Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2019, 6(6):48-51

Research Article ISSN: 2394 - 658X

Use Case Point Estimation for Testing Projects

Bhupinder Paul Singh Sahni

bhupinder.sahni@gmail.com

ABSTRACT

Estimation effort for Software testing is one of the most important facets of the entire testing life cycle as it is directly proportional to the cost of the project. Estimation has a great impact on all the most important aspects of a customer’s expectation – Time, Cost and Quality. Use Case Point Test Estimation technique is one of the widely used Testing techniques in software development. The purpose of this paper is to guide through the parameters used for Use case Point measurement and how to use it by giving an example of a sample Project.

Key words: Use Case Point, Test Estimation Technique, Test Point, Software Effort Estimation, Use Cases.

INTRODUCTION

Test estimation is an important part of a project since test estimation plays a key role in test planning, scheduling project, and allocation of necessary resources. A correct estimation helps in delivering the products at the right time. If estimation is wrong, it might lead to unwanted delays in deliverables, increased cost and inappropriate results. Accuracy in estimation is the primary goal of any development, but various factors related to the environment and technical complexity inflate the size and effort of a project. The goal of a Test Manager is to use the Estimation technique that could best suit the project, considering all the known or unknown variables at the time of inception. Use Case points (UCP) Estimation technique is one such test estimation technique, which derives a reliable estimate of the size and effort an application needs by examining the actors and scenarios of a use case.

In this Paper we use a Sample Project to demonstrate how to use the Use Case Point variables to determine Test Case Execution and Test Scripting Effort, including the Requirement Analysis phase. Thereby calculating the total Testing Effort required for the Sample Project. As for any Project which needs to be estimated from scratch, we also would be taking Assumptions at different points. Let us go in detail of how Use Case Point Estimation works for Test Estimation.

UNDERSTANDING USE CASE POINT ESTIMATION

Use Case point (UCP) method is one of the commonly used size estimation methods in software development. Research work highlighted the disadvantages of using function points as it requires detailed requirements in advance and other issues in the modern object-oriented systems, which are designed using Use Cases methodology. The research work further mentioned that use case points (USP) offer significant practical advantage over the available estimation technique currently in use [1].

The use case points method is a software sizing and estimation method based on use case counts called use case points [2]. The Estimation technique involves estimating testing efforts based on the number of test cases required to validate each use case. It considers factors such as complexity, test case types, actors and environment dependencies.

The UCP method is the extension of Function Point method with the benefit of requirement analysis in the 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) [3].

USE CASE POINT METHOD

An early estimate of testing effort based on use cases can be made when there is some understanding of the project domain, system size, its complexity and after making suitable assumptions.

Let us try and understand the Use Case Point Method with a sample Project and after making some assumptions.

Sample Project: Bank ABC wants Test estimation for its website/ application where Customers fill in their personal information for applying for a Personal Loan. On a high level, there are 11 mandatory and 18 non-mandatory fields per section. The total number of Sections on the website is 6. Errors are thrown if mandatory fields are not filled, when Customers tries to Submit form.

Step 1 - Test Case Estimation: In this sample project of Bank ABC, we first would try to find out the number of test cases that would be required to fully test this application.

# of Sections: 6

# of Mandatory fields per Section: 11

# of Non-Mandatory fields per Section: 18

Total Test Cases Expected = {(Total Test Cases Expected from Mandatory Fields) + (Total Test Cases Expected from Non-Mandatory Fields)} * (# of Sections)

Assumption: 1 Test Cases will be created for every 2 Mandatory fields

TCs from Mandatory fields = (# Mandatory Fields per Module)/2

=1 {(IF Mandatory Fields per Module/2) <= 1}

Assumption: 1 Test Case will be created for every 5 Non-Mandatory fields

TCs from Non-Mandatory fields = (# Non-Mandatory Fields per Module)/5

=1 {(IF Non-Mandatory Fields per Module)/5 < = 1}

TCs from Mandatory fields → 11/2 = 5.5 ~ 6

TCs from Non-Mandatory fields → 18/5 = 3.6 ~ 4

Total Test Cases Expected = {(6+4) * 6} = 60

Here, we will make an assumption for the complexity of the Test Cases. Since the website/ application is on filling a form for website/ application, so assuming 50% of the total test cases are simple, 30% medium and 20% complex.

Complexity of Test Cases	Number of Test Cases
Simple	50% of Total Test Cases
Medium	30% of Total Test Cases
Complex	20% of Total Test Cases

Total # of Simple Test Cases = 60 * 0.5 = 30

Total # of Medium Test Cases = 60 * 0.3 = 18

Total # of Complex Test Cases = 60 * 0.2 = 12

So, from above we see that the total number of Test Cases are determined and also divided into Simple, Medium and Complex Test Cases based on an assumption.

Next step is to do the Use Case Point Measurement.

Step 2 - Use Case Point Measurement: In this section, we will calculate Use Case Point Parameters, which will help us figure out Unadjusted transaction weight (UTW), Unadjusted actor weight (UAW) and Unadjusted Use Case Points (UUCP) Calculations.

Actor Complexity: Use case points can be counted from the use case analysis of the system. The first step is to classify the actors as simple, medium or complex. A simple actor represents another system with a defined Application Programming Interface, API. An average actor is another system interacting through a protocol such as TCP/IP. A complex actor may be a person interacting through a GUI or a Web page. Complexity factor is assigned to each Actor [4]. Actors in the Sample Project can be considered as different Screens/ Systems as illustrated above.

Actor Complexity	**Actor Complexity Factor**
Simple	0.8
Average	1
Complex	1.25

Use Case Type	**Description**	**Weighting factor**
Simple	<5 Test Steps	0.75
Medium	6 -12 Test Steps	1
Complex	> 12 Test Steps	1.5

Unadjusted Transaction Weight (UTW): UTW is determined by adding products of number of test cases in each Use Case Type and its weight.
UTW = (n_{simple} * w_{simple}) + (n_{medium} * w_{medium}) + ($n_{complex}$ * $w_{complex}$)
In Sample Project, UTW = (30 * 0.75) + (18 * 1) + (12 * 1.5) = 58.5
Assumption: Number of Simple Actors = 1
Number of Medium Actors = 1
Number of Complex Actors =2
Unadjusted Actor Weight (UAW): UAW is determined by adding products of counts of actors of each type and their weight [4].
UAW = (n_{simple} * w_{simple}) + ($n_{average}$ * $w_{average}$) + ($n_{complex}$ * $w_{complex}$)
In Sample Project, UAW = (1 * 0.8) + (1 * 1) + (2 * 1.25) = 4.3
Unadjusted Use Case Points (UUCP): UUCP is determined by adding Unadjusted Transaction Weight (UTW) and Unadjusted Actor Weight (UAW) [5].
UUCP = UTW + UAW
In Sample Project, UUCP = 58.5 + 4.3 = 62.8
Similarly, if there are different requirements within the same project, then UUCP is calculated individually and then added together to get Total UUCP. For the purpose of simplicity, we are assuming just 1 requirement as mentioned in Sample Project above.

Step 3 – Test Case Execution Estimation: Test cases differ widely in terms of complexity and the activities necessary to execute it. Therefore, the test cases need to be normalized to one common measure using weighting factors. Here, Test Case normalization will be done as per the project domain weightage and based on the Types of Testing required, we will calculate Time required for Test Case Execution in this section.
Unadjusted Test Point Weightage (TPW): TPW is the product of Unadjusted Use Case Points (UUCP) and Domain Weightage. The Domain Weightage value that multiplies the UUCP represents an adjustment factor that enables us to derive a size measure that more accurately represents the system’s complexity [6].
TPW = UUCP * Domain Weightage
Assumption: Domain Weightage for Financial Projects = 1.5
Domain Weightage for Non-Financial Projects = 1
TPW = 62.8 * 1.5 = 94.2
Test Points (TP): TP is the sum of Unadjusted Test Point Weightage (TPW). With TP, Size of Testing Project is estimated.
TP = Sum (TPW)
In Sample Project, since there is only one requirement, there is only one TPW.
So, TP for Sample Project is 94.2.
Adjusted Test Point (ATP): Test Case execution effort has been determined using Adjusted Test Points. This value can be used to further determine other values e.g. Scripting effort or Person days.
Test Case normalization will be done depending upon the test point weightage considering testing types.
ATP = TP * Weightage Factor
Weightage Factor is determined based on the Testing type required for the Project.

Testing Type	**Weightage Factor**
Functional Test	0.35
System Test	0.3
Acceptance Test	0.2
Virus-free Test	0.15
Total Weight	**1**

n sample Project, considering only Functional Test and System Test would be required, so ATP can be tested accordingly as below:
ATP = 94.2 * (0.35 + 0.3) = 61.23
If a Project requires Acceptance Test, then Weightage factor of 0.2 can also be added.
As discussed earlier, ATP is the Test Execution Effort.
So, Test execution Effort for the Sample Project is 61.23 hrs

Step 4 – Test Plan Scripting Estimation: After we know Time required for Test Case Execution from the previous step, we will now figure out Time required to do Test Case scripting effort (incl. Requirement Understanding) and Total Testing Effort for the Project.
Test Case scripting effort inclusive of Requirement Analysis has been determined from Test Case Execution. Data analysis shows that test scripting effort (incl. Requirement analysis) is almost 1.25 times the test execution effort. This includes Requirement Understanding Efforts as well.
Test Scripting Effort (incl. Requirement Analysis) = 1.25 * Test Case Execution Effort
Test Scripting Effort (incl. Requirement Analysis) = 1.25 * 61.23 = 76.54 hrs
So Total Testing Effort for Sample Project for Bank ABC is:
Test Scripting Effort (incl. Requirement Analysis) + Test execution Effort → 61.23 + 76.54 = **137.77 hrs**
Total Testing Effort can further be divided into Man days and Man Months, if required.

CONCLUSION

Test estimation is a tedious task for every software industry. A reliable effort estimate is crucial for planning and management of software projects. This paper has highlighted the systematic approach to calculate Testing Efforts using Use Case Point Estimation technique. This Technique depends a lot on the correct Assumptions being made but is definitely a good technique when a new Application/ website is being developed, and even for other Projects. Overall, Use Case Points testing technique provides a structured approach to estimating testing effort based on the functional requirements of the software, complexity of the system, test case types, actors and environment dependencies, making it particularly useful in projects where use cases play a significant role in defining system behavior and interactions.

REFERENCES

[1]. M. Damodaran, A. Washington, “Estimation using use case points.”. Computer Science Program. Texas–Victoria: University of Houston. Sd., 2002.
[2]. M. Kirmani, A. Wahid, “Use Case Point Method of Software Effort Estimation: A Review”, International Journal of Computer Applications, vol. 116, no. 15, pp. 43–47, April 2015.
[3]. P. Jena, S. Mishra, “Survey Report on Software Cost Estimation using Use Case Point Method”, International Journal of Computer Science & Engineering Technology, vol. 5, no. 4, pp. 280–287, April 2014.
[4]. D. Kashyap, D. Shukla, A. Misra, “Refining the Use Case Classification for Use Case Point Method for Software Effort Estimation”, Proc. of Int. Conf. on Recent Trends in Information, Telecommunication and Computing, ITC, pp. 183–191, 2014, doi: 02.ITC.2014.5.501
[5]. C. Nagar, “Efforts Estimation by Combining the Use Case Point and COCOMO”, International Journal of Computer Applications, vol. 52, no. 7, pp. 1–5, Aug 2012.
[6]. J. Popovic, D. Bojic, N. Korolija, “Analysis of task effort estimation accuracy based on use case point size”, IET Journals, vol. 9, issue 6, pp. 166–173, April 2014. doi: 10.1049/iet-sen.2014.0254