Revised Use Case Point (Re-UCP) Model for Software Effort Estimation

Mudasir Manzoor Kirmani
Research Scholar, School of CS & IT
Maulana Azad National Urdu University
Hyderabad, India

Abdul Wahid
Dean and Head School of CS & IT
Maulana Azad National Urdu University
Hyderabad, India

***Abstract*—At present the most challenging issue that the software development industry encounters is less efficient management of software development budget projections. This problem has put the modern day software development companies in a situation wherein they are dealing with improper requirement engineering, ambiguous resource elicitation, uncertain cost and effort estimation. The most indispensable and inevitable aspect of any software development company is to form a counter mechanism to deal with the problems which leads to chaos. An emphatic combative domain to deal with this problem is to schedule the whole development process to undergo proper and efficient estimation process, wherein the estimation of all the resources can be made well in advance in order to check whether the conceived project is feasible and within the resources available. The basic building block in any object oriented design is Use Case diagrams which are prepared in early stages of design after clearly understanding the requirements. Use Case Diagrams are considered to be useful for approximating estimates for software development project. This research work gives detailed overview of Re-UCP (revised use case point) method of effort estimation for software projects. The Re-UCP method is a modified approach which is based on UCP method of effort estimation. In this research study 14 projects were subjected to estimate efforts using Re-UCP method and the results were compared with UCP and e-UCP models. The comparison of 14 projects shows that Re-UCP has significantly outperformed the existing UCP and e-UCP effort estimation techniques.**

***Keywords*—Use Case Point; Extended Use case point; Revised Use case Point; Software Effort Estimation**

I. INTRODUCTION

Software being indispensable and inevitable entity which is currently ruling almost all the modern day operability directly or indirectly having crucial attributes associated with it and failure of those can prove out to be of grave damage to different Industrial and societal parameters. Software cost estimation is one of the pivotal issues in modern software development industry making it the most important activity in software engineering and software project management domain [1]. Effort estimation is an activity to estimate the number of business activities of workers as well as how long it takes to accomplish a software development project. The effort estimation activity is very important to know how much relevant value of software is generated within the specified parameters. Accurate and reliable software development effort estimates have always been encouraging for project managers [10]. There are number of methods, tools and techniques which can be put into practice to estimate the cost of the software ranging from traditional modelling to modern day modelling.

Based on the literature available the different models like analogy based model, experience based model, LOC, KLOC, COCOMO and Function points, have played vital role in estimating effort of software development projects [4]. However, the requirement engineering spawned to higher levels of complexity these methods had to counter the challenging task of performing at higher levels of acceptability and scalability [6] [7]. To overcome this Use Case Points (UCP) were introduced to estimate the effort of the software development project in early stage of development [11] [18].

Most of the software development organizations are using Object-Oriented technology based approach for developing software. The basic building block of an object oriented design is Use Case diagram which is prepared in the early stages of design after clearly understanding the requirement [9]. A use case diagram is the simplest representation of a user's interaction with the system. These diagrams can portray different types of users and their interaction pattern with the system. Use Case Diagrams are considered to be useful for approximating early estimation of a software project. The use case point model for effort estimation was first proposed by Gustav Karner in 1993 [14], which was focused to predict the total amount of resources required for developing a software system with object-oriented technology in the early stages of software development process.

This new use case point method performed well in comparison to the other techniques in practice and has also gained wide popularity [3] [5] [12] [13] [24]. Researchers from academia as well as industry have shown interest in the Use Case point based approaches because of the promising results obtained along with their applicability in early steps of software development [17]. There have been a number of approaches proposed in the literature [12] [15] [16] [22] [23] [24]. However, there is no criteria that could be used to aid practitioners in selecting appropriate approaches which is suitable for estimation of efforts for different software development projects. Even though UCPs have played a challenging role in software effort estimation, further enhancement is needed in some of its corresponding parameters to ensure further improvement in bridging the gap between the actual and estimated efforts [1] [2] [19] [20].

This research work is an effort to propose a refined model based on UCP and e-UCP methods in order to improve the

efficiency of effort estimates for software development projects in the early stages of development. The rest of research paper gives an overview about UCP and e-UCP models in section II & III respectively. Section IV Re-UCP model for estimation of effort for software development projects is proposed with refinement in the complexity of actors and use cases. In section V experimental results of 14 projects are presented and discussed by highlighting the effectiveness of the proposed model. The conclusion and the proposal of future works are given in Section VI

II. Use Case Points Method

The UCP method is the extension of Function Point method with the benefit of requirement analysis in object-oriented process. It starts with measuring the functionality of the system based on the use case model in a count called Unadjusted Use Case Point (UUCP). The same technical factors are used as of Function Points. The UCPs shows an estimation of the size of the system which can be further mapped to man hours in order to calculate the effort required to develop a system.

Actors and use cases are classified into simple, average and complex categories according to their complexity and is assigned some weight factor. An actor is defined as "Simple", if it interacts with the system with the help of a defined application programming interface (API). An actor is defined as "Average", if it interacts with the help of an Interactive or Protocol-Driven Interface. The actor is defined as "Complex", if it interacts through a Graphical User Interface. The assigned weight factor for simple, average and complex are 1, 2 and 3 respectively.

Similarly use case is defined as "simple", if the number of transaction is less than 3, "average", if the number of transaction is between 4 and 7 and "complex", if the number of transaction is more than 7. The assigned weight factors for simple, average and complex are 5, 10 and 15 respectively.

After calculating UUCP, the use case points are calculated by multiplying UUCP to technical complexity factor (TCF) and Environmental factors (EF). The TCF corresponds of 13 different parameters and ECF corresponds of 8 parameters.

UUCP = UAW + UUCW

UCP = UUCP * TCF * EF

Further the effort is estimated by mapping the UCP with man-hours.

Effort = UCP * PHper UCP

Where PHper UCP is Person Hours per UCP.

III. Extended Use Case Point Method (E-Ucp)

The extended use case point method (e-UCP) is a revised version of UCP method and was proposed by Kasi Perivasamy and Aditi Ghade in 2009 [21]. The e-UCP model considers some additional information about the relationships between actors and use cases. The e-UCP is focussed on internal details of a use case by including the use case narrative in effort estimation process of a software development project in the early stages of development. It starts with measuring the functionality of the system based on the use case model in a count called Unadjusted Use Case Point (UUCP). The technical factors and environmental factors used were similar to UCP method of effort estimation. The e-UCPs shows an estimation of the size of the system which can further be mapped to man-hours in order to calculate the effort required to develop the system.

The categorization of actors was modified in e-UCP method of software effort estimation and the number of actor categories was increased from 3 to 7. The modified categories of actors were 'very-simple', 'simple', 'less-average', 'average', 'complex', 'more-complex', 'most-complex' and the corresponding weight assigned were 0.5, 1.0, 1.5, 2.0, 2.5, 3.0 and 3.5 respectively [21]. All the assigned values from 'very simple' to 'most complex' were multiplied by their corresponding weight factor and the summation of all calculated values is the actor weight.

Similarly the categorization was modified by increasing the number of use cases from 3 used in UCP to 4 in e-UCP [21]. The modified categories of use cases were 'simple', 'average', 'complex' and 'most complex' and the corresponding weight assigned was 0.5, 1.0, 2.0 and 3.0 respectively [21]. All the assigned values from 'simple' to 'most complex' were multiplied by their corresponding weight factor and the summation of all calculated values is the value of use case weight.

A new adjustment factor named as 'use case narrative' was added in extended use case point method of software effort estimation (e-UCP). The classified parameters for use case narrative were 'input-parameter', 'output-parameter', 'a-predict-in-precondition', 'a-predict-in-post-condition', 'an-action-in-successful-scenario' and 'an-exception' and the corresponding parameter weight were 0.1, 0.1, 0.1, 0.1, 0.2 and 0.1 respectively [21]. All the assigned values for different use case narrative parameters were multiplied by their corresponding weight factor and the summation of all calculated values is the value of use case narrative weight.

The calculated value of actor weight, use case weight and use case narrative weight was used in the following equation to calculate unadjusted use case points (UUCP).

Unadjusted Use Case Points (UUCP)

UUCP = Use Case Weight + Actor Weight + Narratives Weight

e-UCP = UUCP * TCF * EF

where TCF -Technical Complexity Factor

EF - Environmental Factor.

The number of parameters in TCF technical complexity factor and EF environmental factor used in e-UCP were same as in case of UCP.

Effort = e-UCP * PHper UCP

where PHper is Person Hours per UCP.

IV. Revised Use Case Point Method (Re-Ucp)

Re-UCP is an extension to UCP and e-UCP model wherein all the existing behaviour and implementation parameters of

the two models are pondered comprehensively in order to design the generic framework for software effort estimation which can have adaptable behaviour for all range of projects with varying level of complexity and have the futuristic scope of scalability which can handle the agile software development activities. In Re-UCP model the functionality of the system is measured by calculating all the use case points in the system. The functionality of the system is estimated by the collective impact of corresponding factors associated with actors of the system, behaviour of use case, impact of environment and role of technical factors over the use case point.

In Re-UCP, actors, use cases, environmental and other technical factors are further categorised to associate a particular impact factor on the specific use case activity rather than functionality of the system. Unlike UCP or e-UCP wherein either actor or use case is divided into simple, average, or complex with some weighting strategy, Re-UCP uses different categorization of actor and use cases. Actor in Re-UCP are categorised into simple, average, complex and critical with weight parameters of 1, 2, 3 and 4 respectively. Similarly use cases are also divided into simple, average, complex and critical categories and the detailed description of actors and use cases with their corresponding weighting factor is given in table [I].

TABLE I. ACTOR TYPE AND RESPECTIVE WEIGHT

Actor Type	Weight
Simple	1
Average	2
Complex	3
Critical	4

An actor is defined as "Simple", if it interacts with the system with the help of a defined application programming interface (API). An actor is defined as "Average", if it interacts with the help of an Interactive or Protocol-Driven Interface. The actor is defined as "Complex", if it interacts through a Graphical User Interface. The actor is defined as "Critical", if it interacts with modules wherein real time action is taken or complexity is very high. The weight parameters for simple, average, complex and critical are 1, 2, 3 and 4 respectively.

Similarly use case type is defined as "Simple", if the number of transaction is less than or equal to 4, "Average", if the number of transaction is between 5 and 8 , "Complex", if the number of transaction is between 9 and 15 and "Critical", if the number of transactions is greater than 15. The assigned weight factors for simple, average, complex and critical are 5, 10, 15 and 20 respectively and the same is given in table [II].

TABLE II. USE CASE TYPE AND RESPECTIVE WEIGHT

Use Case Type	No. of Transactions	Weight
Simple	<=4	5
Average	5 to 8	10
Complex	9 to 15	15
Critical	>15	20

Total Actor and Use case weight is calculated as:

UAW ---- Unadjusted Actor Weight

UAW =Σ (No. of actors * their respective weight factors) -------- (1)

UUUW ---- Unadjusted Use Case Weight

UUCW = Σ (No. of Use cases * their respective weight factors) -------- (2)

UUCP ---- Unadjusted Use Case Points

UUCP = UAW+UUCW ---------- (3)

The revised use case points are calculated as

Re-UCP = UUCP * TCF * ECF ---------- (4)

Where TCF-Technical Complexity Factor

ECF-Environmental Complexity Factor

In order to estimate the overall use case points of the system some other factors corresponding to development environmental and technical parameters need to be considered in development process and are called as Technical Complexity Factor (TCF) and Environmental Complexity Factor (ECF) respectively.

In the UCP and e-UCP models of effort estimation the TCF (Technical Complexity Factor) correspond of 13 different parameters which were assigned value from range 0 up to 5. The value '0' implies that the parameter is irrelevant and the assigned value will increase with the increase in significance. However, if the value is '5' then the significance of the corresponding parameter is treated as essential. In Re-UCP the number of parameters in TCF has been increased from 13 to 14 wherein scalability parameter was included as 14th parameter. The label for the 14th parameter in TCF is given as T14 and the value assigned for the parameter will be 0 in case of irrelevant up to 5 in case of essential.

Scalability can be defined as the ability of the system to handle increased workloads without adding resources to the existing system by repeatedly applying a cost-effective strategy for extending system capacity. By the progress of time the level of administering the projects varies with good levels of complexity. To make our system to be more adaptable to handle such dynamic projects is very much indispensable activity in present scenario and hence the scalability factor in Re-UCP has been incorporated to address this issue.

TCF is one of the factors used to integrate the predominance of the various enlisted technical factors of the system on the overall developmental process and simultaneously estimating the effect of impact on the overall software effort estimation process. All the technical factors from T1 to T14 are multiplied by their corresponding weight factor as described in the table [III] and the summation of all calculated values is the calculated value of technical complexity factor (TCF).

TABLE III. TECHNICAL COMPLEXITY FACTOR AND WEIGHT

Factor	Description	Weight
T1	Distributed system	2
T2	Response or throughput performance objectives	1
T3	End-user efficiency (online)	1
T4	Complex internal processing	1
T5	Code must be reusable	1
T6	Easy to install	0.5
T7	Easy to use	0.5
T8	Portable	2
T9	Easy to change	1
T10	Concurrent	1
T11	Includes special security features	1
T12	Provides direct access for third parties	1
T13	Special user training facilities are required	1
T14	Scalability	2

TF subsequently is used to obtain the value of the Technical Complexity Factor (TCF).

$$TCF = 0.6 + (0.01 * \sum_{i=1}^{14} TF_i) \quad \text{---------- (5)}$$

In the UCP and e-UCP models of software effort estimation the ECF (Environmental Complexity Factor) corresponded of eight parameters and were labelled from E1 to E8. Each parameter of ECF was assigned a value from the range 0 up to 5 where '0' implied that the developed had no experience of the corresponding parameter and if the developer experience was better than a higher value was assigned from the range. However, if the assigned value of the parameter was '5' then the developer was considered as expert. In Re-UCP the number of parameters in ECF was increased from 08 to 09. The ninth parameter included was project methodology which describes the experience of the developer in the project methodology selected for the development of the software project. The label for the ninth parameter in ECF is given as E9 and the value assigned for the parameter will be 0 in case of inexperienced developer up to 5 in case of expert developer.

All the environmental complexity factors from E1 to E9 are multiplied by their corresponding weight factor as described in the table [IV] and the summation of all calculated values is the value of environmental complexity factor (ECF).

TABLE IV. ENVIRONMENTAL FACTOR AND WEIGHT

Factor	Description	Weight
E1	Familiarity with the project	1.5
E2	Application Experience	0.5
E3	OO Programming Experience	1.0
E4	Lead Analyst Capability	0.5
E5	Motivation	1.0
E6	Stable requirements	2.0
E7	Part Time Staff	-1.0
E8	Difficult Programming Language	-1.0
E9	Project Methodology	1.0

EF value is used to obtain the Environmental Complexity Factor (ECF).

$$ECF = 1.4 + (-0.03 * \sum_{i=1}^{9} EF_i) \quad \text{-------- (6)}$$

The total number of all the revised use case points is calculated by multiplying UUCP, TCF and ECF

Re-UCP = UUCP * TCF * ECF---(from eq. 4)

The value of TCF is calculated using equation 5 and the value of ECF using equation 6. Both the values of TCF and ECF are multiplied with UUCP to calculate the number of revised use case points (Re-UCP). The efforts per Re-UCP is calculated by using 20 man-hours per UCP as was suggested by Karnar [14]. The equation (7) is used to convert number of Re-UCP into person hours and the same is given below

Effort = UCP * PHper UCP ----------(7)

Where PHper UCP is Person Hours per UCP

V. RESULTS AND DISCUSSION

A group of nearly 51 students selected from both UG and PG course were trained for a week long time to get hands-on the experience of using UCP, e-UCP and newly designed method Re-UCP. After the completion of training the students were divided into 11 groups wherein 5 groups with 5 members each, 5 groups with 4 members each and 2 groups with 3 members each. 14 different software projects with varying complexity were given to these 11 groups with some groups working on maximum of two projects. All the groups were subjected to use UCP, e-UCP and Re-UCP models for estimation software effort for 14 projects. The resultant data reporting format was standardized for all projects wherein all groups were asked to document use case points, using UCP, e-UCP and Re-UCP separately. The resultant data is given in the table [V] and the behaviour of this trend is represented graphically in fig. 1.

TABLE V. ESTIMATING THE TOTAL NUMBER OF USE CASE POINTS IN UCP, E-UCP AND RE-UCP

Project Name	use case points in UCP model	use case points in e-UCP model	use case points in re-UCP model
P1	128.0	134.9	138.7
P2	342.6	304.7	291.3
P3	253.4	279.8	282.8
P4	073.6	090.3	108.9
P5	096.3	099.1	122.7
P6	115.0	117.2	114.9
P7	276.4	250.4	260.3
P8	169.4	161.6	156.4
P9	067.7	065.8	063.7
P10	121.4	110.7	105.2
P11	228.2	210.9	199.8
P12	187.3	185.0	183.0
P13	208.6	210.6	192.0
P14	189.7	190.0	169.6

After analysing the bars from project 1 up to project 14 as given in fig. 1, in most of the projects the number of use cases points in case of Re-UCP is lesser when compared with UCP and e-UCP.

The use case points were calculated and then accordingly the total efforts estimation were carried out by multiplying number use case point with productivity factor of 20 man-hour as recommended by Karner [14]. The results obtained are given in table [VI] and the behaviour of all the three models is shown in fig. 2.

TABLE VI. EFFORT ESTIMATION BY USING UCP, E-UCP AND RE-UCP

Project Name	Actual Efforts	UCP Estimated Effort	e-UCP Estimated Effort	Re-UCP Estimated Effort
P1	2890	2560	2698	2774
P2	5600	6852	6094	5826
P3	5760	5068	5596	5656
P4	1925	1472	1806	2178
P5	2175	1926	1982	2454
P6	2180	2300	2344	2298
P7	4230	5528	5008	5206
P8	2870	3388	3232	3128
P9	1190	1354	1316	1274
P10	1930	2428	2214	2104
P11	3880	4564	4218	3996
P12	3350	3746	3700	3660
P13	3290	4172	4212	3840
P14	3080	3794	3800	3392

The pattern shown in fig. 2 gives an overview of estimated efforts in person hours using UCP, eUCP and Re0UCP. However, the observations from the table[vi] clearly indicate that the estimated effort using Re-UCP methods are more closed to actual efforts in comparison with estimated effort using UCP & e-UCP methods.

The deviation which is calculated by subtracting actual efforts from estimated efforts was calculated. The value of deviation can either be positive or negative and positive deviation explains that the estimated effort us greater than actual effort whereas negative deviation explained that the estimation effort is lesser than actual effort. The results of calculation for deviation of UCP, e-UCP and Re-UCP against the actual estimated effort is given in table [VII].

TABLE VII. DEVIATION OF UCP, E-UCP AND RE-UCP FROM THE ACTUAL ESTIMATED EFFORT

Project Name	Actual Efforts	UCP Deviation	e-UCP Deviation	Re-UCP Deviation
P1	2890	-330	-192	-116
P2	5600	1252	494	226
P3	5760	-692	-164	-104
P4	1925	-453	-119	253
P5	2175	-249	-193	279
P6	2180	120	164	118
P7	4230	1298	778	976
P8	2870	518	362	258
P9	1190	164	126	84
P10	1930	498	284	174
P11	3880	684	338	116
P12	3350	396	350	310
P13	3290	882	922	550
P14	3080	714	720	312

The graphical representation of the calculated deviation for UCP, e-UCP and Re-UCP against the actual effort estimation is given in fig. 3. The results represented graphically in fig. 3 shows that the project P1, P3, P4 and P5 have negative deviation whereas projects P2, P6 through P14 have positive deviation. In most of the cases across all projects from p1 to p14 there is very less deviation either positive or negative calculated using Re-UCP software effort estimation methods when compared to UCP and e-UCP method of effort estimation.

The proposed model further showed that among the methods compared (the UCP, e-UCP and Re-UCP, effort and deviation of the effort estimation methods) the performance of Re-UCP in comparison to UCP and e-UCP has improved. In project P1 and P3 the estimated effort is greater than the efforts obtained by either of the three effort estimation methods. In projects P2, P6 to P14 the estimated effort is less than the efforts acquired by either of these specified methods. Among the fourteen projects the efforts obtained in nine projects by Re-UCP is less than the efforts estimated using UCP and e-UCP methods. In projects P1, P3, P6, P8 to P14 the estimated deviation by using Re-UCP method is very less as compared to UCP and e-UCP.

VI. CONCLUSION

The effort estimation for any software development project should be carried out in the early stages of development in order to reduce the gap between the estimated effort and actual effort. To cope with the effort estimation of the projects with varying complexities Re-UCP method is proposed to cater needs of estimating effort in early stages of software development. The actors and use cases were categorized in four categories as simple, average, complex and critical. Scalability parameter was incorporated in TCF as the fourteenth parameter and Project Methodology was introduced in ECF as the ninth parameter. The performance of revised use case point model (Re-UCP) has shown improvements in estimating the efforts for software development projects with minimum trends in deviation from the actual efforts when compared with UCP and e-UCP effort estimation methods on 14 projects carried out by 11 groups after receiving proper training in the beginning.

Proposal for Future Work

The comparison of Re-UCP, UCP and e-UCP estimates, needs to be tested with data from successfully completed projects, from international and national software estimation data store organizations. The data from software development organization can be used as well to further test the performance of estimates using Re-UCP, UCP and e-UCP methods of software effort estimation. Therefore, future research in this domain needs to be carried out to strengthen the approaches available for software effort estimation which will help the developers in reducing the gap between actual efforts and estimated efforts.

REFRENCES

[1] Alves, R., Pedro V., and Nuno J. N., 2013. Improving software effort estimation with human-centric models: a comparison of UCP and iUCP accuracy. Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems. ACM.

[2] Anda B., Endre A., and Kirsten R., 2002. Improving estimation practices by applying use case models. Product Focused Software Process Improvement. Springer Berlin Heidelberg, 383-397.

[3] Apol P. S., Sholiq and Ningrum P. A., 2014. Critical Review of the effort rate value in use case point method for estimating software development effort. Journal of Theoretical and Applied Information Technology. 59(3):735-744.

[4] Ashman, R., 2004. Project estimation: a simple use-case-based model. IT professional. 6(4): 40-44.

[5] Azzeh, M., 2013. Software cost estimation based on use case points for global software development. 5th International Conference on Computer Science and Information Technology (CSIT), IEEE. 214-218.

[6] Carroll, E. R., 2005. Estimating software based on use case points. In Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, ACM. 257-265.

[7] Clemmons, R. K., 2006. Project estimation with use case points. The Journal of Defense Software Engineering. 18-22.

[8] Eberendu, A. C., 2014. Software Project Cost Estimation: Issues, Problems and Possible Solutions. International Journal of Engineering Science Invention 3(II): 38-43.

[9] Issa, A., Odeh, M., and Coward, D., 2007. Can Function Points be Mapped to Object Points. The International Arab Journal of Information Technology. 4(1): 41-49.

[10] Lynch J., and Chaos M., October 2009. The Standish Group. Boston. Available online: http://www.standishgroup.com/ newsroom/ chaos_2009.php.

[11] Jena, P. P., and Mishra, S., 2014. Survey Report on Software Cost Estimation using Use Case Point Method. International Journal of Computer Science & Engineering Technology. 5(04): 280–287.

[12] Jha, P., Jena, P. P., and Malu, R. K., 2014. Estimating Software Development Effort using UML Use Case Point (UCP) Method with a Modified set of Environmental Factors. International Journal of Computer Science and Information Technologies (IJCSIT). 5(3): 2742-2744.

[13] Jones, T. C. , Estimating software costs. McGraw-Hill, Inc. 1998.

[14] Karner, G., 1993. Metrics for objectory. Sweden: University of Linköping. Sweden. No. LiTHIDA- Ex-9344:21.

[15] Kumari, S., and Pushkar, S., 2013. Performance Analysis of the Software Cost Estimation Methods: A Review. International Journal of Advanced Research in Computer Science and Software Engineering. 3(7): 229-238.

[16] Kumari, S., and Pushkar S., 2013. Comparison and Analysis of Different Software Cost Estimation Methods. International Journal of Advanced Computer Science and Application. 4(1).

[17] Mohagheghi, Parastoo, Bente A., and Reidar C., 2005. Effort estimation of use cases for incremental large-scale software development. Proceedings. 27th International Conference on Software Engineering, ICSE 2005, IEEE.

[18] Nagar, C., 2011. Software efforts estimation using Use Case Point approach by increasing Technical Complexity and Experience Factors. International Journal of Computer Science and Engineering (IJCSE). 3(10): 3337-3345

[19] Nassif, A. B., 2010. Enhancing Use Case Points Estimation Method using Soft Computing Techniques. Journal of Global Research in Computer Science. 1(4): 12-21.

[20] Ochodek M., Nawrocki, J., and Kwarciak, K., 2011. Simplifying Effort Estimation based on Use Case Points. Sciencedirect, Elsevvier. 200-213.

[21] Periyasamy, K., and Ghode, A., 2009. Cost estimation using extended use case point (e-UCP) model. International Conference on Computational Intelligence and Software Engineering, CiSE 2009 IEEE. 1-5.

[22] Robiolo, G., and Orosco, R. 2008. Employing use cases to early estimate effort with simpler metrics. Innovations in Systems and Software Engineering. 4(1): 31-43.

[23] Ruhe, M., Jeffery, R., and Wieczorek, I., 2003. Cost estimation for web applications. Proceedings. 25th International Conference on Software Engineering, IEEE. 285-294.

[24] Schneider, G., and Winters, J., 1998 Applying Use Cases – A Practical Guide.Addison-Wesley.

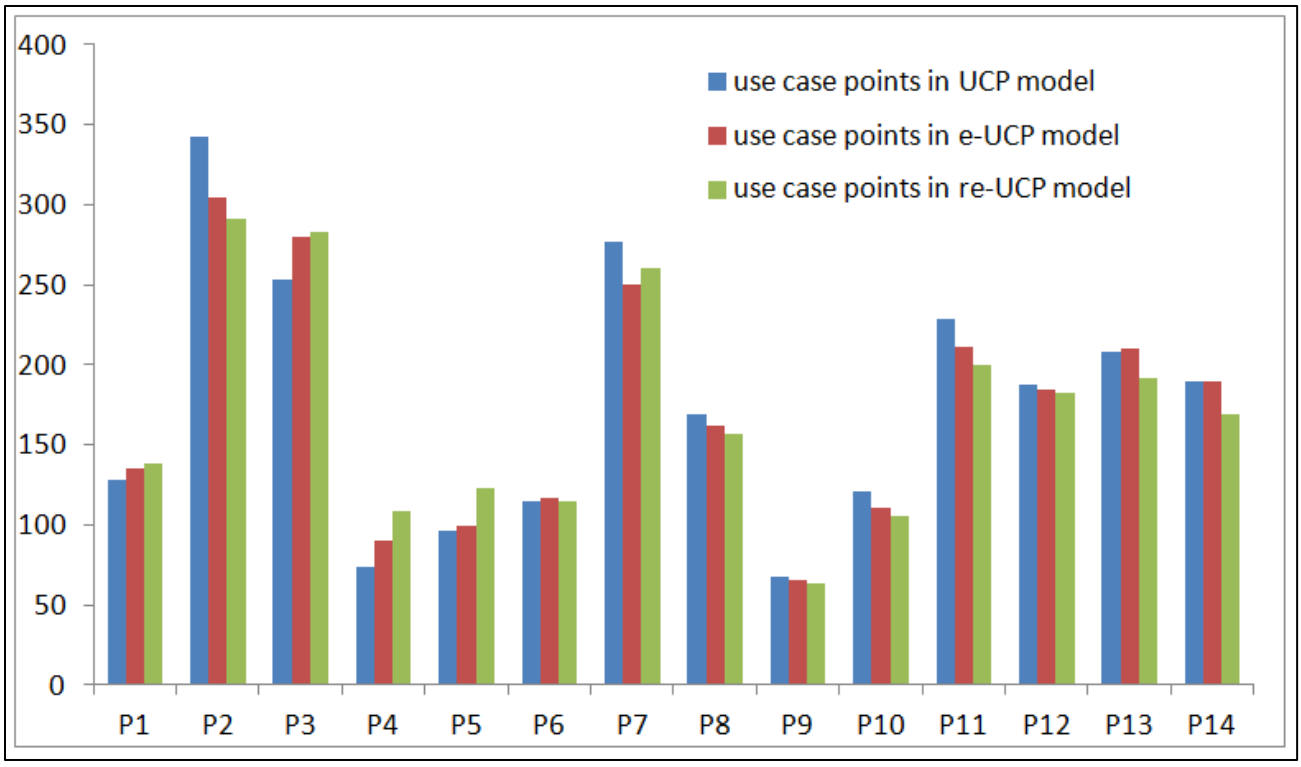

Fig. 1. Number of Use Case Points using UCP, e-UCP and Re-UCP

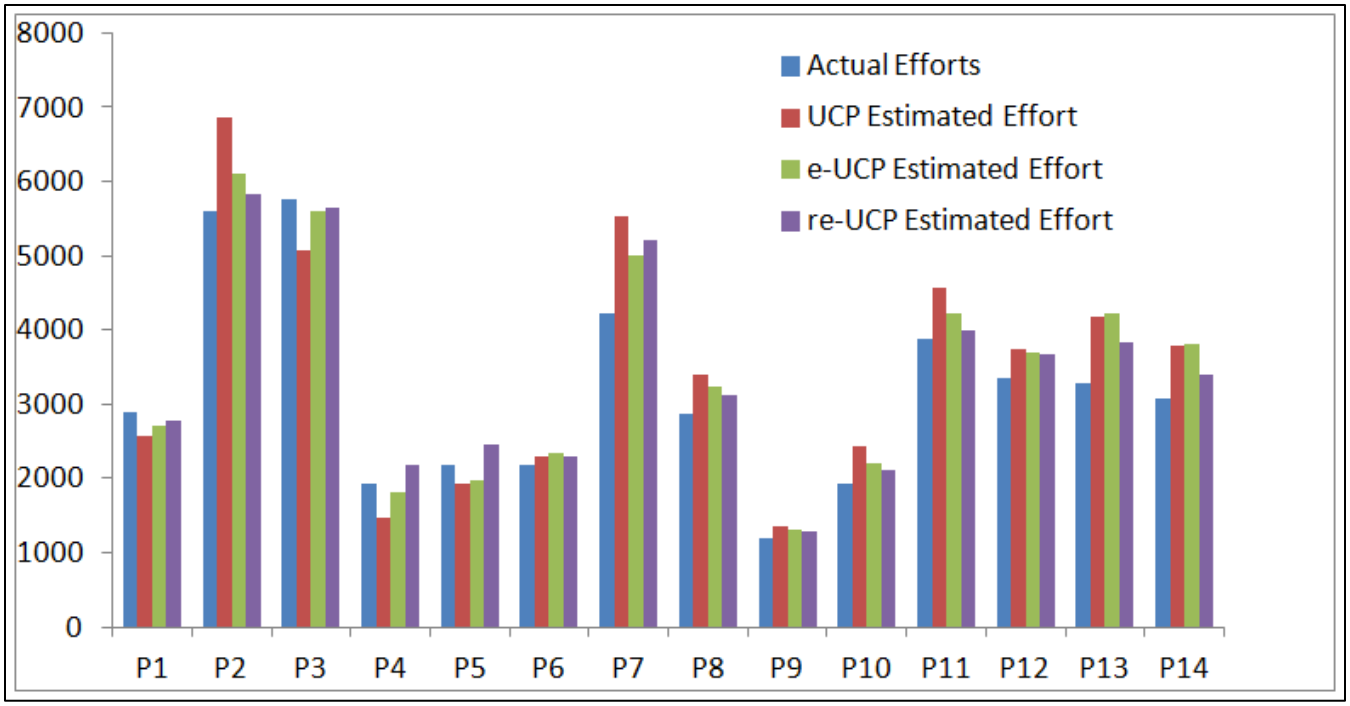

Fig. 2. Actual efforts and estimated effort using UCP, e-UCP and Re-UCP

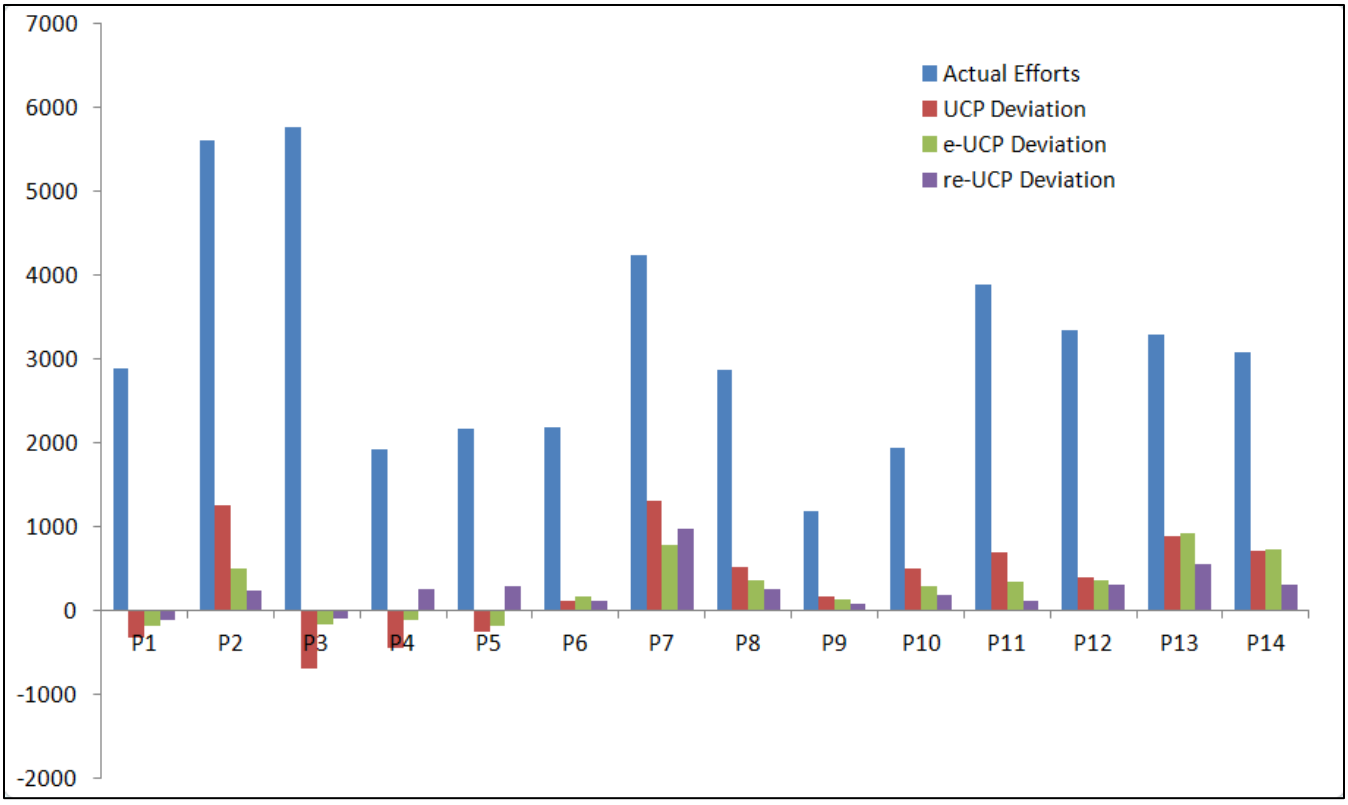

Fig. 3. Actual efforts and Deviations using UCP, e-UCP and Re-UCP