the project requirements and therefore were applicable only to procedural paradigm. The idea of UCP was presented to estimate software developed using Object Oriented paradigm [2]. Since software products are currently being developed using object oriented paradigm and procedural languages are now rarely used, therefore, considering the practicality of implementation of projects in object oriented languages the authors chose UCP model for the study. However the original UCP presented by [3] has become obsolete as it does not consider the trends and practices of the current IT industry. Hence, to make UCP model a better fit for current IT industry the authors extend the original UCP model.

2. ISSUES AND PROBLEMS

The original UCP model was proposed in 1993 [1] and there have been efforts by other researchers to improve the UCP model from different perspectives using different approaches [6]. However there is a need to revamp the original UCP model for its affective use and applicability for projects that are undertaken by the IT industry with recent considerations not only with respect to technology and the environment but also in accordance with the project classification and work items (use cases and reports).

As UCP model is rightly based on a defined set of technical and environmental factors but some of the technical and environmental factors, considered by IT professionals of recent times, are not addressed by the original UCP model. Hence the authors addressed the need of extending the UCP by extending the set of technical and environmental factors with the prevailing trends. Moreover the difference in the estimation mechanism between reports and Use cases; and difference in calculating the estimation effort based on project classification is also addressed.

Hence, the four problem areas addressed are listed as follows:

a) Prevailing Technical Factors affecting the Estimation [2]

b) Prevailing Environmental Factors affecting the Estimation [2]

c) Differentiating estimation of Reports from Use Cases

d) Estimating the Project Effort based on Project Classification (along with UCP)

3. APPROACHES AND METHODS

A survey was conducted as part of the study to identify the technical and environmental factors and their affect on the estimation. The survey was also used to identify the complexity level of the reports based on their description and the need to differentiate estimation of reports from use cases.

Also a formula has also been inferred from the effort estimation formula of COCOMO [5] to accommodate the project classification attribute in context of UCP.

Finally a methodology for extended UCP model is proposed by consolidating the above two approaches.

4. CONDUCTING THE QUESTIONNAIRE SURVEY

Because recent trends in software estimation could only be identified from IT professionals working in the IT industry, therefore a questionnaire based survey was prepared and distributed to the IT professionals.

Seven steps process defined in the SEI Guideline of Survey Design [3] was followed. Below is the seven-step survey process that was followed with related information:

1) Identify the research objectives - Potential improvements in estimation in context of UCP model.

2) Identify and characterize the target audience - IT professionals with insight of software estimation.

3) Design the sampling plan – IT professionals of diversified experiences in the industry, across different organizations, in different countries were focused.

4) Design and write the questionnaire – All of the survey questions are close-ended, in order to make it easier to be completed within limited time.

5) Pilot test the questionnaire – The initial draft of survey was distributed to limited set of people to get feedback for improvement. The feedback was incorporated and finalized survey was devised.

6) Distribute the questionnaire - The survey was distributed to the audience in two ways:

    a) Emailing the Electronic Survey Form (Word Document)

    b) A web based version of the survey form was prepared and URL shared with the IT professionals who submitted their responses to the survey – URL: https://spreadsheets.google.com/spreadsheet/viewform?formkey=dExFMnFhRmZISmluXzJ3N3lFYThyUlE6MQ

7) Analyze the results and write report – In total, there were 38 respondents of 23 organizations, from 8 countries, with many organizations with software process improvement under CMM, CMMI [8] or ISO-9000 standards [7]. The average experience of the respondents was 8 years.

In total seven questions were asked in the survey to facilitate four areas of consideration, identified earlier.

4.1 Prevailing Technical Factors Affecting the Estimation

From the survey two questions were asked to identify the technical factors and their respective impact, which are considered current estimation.

First question was asked to identify the impact of the following factors for increase in estimation – i.e. factors that may contribute to increase the project development effort:

a) Technological Risks in the project

b) Multilingual Support in the project

c) Integration with other systems

d) Scalability requirements

e) Required Software Reliability (e.g. Space shuttle, satellite operations etc.)

f) Documentation (of code and other technical artifacts)

g) Development of Unit Test Cases for unit testing

Second question was asked to identify the impact of the following factors for decrease in estimation– i.e. factors that may contribute to decrease the project development effort:

a) Use of Third Party API / Pre-build components (e.g. Reporting Engine - Crystal Report, MS Reporting Services, JReport etc. Data Access Layer e.g. Hibernate etc.)

b) IDE (Integrated Development Environment) supporting automatic code generation and/or automatic code introspection

c) User of Automated testing tool for testing (reducing testing cycles time)

Each of the factors listed above were rated by survey respondents on the basis of set of ratings – Very High, High, Nominal, Low, Not Applicable.

4.2 Prevailing Environmental Factors Affecting the Estimation

From the survey two questions were asked to identify the environmental factors and their respective impact, which are considered current estimation.

First question was asked to identify the impact of the following factors for increase in estimation – i.e. factors that may contribute to increase the project development effort:

a) Lack of stakeholder (sponsor, subcontractor etc.) collaboration

b) Distractions for the Development Team (Facebook, Twitter, YouTube, Emails, Online News / Stocks, Blogging, Network Games, Online Games, Gossiping etc.)

c) Geographically Dislocated team

Second question was asked to identify the impact of the following factors for decrease in estimation– i.e. factors that may contribute to decrease the project development effort:

a) Project Manager’s Capability

b) CMM / CMMi compliance level of the organization

c) Testing team’s experience with testing tools & methodologies

d) Testing team’s experience with the application / project domain

Each of the factors listed above were rated by survey respondents on the basis of set of ratings – Very High, High, Nominal, Low, Not Applicable

4.3 Differentiating Estimation of Report from Use Cases

From the survey two questions were asked. First question was asked to identify the complexity of reports on a scale of 1 to 10 – with 10 being the most complex. Following are the most common types of reports for which were rated by the survey respondents.

a) Report based on query from one entity only

b) Report based on query from two or more entities (table join) only

c) Report based on query with table joins along with inner queries

d) Report based on query with table joins along with calls to one or more user defined database function / stored procedure

e) Report based on query with recurring function (due to recurring dependency of records i.e. each record has a Foreign Key reference to the table itself)

In second question respondents were asked if they agree with the notion that reports should be estimated differently from use cases - ‘Yes’ and ‘No’ were the only two options available.

4.4 Estimating the Project Effort Based on Project Classification by COCOMO

Respondents were asked if they agree that the project classification used by COCOMO [5] (based on lines of code – procedural languages) for effort estimation may also be used for estimating projects that are developed using object oriented languages - ‘Yes’ and ‘No’ were the only two options available.

5. ANALYSIS AND DERIVED DATA

5.1 Prevailing Technical and Environmental Factors Affecting the Estimation

For the first four questions that required rating of technical and environment factors for increased and/or decreased estimation from the set of ratings (‘Very High’, ‘High’, ‘Nominal’, ‘Low’, ‘Not Applicable’); the set of ratings were converted to numerals for data analysis and deduction of results.

Table 1: Conversion of response ratings to numerals

Answer Option	Value
Very High	4
High	3
Nominal	2
Low	1
Not Applicable	0

Converting the respondents’ ratings to corresponding numerals, the average of values of the respondents’ responses, for Prevailing Technical Factors that should be considered for increased estimates (Ti) were calculated to be as follows:

Table 2: Technical Factors for increased estimates - Ti

	Technical Factors for increased estimates - Ti	Value
a.	Technological Risks in the project	2.74
b.	Multilingual Support in the project	2.05
c.	Integration with other systems	2.95
d.	Scalability requirements	2.71
e.	Required Software Reliability (e.g. Space shuttle, satellite operations etc.)	2.32
f.	Documentation (of code and other technical artifacts)	2.08
g.	Development of Unit Test Cases for unit testing	1.92

Converting the respondents’ ratings to corresponding numerals, the average of values of the respondents’ responses, for Prevailing Technical Factors that should be considered for decreased estimates (Td) were calculated to be as follows:

Table 3: Technical Factors for decreased estimates - Td

	Technical Factors for decreased estimates – Td	Value
a.	Use of Third Party API / Pre-build components (e.g. Reporting Engine - Crystal Report, MS Reporting Services, JReport etc. Data Access Layer e.g. Hibernate etc.)	2.79
b.	IDE (Integrated Development Environment) supporting automatic code generation and/or automatic code introspection	2.29
c.	User of Automated testing tool for testing (reducing testing cycles time)	2.11

Converting the respondents’ ratings to corresponding numerals, the average of values of the respondents’ responses, for Prevailing Environmental Factors that should be considered for increased estimates (Ei) were calculated to be as follows:

Table 4: Environmental Factors for increased estimates - Ei

	Environmental Factors for increased estimates – Ei	Value
a.	Lack of stakeholder (sponsor, subcontractor etc.) collaboration	2.95
b.	Distractions for the Development Team (Facebook, Twitter, YouTube, Emails, Online News / Stocks, Blogging, Network Games, Online Games, Gossiping etc.)	2.03
c.	Geographically Dislocated team	2.42

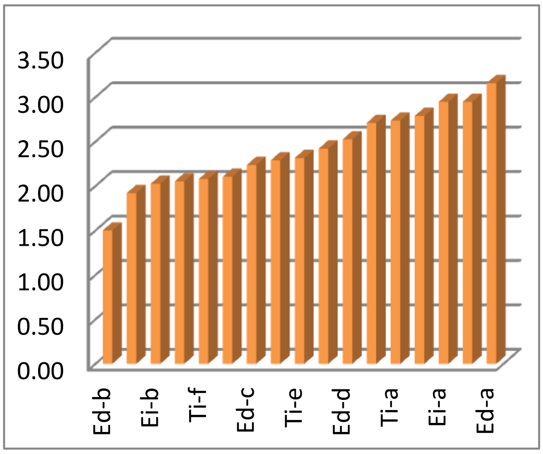
Converting the respondents’ ratings to corresponding numerals, the average of values of the respondents’ responses, for Prevailing Environmental Factors that should be considered for decreased estimates (Ed) were calculated to be as follows:

Table 5: Environmental Factors for increased estimates - Ed

	Environmental Factors for decreased estimates – Ed	Value
a.	Project Manager’s Capability	3.16
b.	CMM / CMMi compliance level of the organization	1.5
c.	Testing team’s experience with testing tools & methodologies	2.24
d.	Testing team’s experience with the application / project domain	2.53

As all the values of Ti, Td, Ei and Ed were evaluated against the same scale therefore all the values were plotted in increasing order of values of technical and environmental factors over different types of graph for analysis.

Figure 1: Bar Chart - Technical and Environmental Factors



All the three graph view shows that there is a sharp drop in the graph for values under 2, therefore all the values that fall under 2 are pruned from the analysis – this is to ensure that insignificant (low value and sharp decrease in curve) factors aren't introduced to the UCP model and cause errors. As a matter of fact, even an insignificant factor would have a very high probability (37/38 = 0.97) of having a value of greater than zero – thus pruning the values greater than zero but less than 2 makes sense.

Table 6: Technical and Environmental Factors and their Values in increasing order

	Factors	Values
~~1.~~	~~Ed-b~~	~~1.50~~
~~2.~~	~~Ti-g~~	~~1.92~~
3.	Ei-b	2.03
4.	Ti-b	2.05
5.	Ti-f	2.08
6.	Td-c	2.11
7.	Ed-c	2.24
8.	Td-b	2.29
9.	Ti-e	2.32
10.	Ei-c	2.42
11.	Ed-d	2.53
12.	Ti-d	2.71
13.	Ti-a	2.74
14.	Td-a	2.79
15.	Ti-c	2.95
16.	Ei-a	2.95
17.	Ed-a	3.16

Table 7

Now, we have the minimum value 2.03 (say V-min) and maximum value 3.16 (say V-max). Again it should be noted that the value of V-max would almost always be less than 4 (the max possible value).

Another, interesting fact to note is that the set of values for all the Technical and Environmental factors could be graded into 4 grades for each factors based on possible weights along with the impact of the factors in the survey.

With four bands of impact and the weight factors for technical and environmental factors, the minimum (V-min) and maximum (V-max) values for factors are used to define 4 bands that could correspond to the 4 impact grades and corresponding weights of factors.

Table 8: Factor Impact Band and Factor Weights

Impact of Factor	Weights for Ti	Weights for Td	Weights for Ei	Weights for Ed	Value Band
Very High	2	-1	-1	2	>= 2.88 and <= 3.16
High	1.5	-0.75	-0.75	1.5	>= 2.60 and < 2.88.
Nominal	1	-0.5	-0.5	1	>= 2.31 and < 2.60
Low	0.5	-0.25	-0.25	0.5	>= 2.03 and < 2.31

Table 9: Prevailing Factors with Impact Weights for UCP

	Factor Code	Factor	Values	Weight
1.	**Ti-a**	Technological Risks in the project	**2.74**	**1.5**
2.	**Ti-b**	Multilingual Support in the project	**2.05**	**0.5**
3.	**Ti-c**	Integration with other systems	**2.95**	**2**
4.	**Ti-d**	Scalability requirements	**2.71**	**1.5**
5.	**Ti-e**	Required Software Reliability	**2.32**	**1**
6.	**Ti-f**	Documentation (of code and other technical artifacts)	**2.08**	**0.5**
~~7.~~	~~Ti-g~~	~~Development of Unit Test Cases for unit testing~~	~~1.92~~	(N/A)
8.	**Td-a**	Use of Third Party API / Pre-build components	**2.79**	**-0.75**
9.	**Td-b**	IDE (Integrated Development Environment) supporting automatic code generation and/or automatic code introspection	**2.29**	**-0.25**
10.	**Td-c**	User of Automated testing tool for testing	**2.11**	**-0.25**
11.	**Ei-a**	Lack of stakeholder (sponsor, subcontractor etc.) collaboration	**2.95**	**-1**
12.	**Ei-b**	Distractions for the Development Team	**2.03**	**-0.25**
13.	**Ei-c**	Geographically Dislocated team	**2.42**	**-0.5**
14.	**Ed-a**	Project Manager’s Capability	**3.16**	**2**
~~15.~~	~~Ed-b~~	~~CMM / CMMi compliance level of the organization~~	~~1.5~~	(N/A)
16.	**Ed-c**	Testing team’s experience with testing tools & methodologies	**2.24**	**0.5**
17.	**Ed-d**	Testing team’s experience with the application / project domain	**2.53**	**1**

Table 10: New Set of Technical Factors

F_i	Factors Contributing to Complexity	W_i
F1	Distributed systems.	**2**
F2	Application performance objectives, in either response or throughput.	**1**
F3	End user efficiency (on-line).	**1**
F4	Complex internal processing.	**1**
F5	Reusability, the code must be able to reuse in other applications.	**1**
F6	Installation ease.	**0.5**
F7	Operational ease, usability.	**0.5**
F8	Portability.	**2**
F9	Changeability.	**1**
F10	Concurrency.	**1**
F11	Special security features.	**1**
F12	Provide direct access for third parties	**1**
F13	Special user training facilities	**1**
F14	Technological Risks in the project	**1.5**
F15	Multilingual Support in the project	**0.5**
F16	Integration with other systems	**2**
F17	Scalability requirements	**1.5**
F18	Required Software Reliability	**1**
F19	Documentation (of code and other technical artifacts)	**0.5**
F20	Use of Third Party API / Pre-build components	**-0.75**
F21	IDE (Integrated Development Environment) supporting automatic code generation and/or automatic code introspection	**-0.25**
F22	User of Automated testing tool for testing	**-0.25**

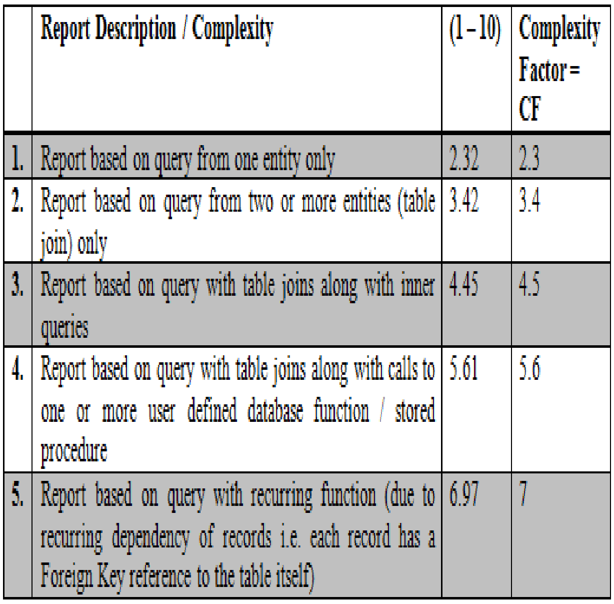
Table 11: New Set of Environmental Factors

E_i	Environmental Factors Contributing to Complexity	W_i
F1	Familiar with Application Domain	**1.5**
F2	Part time workers	**-1**
F3	Analyst capability	**0.5**
F4	Application experience	**0.5**
F5	Object oriented experience	**1**
F6	Motivation	**1**
F7	Difficult programming language	**-1**
F8	Stable requirements	**2**
F9	Lack of stakeholder (sponsor, subcontractor etc.) collaboration	**-1**
F10	Distractions for the Development Team	**-0.25**
F11	Geographically Dislocated team	**-0.5**
F12	Project Manager's Capability	**2**
F13	Testing team's experience with testing tools & methodologies	**0.5**
F14	Testing team's experience with the application / project domain	**1**

5.2 Differentiating Estimation of Reports from Use Cases

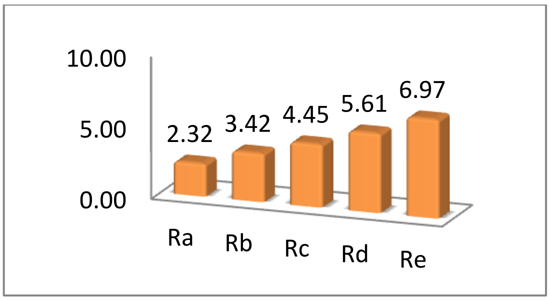
From the survey 92% of the respondents agreed that reports should be estimated differently from use cases. It is worth noting that there have been studies [4] that have proposed estimation of reports separately from the use cases however the survey carried as part of this study also identified the complexity scale of the reports that could be used to estimate the required effort to develop reports.

Table 12: Reports Complexity Level



	Report Description / Complexity	(1 - 10)	Complexity Factor = CF
1.	Report based on query from one entity only	2.32	2.3
2.	Report based on query from two or more entities (table join) only	3.42	3.4
3.	Report based on query with table joins along with inner queries	4.45	4.5
4.	Report based on query with table joins along with calls to one or more user defined database function / stored procedure	5.61	5.6
5.	Report based on query with recurring function (due to recurring dependency of records i.e. each record has a Foreign Key reference to the table itself)	6.97	7

Figure 1: 2D Bar Graph – Report Complexity



Hence, if the simplest report ('Report based on query from one entity only') requires 'h' estimated effort with 'CF' being the Complexity Factor for a report. From the illustrations above, it signifies that the 'Estimated Effort for Report' (EEF) would be a linear function:

Finally, Estimated Effort for Report (EEF) in man hours = $(h\ /\ 2.3)\ x\ CF$

5.3 Estimating the Project Effort based on Project Classification

COCOMO has been one of the most advanced, widely used and sophisticated estimation model that is used extensively for projects developed using procedural languages. As COCOMO depends on program size, measured in lines of code therefore for projects developed using Object Oriented languages could not be estimated using COCOMO. However the classification of projects defined by COCOMO is irrespective of the programming language paradigm [9], hence the classification of projects by COCOMO makes sense for projects developed using Object Oriented paradigm. And since, UCP model fits for the projects developed using Object Oriented languages [2] therefore the commonality of project classification concept can be used for UCP for improved estimation.

The above notion is supported from the survey results - 76% of the respondents agreed that the project classification by COCOMO also matters for projects developed using Object Oriented languages [10]; and because UCP model is for object oriented languages therefore the project classification of COCOMO would be considered for UCP model as well.

COCOMO classifies projects into three classes to calculate the Effort with the following formula:

Effort in man months = $a(KLOC)^b$

Where the values of variables 'a' and 'b' depends on the project classification as follows:

Table 13: Software Projects Classification

Software Projects	a	b
Organic	2.4	1.05
Semi-detached	3.0	1.12
Embedded	3.6	1.2

- Organic projects - "small" teams with "good" experience working with "less than rigid" requirements
- Semi-detached projects - "medium" teams with mixed experience working with a mix of rigid and less than rigid requirements

- Embedded projects - developed within a set of "tight" constraints (hardware, software, operational etc.)

Since, for object oriented languages the concept of KLOC doesn't exist and we could assume KLOC to be constant 'C' for object oriented languages. As, UCP model does not currently incorporate Software Project Classification therefore, the effort estimated by UCP model would be equivalent to the effort estimated for Organic Projects using COCOMO formula, hence:

Effort in man months = $a\ (C)^b$

Hours required per $UCP\ =\ HUCP$ (generally it is 20 hours)

Moreover, for UCP Effort in man hours = $UCP * HUCP$

Therefore:

$$\frac{UCP\ * HUCP}{30 * 8} = a\ (C)^b$$

$UCP * HUCP$ is estimated time for organic projects i.e. organic projects as baseline.

In $C =\ \ In\ UCP - In\ (\frac{576}{HUCP})$) / 1.05

Hence the equation for Base Line Constant for Estimation of projects 'in man hours' is derived to be:

$$\boldsymbol{C = e^{\frac{\left(In\ UCP - In\left(\frac{576}{HUCP}\right)\right)}{1.05}}}$$

For $HUCP = 20$

$$\mathrm{C} = \boldsymbol{e}^{(\frac{\boldsymbol{In\ UCP - 3.36}}{\boldsymbol{1.05}})}$$

The above results can be summarized as follows:

Table 14: Estimated Effort Method based on Project Classification

Software Projects	**Estimated Effort**	**Description**
Organic	$UCP * HUCP$	UCP is the calculated Use Case Points and HUCP is the number of hours required to implement one UCP.
Semi-detached	$a(C)^b$	$a\ =\ 3.0, b\ =\ 1.12$ and C is the baseline constant
Embedded	$a(C)^b$	$a\ =\ 3.6, b\ =\ 1.2$ and C is the baseline constant

6. THE EXTENDED UCP MODEL

From the results of section '4 - **Error! Reference source not found.**' the extended UCP Model would calculate the estimated effort in four steps:

Step-1: Calculate the UCP using new / extended set of Technical and Environmental Factors.

Step-2: Calculate the Estimated Effort for Reports in man hours.

Step-3: Calculate the Estimated Effort of Project based on Project Classification in man hours.

Step-4: Calculate the Total Estimated Effort in man hours by adding the Estimated Efforts from step-2 and step-3.

7. CONCLUSION

Estimating the effort of a software project has always been a difficult task. Several methods of software estimation proposed in the literature before object oriented technology became popular [2]. Since, object oriented approach has become *de facto* standard for software development; therefore it became necessary to revamp the estimation models to suite the object oriented technology [2]. Karner [1] proposed a method to estimate the effort using use case model which was attractive and suitable [2] but with the ever changing IT industry it became a bit obsolete due to emerging trends in IT industry. The authors of this paper extended Karner's approach by incorporating the current trends in software estimation.

8. REFERENCES

[1] Karner, Gustav. Metics for Objectory. Diploma Thesis, University of Linkoping, Sweden. No. LiTHIDA-Ex-93441:21. December 1993

[2] Kasi Periyasmay and Aditi Ghode. Cost Estimation using extended Use Case Point (e-UCP) Model .IEEE

[3] Kasunic, M. 2005. Designing an Effective Survey. Technical Handbook. Pittsburg: Software Engineering Institute. Carnegie Mellon University. September, 2005

[4] Carroll, Edward R., Estimating Software Based on Use Case Points. Agilis Solutions, A Business Unit of Hepieric, Inc. ACM. October 2005

[5] Boehm, B., Software Engineering

[6] Krintin & Monika "software effort estimation as collective accomplish: an analysis of estimation practice in a multi specialist team" http://iris.cs.aau.dk/tl_files/volumes/Volume22/no2/5Borte.pdf

[7] David Hoyle (2012): ISO 9000 Quality Systems Handbook; Fifth edition

[8] Mary Beth, Mike Konrad, Sandra Shrum., CMMI for development: Guideline for process integration and product Improvement, 2011

[9] Allen B. Tucker: Programming languages Principles and Paradigms, 2001

[10] M. Nadeem, M R Asim and M R J Quershi " A step forward to component based software cost estimation in object oriented environment" http:// arxiv.org/ftp/arxiv/papers/1202/1202.2511.pdf

[11] Mange and Stein "software development effort estimation: Demystifying and improving expert estimation" http://simula.no/research/se/publications/Simula.SE.427/simula_pdf_file