Estimates

Software Testing Estimation (STE) can be defined as the management of activity that calculates a rough estimation of how long a task or work or test will take in completing.

Software testing estimation is a management activity to calculate and approximate time, resources and expenses needed to complete test activities in a specified environment.

Estimating effort for the test is one of the major and important tasks in Test Management.

Factors needed in Estimating Effort for Test:

The complexity of your application needs to be tested.
Proper availability of requirements.
Proper requirement's availability of documents other artifacts for testing.
Proper external application linkages.
Previous testing experience.
Specific & domain software knowledge.
Estimate the risks that may occur in application deployment.
Various methods were implemented for conducting the test.
The method and technology needed for developing the application.
Tools needed for testing.

Rules for estimate/Consider while estimating /Approach to estimate:

For the estimation of any testing project, we highly recommend following these rules:

- Base the estimation of the software requirements
- Make use of the data, experience and knowledge from previous projects (it helps get results cheaper and faster, up to 25%)
- Add reasonable buffer time
- Consider possible internal and external risks (downtime, skill improvements, lack of resources, etc.) and the complexity of the application
- Consider resources availability and team knowledge and skills
- Update the estimation if there are changes in the project
- Verify the estimation and ask an outside expert to review it.

How to Estimate/Technique:

What to Estimate?

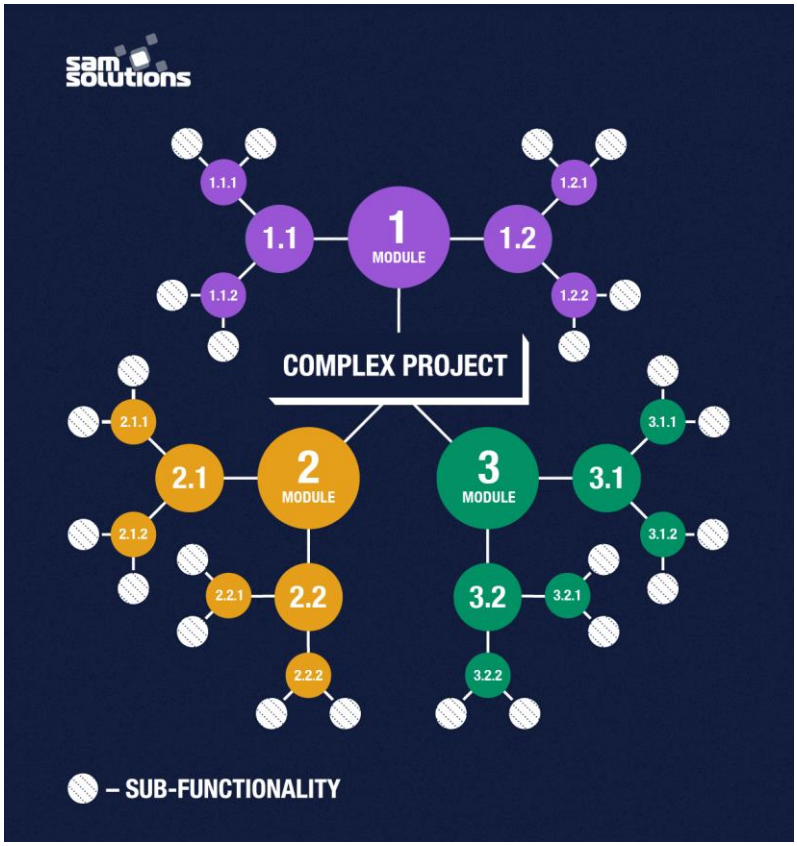
- **Resources:** Resources are required to **carry out** any project tasks. They can be people, equipment, facilities, funding, or anything else capable of definition required for the completion of a project activity.
- **Times :** Time is the most valuable resource in a project. Every project has a deadline to delivery.
- **Human Skills :** Human skills mean the **knowledge** and the **experience** of the Team members. They affect to your estimation. For example, a team, whose members have low testing skills, will take more time to finish the project than the one which has high testing skills.
- **Cost:** Cost is the project **budget**. Generally speaking, it means **how much money** it takes to finish the project.

Estimation Techniques for testing software:-

- Work Breakdown Structure
- 3-Point product Estimation Test
- Wideband Delphi method
- Function point and testing point breakdown
- Use-case Methodologies
- Distribution in percentage
- Method of ad-hoc
- Agile model

- **Work Breakdown Structure**

The essence of this technique is to divide a complex test project into small components to obtain the following hierarchy: the project is broken down into sub-modules; each sub-module, in turn, is divided into functionalities, which are split into sub-functionalities. As a result, we get a very detailed and transparent structure (the sample is given below).



Since small tasks are much easier to estimate, they are allocated among the team members. Every person assumes responsibility for a given task and provides the result in a certain period of time. The project can, therefore, be completely checked step by step, without missing any detail. This approach eliminates situations in which some testers may perform the same task simultaneously, while other tasks may remain unaccomplished.

The WBS is the most powerful and effective method among QA estimation techniques. It enables a project manager to create a clear schedule and to plan a reliable budget. It also cultivates team cooperation and personal commitment.

Following Steps process to arrive at an estimate-

Step1) Divide the whole project task into subtasks Example-

Task	Sub task
Analyze software requirement specification	Investigate the soft requirement specs
	Discuss with team for analys the requirement
Create the Test Specification	Design test scenarios
	Create test cases
	Review and revise test cases
Execute the test cases	Build up the test environment
	Execute the test cases
	Review test execution results
Report the defects	Log bugs and retest bugs
	Create the defect reports
	Report the defects

Step 2) Allocate each task to team member

Task	Members
Analyze software requirement specification	All the members
Create the test specification	Tester/Test Analyst
Build up the test environment	Test Administrator,DEV,IT DESK
Execute the test cases	Tester
Report defects	Tester
TEST PLAN	SENIOR/TL
STRATEGY	PM

Effort and Estimation For Tasks/requirement/user stories can be calculated by-

1. **Functional Point Method**
2. **Three Point Estimation**

<u>Functional Point Method-</u>

This technique indicates software functionality from the user's perspective. FPA is based on specification documents, such as SRS document or Design.

Again, as with WBS, the project is split into modules. Each module — depending on its complexity — is assigned a *functional point (FP).*

Simple tasks get lower points, difficult tasks get higher points.

Total effort is calculated by the formula:

Total Effort = Total FP x Estimate Defined per FP

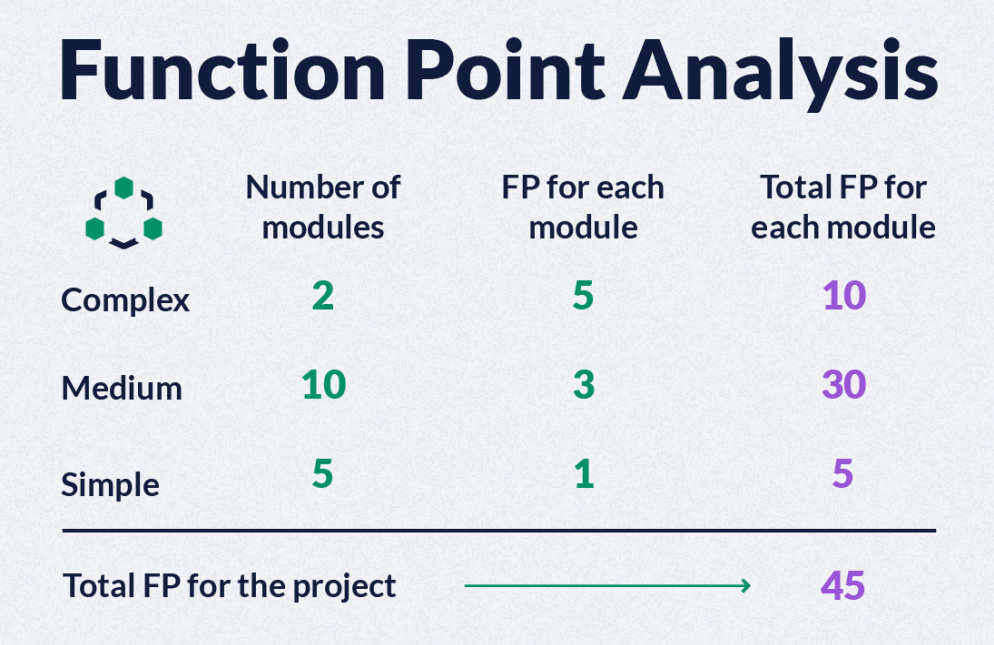
Estimate defined per FP is a **given value** that a test manager defines on the basis of the team experience.

Example:

Let's consider the total effort with respect to cost and take the estimate defined per FP as equal to **$100/points**.

The whole project is divided into three groups of modules:

- Complex modules (FP is 5) — 2 pieces
- Medium modules (FP is 3) — 10 pieces
- Simple modules (FP is 1) — 5 pieces



Total effort = 45 x 100 = $4,500

This means that to complete the project, you need $4,500.

FPA is not a really accurate method, as it involves risks and may give false results.

Nevertheless, it is one of the more popular estimation techniques in software development and can be used in combination with other tools.

Three-Point Estimation

This is a statistical method, but it also breaks down the task into subtasks (in this it is similar to WBS).

Then, three possible scenarios should be estimated for each sub-task.

1. **The best case:** assuming that you have a talented team and all of the necessary resources, and assuming that no problem occurs and everything goes right, you can complete the task.

   Example, in *100 man-hours (B).* This is an optimistic scenario.

2. **The most likely case:** assuming that you have a good team, enough resources, and almost everything goes right, although some problems may occur, you can complete the task in *150 man-hours (M).*

   This is a normal scenario.

3. **The worst case:** assuming that your team is not experienced, everything goes wrong and you have to solve numerous problems, you can complete the task in *200 man-hours (W).*

   This is a pessimistic scenario.

Thus, you have three values:

B = 100

M = 150

W = 200

Now, you can calculate the average value for the test *estimation (E)* using the following formula:

E = (B + 4M + W) / 6

E = (100 + 4 x 150 + 200) / 6 = 150 man-hours

As the average value may fluctuate a little bit, to be more accurate, you need to calculate *standard deviation (SD)* — . The formula is as follows:

SD = (W – B) / 6

SD = (200 – 100) / 6 = 16.7 man-hours

You can present the final estimate as this: the team needs 150 +/- 16.7 person-hours to accomplish the sub-task.

Three-Point Estimation is one of the most effective methods for software testing

This method is to find out the best and the worst working conditions for your team.

Agile Estimation

Agile estimation is about evaluating the effort required to complete each work item listed in the prioritized backlog, which, in turn, helps improve sprint planning.

Agile estimation is the process for estimating the effort required to complete a prioritized task in the product backlog. This effort is usually measured with respect to the time it will take to complete that task, which, in turn, leads to accurate sprint planning.

Agile teams also make estimations with reference to story points. A story point is used in Agile Development projects to estimate the difficulty of implementing a given user story.

The steps involved in the estimation method with story points are as follows:

- Identify user stories
- Discuss the requirements of the user story. It is the responsibility of the Product Owner or business analyst to answer the questions and explain what precisely the referred story entails

•Create a matrix for estimation: The estimation matrix is a numeric scale that is used to evaluate the selected pieces of work. This scale can be the Fibonacci sequence (…5, 8, 13, 21, 34 …) or the straightforward linear scale (… 3, 4, 5, 6, 7 …).

- Choose an Agile estimation technique

•Conduct sprint planning

•Validate that the estimates are internally consistent and align with the stories as the process goes along

Here is a list of some **agile estimation techniques:**

- Planning Poker
- Affinity Mapping
- Bucket System
- Big/Uncertain/Small
- T-Shirt Sizes
- Dot Voting
- Ordering Protocol

Test estimation best practices

- **Add some buffer time:** Many unpredictable things may happen to your project, such as a talented team member quits his job suddenly, the testing takes more time than estimated to complete… etc. That why you need include some buffer in your estimation.
- **Account Resource planning in estimation:** What should you do if some members in your team take long leaves? It may delay the project.
  Here you have to consider the leaves for your team member, generally long leaves.
- **Use the past experience as reference:** Experiences from past projects is good while preparing the time estimates. Because some project may be some similarity, you can reuse the past estimation.
  For example, if you use to do a project like testing a website, you can learn from that experience, try to avoid all the difficulties or issues that were faced in past projects.
- **Stick to your estimation:** Estimation is just estimate because it may go **wrong**.In early stages of the project, you should frequently **re-check the test estimations and make modification** if needed. We should not extend the estimation after we fix it, unless there are major changes in requirement or any unavoidable situation.