ISSN: 2321-7782 (Online)
Volume 2, Issue 7, July 2014
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

A Tool to Evaluate the Performance of UCP

Punam Bajaj[1]
Assistant Professor, Department of CSE
Chandigarh Engineering College
Mohali, Punjab – India

Dipeeka Rani Bathla[2]
Research Scholar, Department of CSE
Chandigarh Engineering College
Mohali, Punjab – India

Abstract: ***Software effort estimation is an important part of software development work and provides essential input to project feasibility analyses, bidding, budgeting and planning. A no. of techniques is used for estimation of effort amongst which widely used are COCOMO I and COCOMO II that estimates the cost in terms of efforts. But effort is linearly or non-linearly dependent on size of developing software. For size estimation Lines of Code(LOC) counts technique showed its comforts as developer do not have to solve any mathematical equation , just count the no. of lines of developed software. But uncertainties in its results and other limitations led us to different techniques. Then came Functions Point (FP) technique proving its goodness in almost every field where LOC lacked. FP had been used for long time for size estimation which is the main input for COCOMO I and COCOMO II. Nowadays software's are based on Object Oriented Paradigm and OOP languages. Developers use Unified Modeling Language (UML) notations and diagrams for estimation of each aspect of software development. Hence, this paper presents a technique that supports OOPs for estimation purposes by implementing USE CASE POINT ESTIMATION TECHNIQUE (UCP) which overcomes the drawbacks of FP which is Procedural Oriented (POP). Further evaluation of the performance of UCP in comparative to COCOMO I and COCOMO II is taken. This all will be performed with the help of a self implemented tool having all the functionalities at one location.***

Keywords: Use Case Point (UCP), FP, COCOMO, Software Cost Estimation, Cost-Benefit Analysis, Effort estimation, UML.

I. INTRODUCTION

Proper software effort estimation is the foremost activity adopted in every software development life cycle. Several features offered by OO programming concept such as Encapsulation, Inheritance, Polymorphism, Abstraction, Cohesion and Coupling play an important role to manage the development process. Currently used software development effort estimation models such as, COCOMO and Function Point Analysis, do not provide accurate project cost and effort estimates. These techniques have been proven unsatisfactory for estimating cost and effort because the LOC and FP are both used for procedural oriented paradigm. Both of them have limitations. The LOC is dependent on the programming language and the FP is absed on human decisions. Hence effort estimation during early stage of software development life cycle plays a vital role for determining whether a project is feasible in terms of cost-benefit analysis. The UCP model relies on the use Case diagram to estimate the effort of a given software product. UCP helps in providing more accurate effort estimation from the design phase of software development life cycle. UCP is measured by counting the number of use case and the number of actors, each multiplies by its complexity factors. Use cases and actors are classified into three categories including simple, average and complex. The determination of the complexity value of use cases is determined by the number of transactions per use case. Then the **unadjusted use case weights (UUCW)** are calculated by counting the number of use cases in each category, multiplying each category of use case with its weight and adding the products. The **unadjusted actor weight (UAW***)* is added to the **UUCW** to get the *unadjusted use case points* **(UUCP***)*. Next, the use case points are adjusted based on the values assigned to a number of technical factors and environmental factors. Each factor is assigned a value between 0 and 5 depending on its assumed influence on the overall project.

II. UML AND UCP

Object oriented development is the combination of object oriented analysis, object oriented design, and object oriented programming. In object oriented analysis we are preparing the model which helps us in requirements gathering. In object oriented design can be done by using the UML. In object oriented programming project construction can be done. The Unified Modeling Language (called UML) was developed to provide a common language for object oriented modeling. It was designed to be extensible in order to satisfy a wide variety of needs and was also intended to be independent of particular programming language and development methods.

UML notations and diagrams are as follow:

- Use Case Diagram
- Class Diagram.
- Sequence Diagram
- Interaction Diagram
- Collaboration Diagram
- State chart Diagram
- Activity Diagram

Every diagram and notations used in diagrams have its own significance and importance. The main concern is on Use case diagram as these are used by the developers for requirement gathering at early stages. In order to develop a method to measure effort other than that which uses function points, are due to the fact that UML is widely used and that the object oriented paradigm is extensively accepted the selected option of using use cases. They have the advantage that many people are using them, and the use case textual description is written earlier within the life cycle of a system. As it was necessary to estimate the weight of each use case in the total size calculation, transactions and entity objects were used as size notions of the use cases. To point out the advantage of utilizing the use case model instead of another alternative model as a resource for size calculation, Fig. shows, in a temporal sequence, the different artifacts that are commonly developed. The first artifact in the figure is the Client Requirements Definition. Based on this, a List of Use Cases -which defines the scope of the application, may be written. Then a textual description of each use case may be made. With this information the Analyst may define the Graphic User Interface, the Inputs and Outputs of the system.

Use case diagrams depict:

- **Actors**. An actor is a person, organization, or external system that plays a role in one or more interactions with your system. Actors are drawn as stick figures.

  For e.g., actor named employee

Figure 1: Actor representing use case diagram.

- **Use cases**. A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse.
  For e.g., employee detailed named emp detail use case is shown in diagram.

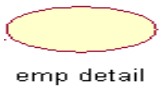

Figure 2: Use Cases representation in use case diagram.

- **Associations**. Associations between actors and use cases are indicated in use case diagrams by solid lines. An association exists whenever an actor is involved with an interaction described by a use case. Associations are modeled as lines connecting use cases and actors to one another, with an optional arrowhead on one end of the line. The arrowhead is often used to indicating the direction of the initial invocation of the relationship or to indicate the primary actor within the use case. The arrowheads are typically confused with data flow and as a result I avoid their use.

Figure3: Association representation in use case diagram

III. USE CASE POINT ESTIMATION TECHNIQUES

This technique was proposed by Gustav Karner in 1993 to estimate the projects based on OO. Some little changes were being made in the technique, but the main concept is used as it is in afterward extensions.

Use case point estimation is calculated from use case model. The main aspects of use case point estimate technique are actors, use cases, associations between actors and use cases, relationships between actors, relationships between use cases. UCP is measured by counting the number of actors and transactions included in the flow of events with some weight. A transaction is an event that occurs between an actor and the target system, the event being performed entirely or not at all. The basic formula for converting all of this into a single measure, use case points, is that we will weigh the complexity of the use cases and actors and then adjust their combined weight to reflect the influence of the nonfunctional and environmental factors.

The UCP counting process consists of the following steps:-

- Unadjusted Actors Weights (UAW)
- Unadjusted Use Case Weights (UUCW)
- Unadjusted Use Case Points (UUCP)
- Technical Complexity Factor (TCF)
- Environmental Factor (EF)
- Adjusted Use Case Points (AUCP)
- Staff hours per Use Case Point.

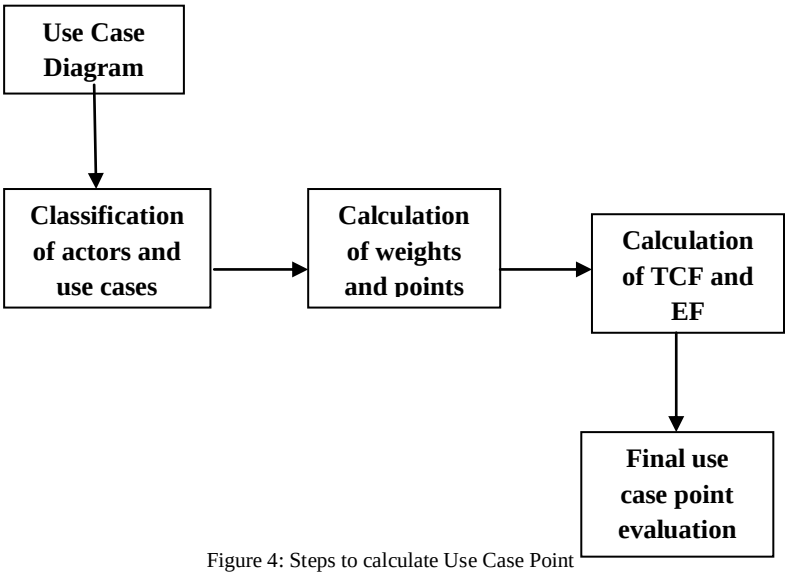

Figure 4: Steps to calculate Use Case Point

The number of use case points in a project is a function of the following:

- The number and complexity of the actors on the system.
- The number and complexity of the use cases in the system.
- Various non-functional requirements (such as portability, performance, maintainability) that are not written as use cases.
- The environment in which the project will be developed (such as the language, the team's motivation, and so on).

Advantages:

- The use case model is the front end model of the Unified Modeling Language (UML).With the emergence of the UML as the most commonly used notation to model and design object-oriented software products, the application of use cases for size and hence for effort and cost estimation seems to be a perfect fit.
- Reduces duration or estimation time as there are some use case tools available that automate the process of counting the number of use case points in system for e.g., U-EST.
- Reduces effort by using OOPs concept.
- Current trend issues for software requirement gathering are through modeling the use case diagrams.
- Use case textual description is written earlier within the life cycle of a system.
- Use case notations and diagrams are easy to understand and to interpret.
- Several environmental factors along with technical factor are present on the behalf of which estimates are assumed to be more accurate.
- Object-oriented paradigm and visual programming form the basis for use of UCP technique.
- Weighting factors are not much complicated.

IV. LITERATURE SURVEY

Work done in Use Case Point Method and Effort Estimation based on use case model are given below:-

"Software Size Measurement and Productivity Rating in a Large-Scale Software Development Department", (M. Arnold, P. Pedross, 1998): The authors proposed that the use of the use case point method is accepted to estimate the size. In this paper Arnold and other compare the object oriented methods like OOSE, OMT, OOAD, UML, ROOM, OBA, and

Syntropy. Although it has been accepted by the researches that a number of syntactic and semantic definitions existed for these methods, they also conclude that no method provides the technique for size measurement based on use cases and scenario. Researchers also described that since the language concepts for documentation are not well understood, it would be important to define the language concepts more precisely in advance.

"The Estimation of Effort Based on Use Cases", (John Smith 1999): The author proposed a framework to estimate LOC from use case diagram. The framework takes account of the idea of use case level, size and complexity, for different categories of system and does not resort to fine-grain functional decomposition.

"**Effort Estimation Tool Based on Use Case Points Method", (Shinji kusumoto, Fumikazu matukawa, Katsuro inoue, Shigeo hanabusa, Yuusuke maegawa):** To effective introduction of UCP method, the author has developed an automatic use case measurement tool, called U-EST. This work describes the idea to automatically classify the complexity of actors and use cases from use case model.

"**Cost Estimation using Extended Use Case Point (e-UCP) Model", (Kasi Periyasamy and Aditi Ghode, 2009):** This research focus on the internal details of use cases. For this a focus on the use case narratives, uses the relationships between the entities in a use case diagram, and hence closely estimates the size of the software product to be developed. The authors extended the original UCP model with additional information obtained from use case narratives also changes some parameters value.

"A Linear Use Case Based Software Cost Estimation Model", (Hasan.O. Farahneh, Ayman A. Issa, 2011): This research reports on the development of new linear use case based software cost estimation model applicable in the very early stages of software development being based on simple metric. Evaluation showed that accuracy of estimates varies between 43% and 55% of actual effort of historical test projects. These results outperformed those of well known models when applied in the same context. Further work is being carried out to improve the performance of the proposed model when considering the effect of non-functional requirements.

V. PROPOSED METHODOLOGY

Presently we are working in object oriented paradigm where a life seems to be automated, although manual workings are present but reduces to a very low level.

Now various researches have been performed in this context. Steps which will be followed in our research are as follows:-

Since the object-oriented approach has become a de facto standard for software development, it became necessary to revise the effort and cost estimation models to suit the object-oriented technology.

To reduce the efforts associated with software development.

To reduce the cost associated with estimation of software resources.

To use simple diagrams and notations of UML to make the estimation process simple such that even a novice developer can perform estimation accurate and reliable.

To make the estimation process easy while working with current trend and issues of Object oriented paradigm.

VI. CONCLUSION

In this paper, we have discussed that how use case point estimation technique is used for estimation of efforts and size and also studied that how these techniques have performed better results when applied on different data sets. Each technique is unique in its own way, which might be suitable for different applications. Hence use case point technique can be effectively

used in estimation purposes. In our future work we will be working on the factor that how to reduce the effort and cost of the project which uses this technique with the help of OOPs and UML.

References

1. Gautam Banerjee, "Use Case Points- An Estimation Approach" Aug. 2001.
2. Mel Damodaran, "Estimations using Use Case Points"
3. Encyclopedia of Software engineering, volume 2, second edition
4. Nancy Merlo – Schett, "Seminar on Software Cost Estimation", Computer Science, 2002
5. Rajiv aggarwal book. K k aggarwal.
6. Bharat Bhushan Aggarwal
7. Hareton Leung Zhang Fan, "Software cost estimation", Department of Computing, The Hong Kong Polytechnic University, pg no.2-8.
8. B.W. Boehm et al "The COCOMO 2.0 Software Cost Estimation Model", American Programmer, pp.2-17, July 1996.
9. Barry Boehm, Bradford Clark, Ellis Horowitz, Chris Westland, "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0*" Science Publishers, 1995.
10. Costar tool table.
11. COCOMO MODEL

