An Analysis of Use Case Point Method

Monika
Research Scholar
DCSE, GJUST, Hisar, India

***Abstract*—Different methods for effort estimation has been discovered and experimented. Some gives better result but some failed. The basic goal is to find an estimation method with minimal gap between actual and estimated effort. One of these methods is UCP. It is an OO as well as easy to implement. Different researchers experiment it with different techniques to minimal the gap. In this paper, some of the core experiments are taken into account and are discusses to point out the main observation.**

Index Terms : Use case point, effort estimation, technical complexity factor, environment complexity factor

I. INTRODUCTION

Time and effort used up during the process of software testing regulates the excellence of the software. Effort estimation concludes the required man power, time span to be required to complete the project successfully. The failures of Software project have been a significant obstacle in the last few decades. However, this failure occurs usually during planning and estimation steps. From 2002 to 2010, there are reported only 37% software project accomplished successfully. However, is the lack of good planning that plays most crucial and largest role in projects failure of the project, which is about 39% [2].

There is need to software project planning for better result in the delivery of the software project. Hence accurate cost estimation is become a challenge for IT industries. The methods and techniques had been put into practice to estimate the software effort ranging from traditional modeling. In this paper, we have tried to compare various estimation methods that have been introduced in the past two decades and how the accuracy of software estimation has been improved with these techniques and by how much.

A brief introduction of various methods has been given in chronological order. First ever reliable methods used were Function Point Analysis and Test Case Point Analysis. After Use Case Point method which gave us such reliable results. Since then there have been many modifications introduced in UCP which have further improved the accuracy of our estimation. This paper is basically an overview of accuracy level of different estimation methods.

A. USE-CASE POINTS

Use Case Points (UCP) is a software estimation technique which is used to determine the software size with help of use cases. The quantity of UCPs in a depends on various factors –

- use cases present in the project system.
- actors present on the project system.
- Nonfunctional requirements.
- The environment in which the project will be developed.

There are six major components which are important for determining the size of a project [21].

- **Calculate Unadjusted Actor Weights (UAW)**:Firstly, the actor present in the system need to be identified and classify as per the Table1.

The UAW is calculated as:

$$UAW = \#\,Simple\ Actors\ \times 1 + \#\ Avg\ Actors\ \times 2$$
$$+\#\,Complex\ Actors\ \times 3$$

Table 1Actor Type, Weight and Explanation

Weight	Factor wt	Criteria / explanation
Simple	1	Interaction occurs via API, as command prompt.
medium	2	Interaction occurs via protocol, like TCP/IP.
Complex	3	Interacts via an interface GUI)

- **Determine Unadjusted Use Case Weights (UUCW):** The use cases identified as mentioned in the Table 2.

Table 2 Use Cases Types, their Weight and Explanation

Wt type	Wt. factor	Criteria / Description

Simple	005	Using ≤ 03 transactions
Medium	010	Using 04-07 transactions
Complex	015	Using >07 transactions

The UUCW is calculated used formula given below:

$$UUCW = \#\, Simple - use - cases \times 5 + \#\, Average - use - cases \times 10 + \#\, Complex - use - cases \times 15$$

- **Compute *Unadjusted Use Case Points (UUCP)*** :

UUCP = UAW + UUCW

- **Define *Technical Environmental Factor (TEF)***

There are about 13-17 factors in any project which are multiplied by their weights and their summation is called TFactor.

Table 3 Technical Complexity Factor

Factor	Description	Weight
T_1	Distributed_system	02
T_2	Response or throughput Performance objectives	02
T_3	End-user efficiency	01
T_4	Complex internal Processing	01
T_5	Reusable code	01
T_6	Easy to install	0.5
T_7	Easy use	0.5
T_8	Portable	02
T_9	Easy to change	01
T_10	Concurrent	01
T_11	Includes security Features	01
T_12	Provide access for third Parties	01
T_13	Special user training Facilities are required	01

Finally, the following formula is applied to calculate TCF:

$$TCF = 0.6 + (0.01 \times TFactor)$$

A numerical value between 0-5 is usually assigned to all the factors of TCF as per their significance on the project.

- **Calculate *Environmental Complexity Factors (EFA*)**

There are about 8-10 environmental factors in any project which are multiplied by their weights and their summation is called EFactor. Finally, to calculate EF:

$$ECF = 1.4 + (-0.03 \times EFactor)$$

A numerical value between 0-5 is usually assigned to all the factors of EFA as per their significance on the project.

.

Table 4 Environmental Complexity Factor

Factor	Explanation	Weight
E_1	Familiar with rational unified process	1.5
E_2	Application experience	0.5
E_3	Object oriented experience	1
E_4	Lead analyst capability	0.5
E_5	Motivation	0
E_6	Stable requirements	2
E_7	Part-time workers	-1
E_8	Difficult programming language	-1

- **Calculate Adjusted Use Case Point (AUCP)**

$$AUCP = UUCP \times [0.65 + (0.01 \times TEFactor)]$$

- **Compute final effort using a conversion factor.**

$$Total\ Actual\ Effort = AUCP \times CF$$

II. Interactive Use Case Point (I-UCP)

In the original UCP, three type of actor weights are used usually.. To reflect the interaction complexity, It identify six actor weights [8,9].

Table 5 Actor Type, Weight and Description in i-UCP

Weight	Factor wt	Criteria / description
Simple_actor (system)	1	Interaction occurs via API, as command prompt.
Average_actor (system)	2	Interaction occurs via protocol, like TCP/IP.
Complex_actor (system)	3	Communication occurs via a complex protocol or data store
Simple_actor (human)	3	User role supported
Average_actor (human)	4	Supported by two or three user roles or one focal role
Complex_actor (human)	5	Supported by >3 user roles or >1 focal role

B. Extended Use Case Point (e-UCP)

The extended use case point method (e-UCP) represents an altered version of the UCP method and was first projected by Kasi Perivasamy and Aditi Ghade in 2009 [6, 7].

This method focuses on the interior particulars of a use case. And it does so by including use case narrative as an important for effort calculation. The first step in this method is compute the of the system functionality via use case model called Unadjusted e-Use Case Point (e-UUCP). .

Table 6 Actor Weight

Actor Weight	Factor wt	Actor Weight	Factor wt
Very Simple	0.50	Complex	2.50
Simple	1.00	Very Complex	3.00
Less Avg.	1.50	Most Complex	3.50
Avg.	2.00		

Once we are done with classification of actors and use case weight, next step is to assign the use case narrative weights to the different parameters. The table 7 given below is the use case narrative weight classification

Table 7 Use Cases Weight

Use Cases Weight	Factor wt
Simple	0.5
medium	1.0
Complex	2.0
Most Complex	3.0

Table 8 Use Cases Narrative weight classification

Use Case Narrative Parameter	Factor
Input Parameter	0.10
Output Parameter	0.10
A Predict in Precondition	0.10
A Predict in Post-condition	0.10
An Action in Successful Scenario	0.20
An Exception	0.10

e-UCP = UUCP * TCF * EF

C. Re-USE CASE POINTS (Re-UCP) :

Re-UCP is a generic structure that adapts behavior for different projects with varying degree level of complexity and futuristic order of scalability. Critical actors and critical use cases added in the actor types and use cases types. Table 9 and Table 10 show the type assigned value for actor and use cases.

Table 9 Re-UCP Actor Weight

Weight	Factor (weightage)	Count of Actors	Weight * Count

Simple	1.0	a	1.0 x a
Medium	2.0	b	2.0 x b
Complex	3.0	c	3.0 x c
Critical	4.0	d	4.0 x d

As per Table 9. And Table 10

$$\boldsymbol{Re - UAW = a + 2b + 3c + 4d}$$

$$\boldsymbol{Re - UUCW} = 5.0a + 10.0b + 15.0c + 20.0d$$

Table 10 calculation of use case points

Weight	No of Transactions	Weight	Use Case Count	Weight * Count
Simple	≤ 04	05.0	a	5.0 x a
Medium	05 – 08	10.0	b	10.0 x b
Complex	09 – 15	15.0	c	15.0 x c
Critical	>15	20.0	d	20.0 x d

First of all, calculation of total actors weight ant and total use cases weight is done.

$$Re - UUCP = Re - UAW + Re - UUCW$$

In Re-UCP, we use a total of 14 TCF parameters as compared to 13 as in case of UCP and e-UCP. The newly added 14th parameter is termed as scalability and is labeled as T_{14}. The Scalability of a system refers to the capability of the system by which it maintains any sudden increased workloads without any further addition of new existing resources in the respective system.

A numerical value between 0-5 is usually assigned to all the factors of TCF as per their significance on the project. The numerical value of ‘0’ indicates that the parameter is irrelevant and numerical value of ‘1-5’ indicates that the parameter is relevant to system. Larger the number means higher the importance. The numerical value of ‘0’ indicates that the parameter is treated as essential for system project.

$$TCF = 0.6 + (0.01 \times TFactor)$$

In Re-UCP, we use a total of 9 TCF parameters in e-UCP The newly added 14th parameter is termed as Project Methodology which defines the expertise of the developer assigned to concerned project in the project methodology and is labeled as T14.

A numerical value between 0-5 is usually assigned to all the factors of EFA as per their significance on the project. The numerical value of ‘0’ indicates that the developer assigned to the project has no previous experience in corresponding parameter. The numerical value of ‘1-5’ indicates the degree of developer experience

$$EF = 1.4 + (-0.03 \times EFactor)$$

Calculation of the revised use case points is done as : :

$$Re - UCP = Re - UUCP \times TCF \times EF$$

Finally, effort is estimated by conversion of UUCP to Man-Hours.

$$Effort = UCP \times PHper\ UCP$$

Where, PHper UCP is person-hours per UCP

D. Experimented on different data

Below, the example used in [22] has been taken into account in the calculation calculated below.

In this paper, Caroll calculated the average project size as 60 man-months using Use Case Point method. The team consisted of 6-12 members in each project. 200 projects completed in the period of 5yrs.By using resources, it increased more than 250 projects due to changing customers and priorities continually. Table 11 and table 12 represents the actor weight and use cases weight calculation.

Table 11 weighing actors for complexity

Actor Type	Description	Qnty	Weight Factor	Sub Total
Simple	Defined API	03	1.0	30
Average	Interactive or Protocol Driven Interface	02	2.0	4.0
Complex	Graphic User Interface	01	3.0	3.0
Total UAW				10.0

*Table 12*weighing use cases for complexity

Use Case Type	Description	Qnty	Weight Factor	Sub Total
Simple	3 or fewer transaction	03	5.0	15.0
Average	4 – 7 Transaction	02	10.0	20.0
Complex	Greater than 7 transactions	01	15.0	15.0
Total UCW				50.0

UUCP = 10.0 + 50.0 = 60.0

A total of 13 technical factors were selected with weightage assigned between 0-5 (Table 12) and 8 environmental factors were calculated with weight factor (-1 to +2) (table 13).

- **Calculation of TCF :**

$$TCF = 0.6 + (0.01 \times TFactor)$$
$$\Rightarrow 0.6 + (0.01 \times 42) = 1.02$$

Table 13 weighing technical factors

Factor	Weight	Project Rating	Sub Total
T_1	2.0	5	10.0
T_2	1.0	3	3.0
T_3	1.0	5	5.0
T_4	1.0	5	5.0
T_5	1.0	5	5.0
T_6	1.0	3	3.0
T_7	0.5	3	1.5
T_8	2.0	0	0.0
T_9	1.0	5	5.0
T_10	1.0	0	0.0
T_11	1.0	5	5.0
T_12	1.0	0	0.0
T_13	1.0	3	3.0
		TFactor	42.0

Table 14 weighing experience factors

Factor	Weight	Project Rating	Sub Total
E_1	1.0	4	4.0
E_2	0.5	2	1.0
E_3	1.0	4	4.0
E_4	0.5	4	2.0
E_5	0.0	4	0.0
E_6	2.0	2	4.0
E_7	-1.0	0	0.0
E_8	-1.0	3	-3.0
		EFactor	12.0

- **Calculation of EF :**

$$EF = 1.4 + (-0.03 \times EFactor)$$
$$\Rightarrow 1.4 + (-0.03 \times 12.0) = 1.04$$

- **Calculate UCP**

$$UCP = UUCP \times TCF \times EF$$

$$\Rightarrow 60 \times 1.02 \times 1.04 = 63.648$$

- **Calculate man-hours**

$$Total\ man - hours = ER \times UCP$$
$$\Rightarrow 20 \times 63.648 = 1272.96\ man - hours$$
OR
$$\Rightarrow 28 \times 63.648 = 1782.144\ man - hours$$

E. Implementation II

To introduce the graduation in assigned value of use case, triangular membership function has been applied. Fig. 2, fig.3 shows the I/O membership resp.16].

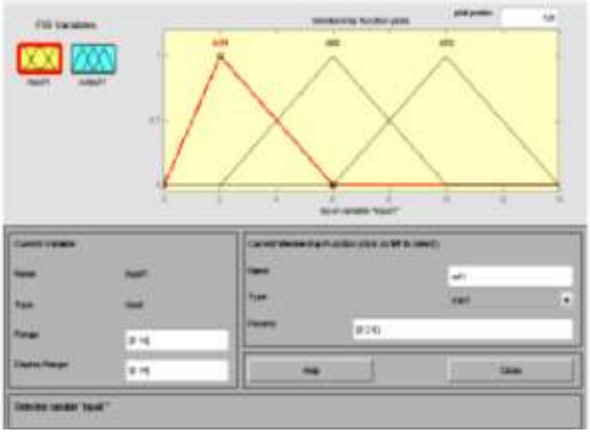

Figure 1 Fuzzy Logic Input Membership

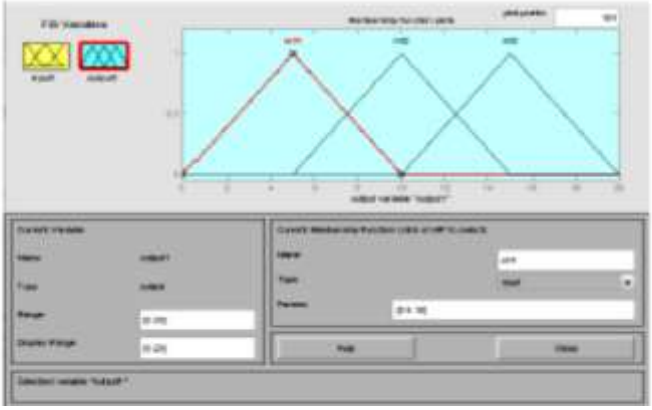

Figure 2 Fuzzy Logic Output Membership

Table 15 Adjusted Weight of Use Cases

Use Case Transactions	Karner's Weight	Adjusted Weight
1 Trans-action	5.00	5.00
2 Trans-actions	5.00	5.00
3 Trans-actions	5.00	6.45
4 Trans-actions	10.00	7.5.
5 Trans-actions	10.00	8.55
6 Trans-actions	10.00	10.00
7 Trans-actions	10.00	11.40
8 Trans-actions	15.00	12.50
9 Trans-actions	15.00	13.60
10 Trans-actions	15.00	15.00

Table 15 clearly shows the variations of the assigned value on the use cases after application of fuzzy logic. It is beneficial to get an graduation rather than an fixed assigned value.

F. Evaluation

Karner, in his UCP model, did not pay any attention to the "extend" and "include" use cases. However, we have concluded that the "extend" and "include" use cases play a crucial role in effort estimation of a project. It is essential to estimate the software size. The use of fuzzy logic approach in effort estimation is calculated in 3 stages.

Firstly, the assessment of effort estimation was done on a 7 projects. In the UC model, there is not any very few "include/exclude" use cases. In the second stage, five projects were assessed that contained at least between 15% to 25% of "extend" and "include" use cases out of total. Finally, in the last stage, the assessment of effort estimation was done on a total of eight projects. It contained at least more than 25% of "extend" and "include" use cases out of total.

The results obtained in 1st, 2nd and 3rd stage using fuzzy logic approach are given in table 16.

Table 16 result evaluation of three stages using fuzzy logic

S. NO	STAGE	RESULT
1	1stStage	22% improvement in MMRE 9% improvement in MMER
2	2nd Stage	4% improvement in MMRE 2% improvement in MMER
3	3rd Stage	Negative effect

The comparison karner and fuzzy logic approach is shown in Table 17.

Table 17 Comparison of Karner's model and the Proposed Models

Project	Actual Size UUCP	Karner's Estimation	PropoSed Model (Fuzzy)	MRE Karner	MRE Fuzzy Logic	MER Karner	MER Fuzzy Logic	Error Karner (Karner–Actual)	Error Fuzzy (Fuzzy–Actual)
Project 1	72.44	128.96	104.98	0.78	0.45	0.44	0.31	56.52	32.54
Project 2	74.33	128.54	108.65	0.73	0.46	0.42	0.32	54.21	34.32
Project 3	55.50	51.00	48.70	0.08	0.12	0.09	0.14	-4.50	-6.80
Project 4	68.00	108.50	92.40	0.60	0.36	0.37	0.26	40.50	24.40
Project 5	48.75	74.25	61.25	0.52	0.26	0.34	0.20	25.50	12.50
Project 6	94.50	168.75	144.00	0.79	0.52	0.44	0.34	74.25	49.50
Project 7	72.50	108.41	92.44	0.50	0.28	0.33	0.22	35.91	19.94
Mean				0.57	0.35	0.35	0.26	40.34	23.77
Standard Dev								25.33	17
Improv-ement				+22%		+9%			
Project 8	96.80	81.05	74.82	0.16	0.23	0.19	0.29	-15.75	-21.98
Project 9	79.80	98.67	84.54	0.24	0.06	0.19	0.06	18.87	4.74
Project 10	91.50	118.45	109.75	0.29	0.20	0.23	0.17	26.95	18.25
Project 11	86.58	63.21	65.12	0.27	0.25	0.37	0.33	-23.37	-21.46
Project 12	188.64	132.54	128.67	0.30	0.32	0.42	0.47	-56.10	-59.97
Mean				0.25	0.21	0.28	0.26	-9.88	-16.08
Standard Deviation								33.67	30.01
Improv-ement				+4%		+2%			
Project 13	94.36	54.88	48.44	0.42	0.49	0.72	0.95	-39.48	-45.92
Project 14	87.44	52.87	46.55	0.40	0.47	0.65	0.88	-34.57	-40.89
Project 15	111.50	75.84	62.54	0.32	0.44	0.47	0.78	-35.66	-48.96
Project 16	119.88	67.84	72.59	0.43	0.39	0.77	0.65	-52.04	-47.29
Project 17	144.60	86.17	74.85	0.40	0.48	0.68	0.93	-58.43	-69.75
Project 18	102.87	82.40	72.88	0.20	0.29	0.25	0.41	-20.47	-29.99
Project 19	124.60	64.21	52.62	0.48	0.58	0.94	1.37	-60.39	-71.98
Project 20	168.65	72.89	61.25	0.57	0.64	1.31	1.75	-95.76	-107.40
Mean				0.40	0.47	0.72	0.97	-49.60	-57.77
Standard Deviation								23.00	24.47
Improv-ement				-7%		-25%			

G. Implementation III

Fig 3 shows the mapping between uses cases and actors as input vector and UUCP as output vector.One hidden layer along with Multi-Layer Perceptron was utilized as an approach to solve the problem. A total of thirteen input factors were determined in which ten vectors represented use cases and three vectors represented actors. The algorithm used for training purposes in this method was Levenberg-Marquardt back propagation (trainlm). After performing a series of experiments, he number of neurons present in the hidden layer were evaluated.

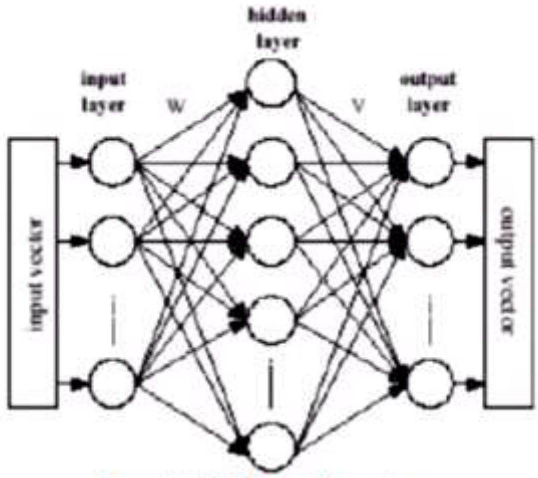

Figure 4: Multi Layer Perceptron

However, evaluation of neurons present in hidden layer can be done only via trial and error method [21]. A total of twelve experiments were performed, with each one having a hidden layer with 14-25 neurons. However, the experiments having 20 neurons in hidden layer produed best results among all. The neural network was first trained using seven projects and in the next step, testing and validation was done using another thirteen projects. In the next step, we describe the outcome of application of the neural network approach.

H. Evaluation

A total of thirteen projects were evaluated and tested using neural model. The training process shows better results. The neural network approach showed an improvement of 20% in the MMER.

Table 18 Comparison of Karner's method and Neural Network Approach

	MRE	MER	Error	SD
Karner	0.44	0.51	36.15	23.66
Neural Network	0.79	0.31	49.45	33.89
Improvement	-35%	20%		

Mean Error = 0.0215

Standard Deviation = 0.0616

- **Implementation IV**

The web based software considered for this experiment [21].

Table 19 Calculation of UAW

Actor Type	No of Actor	Weight Factor	Sub Total
Simple	00	1.0	00
Average	32	2.0	64
Complex	00	3.0	00
Total UAW			64

Table 20 Calculation of UAW

Use Case Type	Type	Factor	Sub-UUCW
Simple	02	5.0	10.0
Average	01	10.0	10.0
Complex	01	15.0	15.0
Very Complex	01	20.0	20.0
Total UUCW			55.0

$$UUCP = UAW + UUCW = 64 + 55.0 = 119$$

Table 20 Technical factors

Factor	**Assigned Value**	**Weight**	**Extended Value**
T_1	5	3.0	15.0
T_2	5	5.0	25.0
T_3	2	1.0	2.0
T_4	3	1.0	3.0

T_5	3	2.0	6.0
T_6	4	4.0	16.0
T_7	2	1.0	2.0
T_8	4	2.0	8.0
T_9	5	2.0	10.0
Total			87.0

- **Calculation of AUCP**

$$\boldsymbol{AUCP = UUCP \times (0.65 + 0.01 \times TEF)}$$
$$\Rightarrow \mathbf{119 \times (0.65 + 0.01 \times 87.0) = 180.88}$$

- **Calculation of Final Effort**

$$\boldsymbol{Final\ Effort = AUCP \times Conversion\ Factor}$$
$$\Rightarrow \mathbf{180.88 \times 13 = 2351.4}$$

- **Efforts**

$$\boldsymbol{Effort = Final\ Effort + Project\ Complexity}$$
$$\boldsymbol{+ management}$$
$$\Rightarrow Final\ Effort + \frac{15}{100} \times Final\ Effort + \frac{10}{100} \times Final\ Effort$$
$$\Rightarrow \frac{125}{100} \times Final\ Effort = \frac{125}{100} \times 2351.4$$
$$= 2939.304 = 367\ man - days$$

Actual effort = 390 man days

- **Magnitude of Relative Error**

$$\textbf{MMRE} = \frac{|Actual\ Effort - Predicted\ Effort|}{Actual\ Effort} * \mathbf{100}$$
$$\Rightarrow \frac{|390-367|}{390} * \mathbf{100} = \mathbf{5.8\ \%}$$

<u>Application of PSO on UCP :</u>

By applying the particle optimization principle on UCP factors a total of 16 parameters values. Initially, 10 particles were taken & the range is calculated by local and global best approach. Table 21 represents the initial values of some particles.

The initial efforts with 10 particles are calculated as 3633.50, 2359.50, 3633.50 and so on.

Table 21 Range Taken for Weights of Parameters

Parameters	**Range**
Actor	01 – 03
Use Cases	05 – 20
Technical Factors	01 -05

A total of 50 iterations are considered in this experiment with local and global best position of the particle and also updating in iteration. Get updated, and the swarm moves towards optimality.

Table 22 Particles's Initial Value

Parameter	**1st Particle**	**2nd Particle**	**3rd Particle**
SimpleActor	01	01	03
AverageActor	02	02	01
ComplexActor	03	02	03
SimpleUseCase	05	06	10
AverageUseCase	10	09	05
ComplexUseCase	15	14	07
VeryComplexUseCase	20	19	18
T_1	03	01	03
T_2	05	02	02
T_3	01	03	04
T_4	01	04	01
T_5	02	05	05
T_6	04	03	03
T_7	01	04	04

T_8	02	02	01
T_9	02	03	01

At the end, the estimation of total man hour will be needed is 3032 man hrs or 379 man days.

As per the definition of MMRE. It is calculated as

$$\textbf{MMRE} = \frac{|390-379|}{390} * 100 = 2.8\ \%$$

I. CONCLUSION

The above experiment shows that the use case point is very effective on the effort estimation. This paper considered some of the optimization techniques. And you can see the vast difference between the effort estimation. As we implement a technique; it improves with a clear difference. It clearly give us an easy approach. Still, we can improve it further using the new developed techniques.

J. REFERENCES :

[1.] Pichai Jodpimai, Peraphon Sophatsathit, and Chidchanok Lursin- sap, "Analysis of Effort Estimation based on Software Project Models", IEEE, 2009.
[2.] Jyoti G. Borade, Vikas R. Khalkar.Software Project Effort and Cost Estimation Techniques. Borade et al., International Journal of Advanced Research in Computer Science and Software Engineering 3(8), August - 2013, pp. 730-739
[3.] Matthias Kerstner, " Software Test Effort estimation Methods", 2 February 2011.
[4.] Periyasamy, K., and Ghode, A., Cost estimation using extended use case point (e-UCP) model. In Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference IEEE, 2009, (pp. 1-5).
[5.] M M Kirmani, A Wahid. Use Case Point and e-Use Case Point method of software effort estimation: A critical performance comparison. International Journal of Computer Application (2250-1797) Volume 5– No. 3, April 2015
[6.] Nunes, N.J., Constantine, L., and Kazman, R. iUCP: Estimating Interactive-Software Project Size with Enhanced Use-Case Points. Software, IEEE 28, 4 (2011), 64–73.
[7.] R. Alves; P. Valente; N. Jardim Nunes. Improving Software Effort Estimation with Human-Centric Models: a comparison of UCP and iUCP accuracy. EICS'13, June 24–27, 2013, London, United Kingdom. June 2013.DOI: 10.1145/2480296.2480300
[8.] Burnett, Margaret M. and Scaffidi, Christopher (2013):End-UserDevelopment.In:Soegaard, Mads and Dam, RikkeFriis (eds.). "The EncyclopediaofHuman-Computer Interaction, 2nd Ed."
[9.] A. Srivastava, Dr.S.K.Singh and Dr.S. Q.Abbas.advancement of ucp with end user development factor: aucp. International Journal of Software Engineering & Applications (IJSEA), Vol.6, No.2, March 2015
[10.] logica Iberia. http://www.logica.pt/.
[11.] Gabinete de Desenvolvimento de AplicaçõesInformáticas. https://gdai.uma.pt/.
[12.] Valente, P. and Sampaio, P. Analysis of Interactive Information Systems Using Goals. Innovative Information SystemsModelling Techniques, InTech, (2012), Availablefrom: http://www.intechopen.com/books/innovativeinformation-systems-modelling-techniques/analysis-ofinteractive-information-systems-using-goals
[13.] Karner, G. Resource estimation for objectory projects. Objective Systems SF AB 17, (1993).
[14.] A B Nassif, L F Capretz, Danny HoEnhancing Use Case Points Estimation Method Using Soft Computing Techniques. Journal of global research in computer science. ISSN-2229-337X. 2010. 12-21.
[15.] R. P. Lippman, "An Introduction to Computing with Neural Nets," IEEE ASSP Magasine, vol. 3, pp. 4-22, 1987.
[16.] I. V. Tetko, D. J. Livingstone and A. I. Luik, "Neural network studies. 1. Comparison of overfitting and overtraining," J. Chem. Inf. Comput. Sci., vol. 35, pp. 826-833, 09/01, 1995.
[17.] Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: IEEE Conference on Neural Networks, Piscataway, NJ, pp. 1942–1948 (1995)
[18.] The PSO website, http://www.swarmintelligence.org/
[19.] Nageswaran, S.: Test Effort Estimation Using Use Case Points. In: 14th International Software/Internet Quality Week, San Francisco (2001)
[20.] Carroll, Edward R. "Estimating Software Based on Use Case Points." 2005 Object-Oriented, Programming, Systems, Languages, and Applications (OOPSLA) Conference, San Diego, CA, 2005.