International Journal of Engineering & Technology, 7 (3) (2018) 1812-1817

SPC

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i3.13010
Research paper

Influence of end user development on software project estimation

Archana Srivastava [1] *, Dr. S. K. Singh [2], Dr. Syed Qamar Abbas [3]

[1] Research Scholar, Amity School of Engineering & technology, Amity University Uttar Pradesh, Lucknow, India-226010
[2] Professor, Amity University Uttar Pradesh, Lucknow, India-226010
[3] Director, Ambalika Institute of Management &Technology, Lucknow, India-226010
**Corresponding author E-mail: asahai@amity.edu, srivastavaarchana891@gmail.com*

Abstract

Use Case Point Method (UCP) is used to estimate software development effort. UCP uses a project's use cases to produce a reasonable estimate of a project's complexity and required man hours. Advance Use Case Point Method (AUCP) is an extension of UCP. AUCP extends UCP by adding the additional effort required in incorporating end user development (EUD) features in the software for overall project effort estimation. Today user needs are diverse, complex, and frequently changing hence need of EUD is also increasing. EUD features if incorporated in the software increases end user satisfaction exponentially but incorporating EUD features increases design time complexity and increases the effort significantly based on the end users requirements. This paper provides a case study to demonstrate the comparative analysis of UCP and AUCP using paired t-test. It also observes that there can be on an average 20% increase in overall effort of development on adding EUD features.

Keywords: *Use Case Point Method (UCP); Advance Use Case Point Method (AUCP); End User Development (EUD); Human Computer Interaction (HCI); End User Computing (EUC); Technical Complexity Factors (TCF); Environmental Complexity Factors (ECF)*

1. Introduction

Estimation of object oriented software cost and effort is an important and hard management activity. This is due to the lack of information to making decisions in the early phases of the development, frequently characterized by uncertainty. To help the managers in this task, there are in the literature many estimation models that usually include two main metrics: Lines of Code (LOC) and Function Points (FP) [12], both of them with skills and limitations. LOC is dependent on the programming language and the FP Analysis (FPA) is subjective and based on human decisions [13]. The most popular technique for object-oriented software cost estimation is Use Case Points (UCP) method.

The Use Case Points Method (UCP) is an effort estimation algorithm proposed by Gustav Karner (1993). Use Cases are frequently used to describe the business process of object oriented projects. Use cases are an effective method of modeling a software system. It builds a mutual vision of the problem at hand by bridging the gap between the people who understand the problem and the people who understand how to build a solution. This method was used to produce the estimate from the project's use cases. The UCP method analyzes the project's use case, actors, scenarios and various technical and environmental factors and abstracts them into an equation [10-13].

Advantages of UCP

1) Each use case describes one way the system is used, but major benefit of use case modeling is that it also describes all of the things that might go wrong. Identifying exceptions to a successful scenario early in the project saves a lot of time by finding subtle requirements.
2) The advantage to estimating with use case points is that the process can be automated. Some use case management tools will automatically count the number of use case points in a system. This can save the team a great deal of estimating time.
3) It is also possible to establish an organizational average implementation time per use case point. This would be very useful in forecasting future schedules. Though, this depends heavily on the assumption that all use cases are consistently written with the same level of detail.
4) Another advantage to use case points is that they are a very pure measure of size. Good estimation approaches allows separating estimation of size from deriving duration. Use case points qualify in this regard because the size of an application will be independent of the size, skill, and experience of the team that implements it [10].
5) An early project estimate helps managers, developers, and testers plan for the resources a project requires. As the case studies indicate, the UCP method can produce an early estimate within 20 percent of the actual effort, and often, closer to the actual effort than experts and other estimation methodologies [11].

2. End user development

End user development is an interdisciplinary field that traditionally relates to areas such as psychology of programming, empirical studies in software engineering, human computer interaction. Technological trends like ubiquitous computing, tangible and embodied interaction, and the internet of things, have renewed the interest in end-user development for diverse audiences looking

into industrial design, online communities, open innovation and crowd sourcing. EUD is inherently different from traditional software development. So supporting EUD by simple traditional approaches is often not sufficient to produce successful results.
End users usually do not have training in programming languages, formal development processes, or modeling and diagramming notations. Moreover, end users often lack the time or inspiration to learn these traditional techniques, since end users usually write code in order to achieve a short- or medium-term goal rather than to create a durable software asset that will produce a long term benefit. Consequently, supporting EUD requires providing appropriate tools, social structures, and development processes that are highly usable, quickly learned, and easily integrated into domain practice. End User Development is related to HCI fields of intelligent user interfaces, programming-by-demonstration, adaptive user interfaces and development tools [16].
End User Development features if incorporated in a software or website it enhances the quality and increases end user satisfaction. EUD features enhance End User Satisfaction, User satisfaction refers to the quality product, system or tool that is able to satisfy specific requirements of the end user. User satisfaction with an application has been defined as 'the affective attitude towards a particular computer application by an end user who interacts with the application directly' [7]. End users concerned are not professional developers but have sufficient knowledge of their respective domains and like to do coding or use various wizards to customize things as per their own requirements. EUD research mainly focuses on approaches for lowering the barrier of entry to software development. Such approaches cover a wide spectrum, from enhancing the macros and spreadsheets that millions use every day to sophisticated algorithms that create programs by example without ever exposing the user to textual code [11].
Demand of EUD is everywhere including the social networking sites. All active social network users produce and shares texts, images and videos. While developers can access such data through application programming interfaces (APIs) for creating games, visualizations and routines, end users have less control on such information. Their access is mediated by the social application features, which limits them in combining sources, filtering results and performing actions on groups of elements. FaceMashup, an end user development (EUD) environment supporting the manipulation of the Facebook graph is introduced to fill this gap [3].
Despite three decades of research on software cost estimation, the research community has yet to provide a viable model for End-User Development (EUD) environments. Hence one element of the size and effort is the additional design time expended in end-user programming [5]. None of the estimation model has included EUD development features as an additional cost driver. EUD essentially out-sources development effort to the end user. Additional EUD features increases the development effort but ensures high quality that is measured by the fulfillment of end user requirements. End User Development enhances the End user satisfaction level as users are involved throughout the development process starting from the requirement gathering to designing phase.

3. Case study UCP

Use Case Point Method (UCP) is as follows [4, 14]:
Let us take 5 projects and calculate UCP using the following steps.

3.1. Classify actors

For Simple WF (Weight Factor) = 1, For Average WF = 2, for Complex WF = 3.

Unadjusted Actor Weights (UAW) = ∑ (Actors in each group * WF)

Table 1: (UAW)

Actor Complexity level	Description	Weight	Project1	Value	Project2	Value	Project3	Value	Project 4	value	Project 5	value
Simple	Interacts through API, as Command Prompt	1	2	2	3	3	8	8	5	5	8	8
Average	Interacts through Protocol as TCP/IP, HTP	2	4	8	2	4	7	14	5	10	5	10
Complex	Interacts through GUI or Web Page	3	3	9	8	24	6	18	5	15	10	30
		UAW	Project 1	19	Project 2	31	Project 3	40	Project 4	30	Project 5	48

3.2. Calculating unadjusted use case weight (UUCW)

Use cases are classified as simple, average or complex depending on the number of transactions.

UUCW = ∑ (Use Case in each group * WF)

Table 2: (UUCW)

Use Case Complexity level	Description	Weight	Project1	Value	Project2	Value	Project3	Value	Project 4	value	Project 5	value
Simple	Using 1 to 3 Transactions	5	8	40	5	25	3	15	10	50	5	25
Average	Using 4 to 7 Transactions	10	5	50	8	80	4	40	8	80	4	40
Complex	Using more than 7 Transactions	15	9	135	4	60	7	105	3	45	10	150
		UUCW	project 1	225	project 2	165	project 3	160	project 4	175	project 5	215

3.3. Calculate the unadjusted use case point (UUCP)

Unadjusted Use Case Point is the sum of Unadjusted Use Case Weight (UUCW) and Unadjusted Actor Weights (UAW).

UUCP=UUCW+UAW

Table 3: (UUCP)

Project	UAW	UUCW	UUCP
Project 1	19	225	244
Project 2	31	165	196
Project 3	40	160	200
Project 4	30	175	205
Project 5	48	215	263

3.4. Calculating the technical complexity factor (TCF)

Technical complexity factor is calculated by using the formula

TCF = 0.6 + (0.01*TF)

Where

TF =T1 +T2 + ….T12+T13

Table 4: (TCF)

			project complexity									
Ti	**Technical factors**	**Weight**	**Project1**	Value	**Project2**	Value	**Project3**	Value	**Project 4**	value	**Project 5**	value
T1	Distributed System Required	**2**	1	2	2	4	1	2	3	6	5	10
T2	Response Time Is Important	**1**	3	3	4	4	1	1	4	4	5	5
T3	End User Efficiency	**1**	3	3	2	2	2	2	5	5	5	5
T4	Complex Internal Processing Required	**1**	3	3	5	5	1	1	3	3	3	3
T5	Reusable Code Must Be A Focus	**1**	0	0	3	3	2	2	3	3	3	3
T6	Installation Easy	**0.5**	0	0	3	1.5	1	0.5	2	1	5	2.5
T7	Usability	**0.5**	5	2.5	4	2	1	0.5	4	2	5	2.5
T8	Cross-Platform Support	**2**	0	0	2	4	0	0	3	6	5	10
T9	Easy To Change	**1**	3	3	3	3	1	1	3	3	4	4
T10	Highly Concurrent	**1**	0	0	3	3	2	2	3	3	4	4
T11	Custom Security	**1**	0	0	3	3	2	2	2	2	5	5
T12	Dependence On Third-Part Code	**1**	3	3	4	4	1	1	3	3	3	3
T13	User Training	**1**	0	0	3	3	2	2	3	3	4	4
		TF	**project1**	19.5	**project2**	41.5	**project3**	17	project 4	44	project 5	61
TCF = 0.6 + (0.01 * TF)		TCF	**project1**	0.795	**project2**	1.015	**project3**	0.77	project 4	1.04	project 5	1.21

3.5. Calculating the environmental complexity factor (ECF)

Environmental Complexity Factor (ECF) is calculated by using the formula

ECF = 1.4 + (-0:03 * EF)

Where:

EF = F1+F2+…+F7+F8

Table 5: (Environmental Factors)

			project complexity									
Fi	Environental factors	**Weight**	**Project1**	Value	**Project2**	Value	**Project3**	Value	**Project 4**	value	**Project 5**	value
F1	Familiar with Objectory	1.5	5	7.5	4	6	1	1.5	3	4.5	4	6
F2	Part time workers	-1	0	0	1	-1	1	-1	3	-3	3	-3
F3	Analyst capability	0.5	5	2.5	4	2	2	1	4	2	5	2.5
F4	Application experience	0.5	0	0	2	1	4	2	3	1.5	5	2.5
F5	Object oriented experience	1	5	5	3	3	4	4	3	3	5	5
F6	Motivation	1	5	5	3	3	4	4	4	4	5	5
F7	Difficult programming language	-1	0	0	2	-2	4	-4	3	-3	3	-3
F8	Stable requirements	2	3	6	4	8	3	6	3	6	5	10
		EF	**project1**	26	**project2**	20	**project3**	13.5	**project 4**	15	**project 5**	25
ECF = 1.4 + (-0.03 * EF)		**ECF**	**project1**	0.62	**project2**	0.8	**project3**	0.995	**project 4**	0.95	**project 5**	0.65

3.6. Calculating the use case point (UCP), where: UCP = UUCP * TCF * ECF

Table 6: (UCP)

Project	UUCP	TCF	ECF	UCP
Project 1	244.00	0.80	0.62	120.27
Project 2	196.00	1.02	0.80	159.15
Project 3	200.00	0.77	1.00	153.23
Project 4	205.00	1.04	0.95	202.54
Project 5	263.00	1.21	0.65	206.85

4. Advance use case point (AUCP)

It is an extension of UCP in which additional cost driver End user development factors are added. The method is as follows [6]:

a) Total eighteen EUD_Technical factors (EUD_TF) are identified and weights are assigned to each factor considering its impact on development. Each factor is assigned value 0 or 1, depending on whether that feature is required or not required in the software. If the feature is required it is rated as 1 else 0, and is multiplied by the assigned weight of EUD_TF. Take the summation of all factors.

b) Total eight EUD_Environmental factors (EUD_EF) are identified and weights are assigned to each factor considering its impact on development. Each factor is assigned value 0 or 1, depending on whether that feature is required or not required in the software. If the feature is required it is rated as 1 else 0, and is multiplied by the assigned weight of EUD_TF. Take the summation of all factors.

c) Calculate EUD Technical Complexity Factor, EUD_TCF = 0.6 + (0.01 * EUD_TF)

d) Calculate EUD Environmental Complexity Factor EUD_ECF = 1.4 + (0.03 * EUD_EF)

e) AUCP = UCP X (EUD_TCF X EUD_ECF)

Advance UCP is now calculated by taking the product of Use Case Point, End User Development Technical Complexity Factors and End User Development Environmental Complexity Factors.

5. Advance use case point (AUCP) case study

Suppose we have to add End user development features in the above five projects. The EUD requirements of all the projects are different.

5.1. Calculate EUD_technical factors (EUD_TF) for five projects

If the EUD_TF is required, it is rated as [1] else zero, and is multiplied by the assigned weight of EUD_TF. Take the summation of all factors. Following result is obtained.

Table 7: (EUD_TF)

EUD_Ti	EUD_TECHNICAL FACTORS	Weight	Project1	Value	Project2	Value	Project3	Value	Project 4	value	Project 5	value
T1	Inbuilt system assistance	1.2	1	1.2	1	1.2	0	0	0	0	0	0
T2	Creating reusable codes	1.4	1	1.4	1	1.4	0	0	1	1.4	1	1.4
T3	Sharing reusable code	1	0	0	1	1	1	1	1	1	1	1
T4	Easy & understandable codes	1.3	1	1.3	1	1.3	0	0	1	1.3	1	1.3
T5	Security features in codes for more control by end users	1.12	1	1.12	1	1.12	1	1.12	1	1.12	1	1.12
T6	Authentication features	1.3	1	1.3	1	1.3	0	0	1	1.3	0	0
T7	Inbuilt feedback about the correctness	1.2	0	0	1	1.2	1	1.2	1	1.2	1	1.2
T8	Testable codes	1.4	1	1.4	1	1.4	1	1.4	1	1.4	0	0
T9	Tools for analyzing by debugging	1.2	0	0	1	1.2	0	0	0	0	1	1.2
T10	Error detection tools	1.3	0	0	1	1.3	1	1.3	0	0	1	1.3
T11	online help availability	1.11	1	1.11	1	1.11	0	0	1	1.11	1	1.11
T12	Self - efficiency	1.20	1	1.2	1	1.2	1	1.2	1	1.2	1	1.2
T13	Perceived ease of use	1	1	1	1	1	0	0	1	1	1	1
T14	Perceived usefulness	1.2	1	1.2	1	1.2	1	1.2	1	1.2	1	1.2
T15	Flexible codes	1.25	0	0	1	1.25	0	0	1	1.25	0	0
T16	Scalability features	1.2	1	1.2	1	1.2	1	1.2	1	1.2	0	0
T17	End user training	1.5	0	0	1	1.5	0	0	1	1.5	1	1.5
T18	Ease of Maintenance	1.14	1	1.14	1	1.14	1	1.14	1	1.14	1	1.14
Summation of EUD_Technical factors		EUD_TF	project1	14.57	project2	22.02	project3	10.76	project 4	18.32	project 5	15.67

5.2. EUD technical complexity factor

EUD Technical Complexity Factor (EUD_TCF) is calculated by the formula
EUD_TCF = 0.6 + (0.01 * EUD_TF)

Where EUD_TF=T1+T2+… +T18

Table 8: (EUD_TCF)

EUD_TCF = 0.6 + (0.01 * EUD_TF)		
Project	EUD_TF	EUD_TCF
Project1	14.57	0.75
Project2	22.02	0.82
Project3	10.76	0.71
Project4	18.32	0.78
Project5	15.76	0.76

5.3. Calculate EUD_environmental factors (EUD_EF) for five projects

If the EUD_TF is required for the particular project it is rated as [1] else 0, and is multiplied by the assigned weight of EUD_TF. Take the summation of all factors. Following result is obtained.

Table 9: (EUD_EF)

Fi	EUD_Environmental factors	Weight	Project1	Value	Project2	Value	Project3	Value	Project 4	value	Project 5	value
F1	Content Level of EUP	1.4	0	0	1	1.4	1	1.4	1	1.4	0	0
F2	End User Computing Capability	0.25	1	0.3	1	0.25	0	0	1	0.25	0	0
F3	Ease of Use & Feedback	1.2	1	1.2	1	1.2	1	1.2	1	1.2	0	0
F4	Inbuilt System Assistance for EUP	1.25	1	1.3	1	1.25	0	0	0	0	1	1.25
F5	Training & learning Time Constraint for end user	1.12	1	1.1	1	1.12	1	1.12	0	0	1	1.12
F6	Reliability of End User Code	1.2	0	0	1	1.2	1	1.2	1	1.2	1	1.2
F7	End User Storage Constraint	1.02	1	1	1	1.02	0	0	0	0	0	0
F8	Risk Factors	1.12	0	0	1	1.12	0	0	0	0	1	1.12
Summation of EUD_Environmental factors		EUD_EF	Project1	4.8	Project2	8.56	Project3	4.92	Project4	4.05	Project5	4.69

5.4. EUD environmental complexity factor (EUD_ECF)

EUD Environmental Complexity Factor (EUD_ECF) is calculated by the formula

EUD_ECF = 1.4 + (0.03 * EUD_EF)

Where EUD_TF= F1+F2+F3+ +F8

Table 10: (EUD_ECF)

EUD_ECF = 1.4 + (0.03 * EUD_EF)		
Project	EUD_EF	EUD_ECF
project1	4.84	1.55
project2	8.56	1.66
project3	4.92	1.55
project4	4.05	1.52
project5	4.69	1.54

5.5. Advance use case point (AUCP) is calculated as given below

AUCP = UCP X (EUD_TCF X EUD_ECF)

Table 11: (AUCP)

AUCP = UCP X (EUD_TCF X EUD_ECF)				
Project	UCP	EUD_TCF	EUD_ECF	AUCP
Project1	120.30	0.75	1.55	138.62
Project2	159.15	0.82	1.66	216.27
Project3	153.20	0.71	1.55	167.77
Project4	202.54	0.78	1.52	241.35
Project5	206.85	0.76	1.54	241.44

6. Result analysis

Calculating the difference between the results of AUCP and UCP we obtain the values of table.12.

Table 12:

UCP	AUCP	difference	%increase
120.30	138.62	18.32	15.23
159.20	216.27	57.07	35.85
153.20	167.77	14.57	9.51
202.54	241.35	38.81	19.16
206.85	241.44	34.59	16.72

These results show that the difference of AUCP and UCP is not zero. There is clear indication of increase in AUCP. It is clearly depicted in the graph Fig.1 given below.
We can further analyze the result by using paired t-test on the above values of UCP and AUCP.

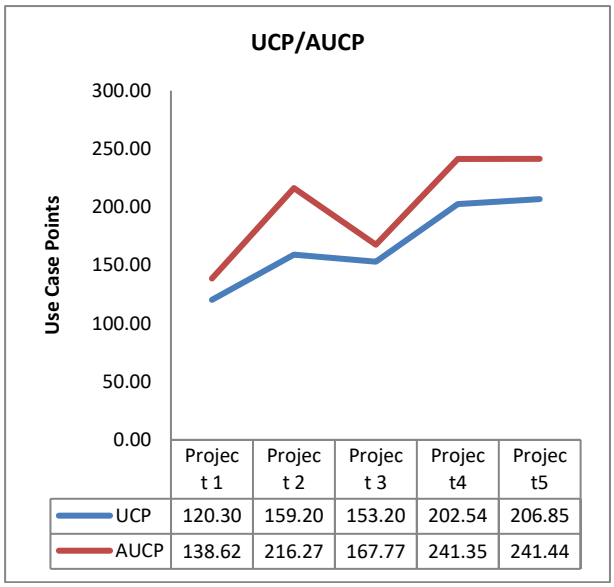

Fig. 1: Paired Sample T-Test.

The paired sample t-test, sometimes called the dependent sample t-test, is a statistical procedure used to determine whether the mean difference between two sets of observations is zero. In a paired sample t-test, each subject or entity is measured twice, resulting in pairs of observations.
Hypothesized difference (D): zero
Significance level (percentage): five
Summary statistics:

Variable	Observations	Obs. with missing data	Obs. without missing data	Minimum	Maximum	Mean	Std. deviation
AUCP	5	0	5	138.616	241.442	201.090	46.080
UCP	5	0	5	120.300	206.850	168.418	36.309

T-Test for two paired samples / Two-tailed test:
95% confidence interval on the difference between the means: [11.430, 53.914], -5.699]

Table 13:

Difference	32.672
t (Observed value)	4.270
|t| (Critical value)	2.776
DF	4
p-value (Two-tailed)	0.013
alpha	0.05

The number of degrees of freedom is approximated by the Welch-Satterthwaite formula.
Test interpretation:
H0: The difference between the means is equal to zero.
Ha: The difference between the means is different from zero.
As the computed p-value is lower than the significance level Alpha=0.05, one should reject the null hypothesis H0, and accept the alternative hypothesis Ha.
The risk to reject the null hypothesis H0 while it is true is lower than 1.29%.

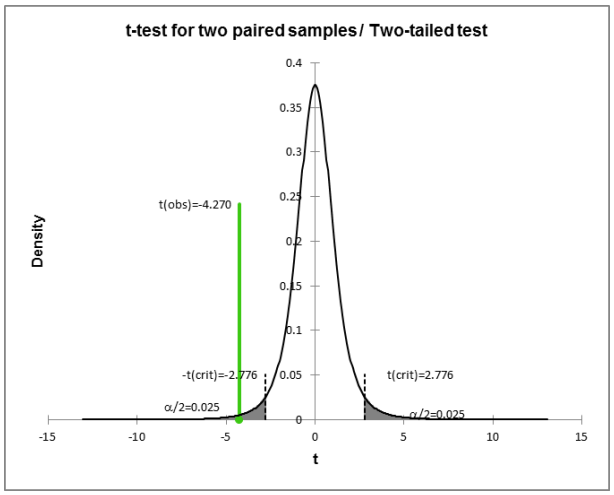

Fig. 2:

7. Conclusion

The results of this study provide further support for the view that when EUD features are incorporated in the system or tools it increases the size and effort. The extra interfaces and quality attributes are provided on demand to the products developed by end user for their own usage. From the result of the case study it has been found that there can be on an average maximum 20% increase in overall effort of development. End User Software Engineering (EUSE) takes care of all quality requirements of these products. End User Development enhances the End user satisfaction level as users are involved throughout the development process starting from the requirement gathering to designing phase. As the demand of EUD features are increasing day by day its role in the overall budget cannot be ignored. As over budgeting and under budgeting both have critical consequences on the successful completion of the project. Unlike many other estimation methods, the concepts and methods behind AUCP are openly described and available for further investigation. This method will give some benefits to the developers in improving the accuracy of software effort estimation. Hence including EUD features as an additional cost driver during the estimation process will reduce the level of uncertainty involved in estimation process.

References

[1] Lieberman, H., Paterno, F., Klann, M. and Wulf, V. 2006. End-user development: An emerging paradigm. End User Development. (2006), 1–8. https://doi.org/10.1007/1-4020-5386-X.

[2] Albrecht, A. (1979). Measuring Application Development Productivity. In: Proc. of IBM Applications Development Symposium,pg. 83–92, October.

[3] Ghiani, G.; Paternò, F.; Spano, L.D.; Pintori, G. An environment for End-User Development of Web mashups. Int. J. Hum. Compu. Stud. 2016, 87, 38–64. https://doi.org/10.1016/j.ijhcs.2015.10.008.

[4] Resource Estimation for Objectory Projects Gustav Karner Objective Systems SF AB Torshamnsgatan 39, Box 1128, 164 22 Kista, September 17, 1993.

[5] Srivastava, Archana, Syed Qamar Abbas, and S. K. Singh. "ENHANCEMENT IN FUNCTION POINT ANALYSIS." airccse.org/journal/ijsea/papers/6215ijsea01.pdf.

[6] Srivastava, A., Singh, S.K. and Abbas, S.Q., ADVANCEMENT OF UCP WITH END USER DEVELOPMENT FACTOR: AUCP, International Journal of Software Engineering & Applications (IJSEA), Vol.6, No.2, March 2015 https://doi.org/10.5121/ijsea.2015.6201.

[7] Gelderman, M. (1998). The relation between user satisfaction, usage of information systems and performance. Information & Management, 34, 11-18. https://doi.org/10.1016/S0378-7206(98)00044-5.

[8] Lieberman, H., Paterno, F., Klann, M. and Wulf, V. 2006. End-user development: An emerging paradigm. End User Development. (2006), 1–8. https://doi.org/10.1007/1-4020-5386-X.

[9] Ko, A.J., Abraham, R., Beckwith, L., Blackwell, A., Burnett, M., Erwing, M., Scaffidi, C., Lawrance, J., Lieberman, H., Myers, B., Rosson, M.B., Rothermel, G., Shaw, M. and Wiedenbeck, S. 2010. The state of the art in end-user software engineering. ACM Computing Surveys. (2010).

[10] Estimating With Use Case Points, Mike Cohn, Mountain Goat Software, www.mountaingoatsoftware.com.

[11] Carroll, Edward R. "Estimating Software Based on Use Case Points." 2005 Object-Oriented, Programming, Systems, Languages, and Applications (OOPSLA) Conference, San Diego, CA, 2005.

[12] B. Anda," Comparing Effort Estimates Based on Use Case Points with Expert Estimates," 2007.

[13] C. Gence, L. Buglione, O. Demirors, P. Efe," A Case Study on the Evaluation of COSMIC-FFP and Use Case Points," 2006 .

[14] Marcio Rodrigo Braz, Silvia Regina Vergilio. (2006). Software Effort Estimation Based on Use Cases, In: Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC'06). IEEE. M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989. https://doi.org/10.1109/COMPSAC.2006.77.

[15] Iraji, Mohammad Saber, and Homayun Motameni. "Object Oriented Software Effort Estimate with Adaptive Neuro Fuzzy use Case Size Point (ANFUSP)", International Journal of Intelligent Systems and Applications, 2012.

[16] Prabhakararao S., Cook C., Ruthruff J., Creswick E., Main M., Durham,M., And Burnett M. 2003. Strategies and behaviors of end-user programmers with interactive fault localization. IEEE Symposium on Human-Centric Computing Languages and Environments, Auckland, New Zealand, September, 15-22. https://doi.org/10.1109/HCC.2003.1260197.

[17] C. Schroth and O. Christ, "Brave new web: Emerging design principles and technologies as enablers of a global soa," in Proceedings of the IEEE International Conference on Services Computing, 2007. SCC 2007. Los Alamitos, CA, USA: IEEE Computer Society Press, 2007, pp. 597–604. https://doi.org/10.1109/SCC.2007.32.