

### 3.8.2 Earned Value Analysis

Q. 3.8.3 What is Earned Value Analysis in Project tracking concept? (Ref. sec. 3.8.2) (5 Marks)

- Earned Value Analysis (EVA) is the key technique used in Project Management.
- The purpose of Earned Value Analysis is to understand how the project is progressing.
- EVA used to estimate the progress of a project based on earnings or money and schedules are calculated on the basis of EVA.
- In simple words Earned Value is a measure of 'Progress' to evaluate 'Percentage of Completion'.

#### 3.8.2(A) Features of EVA

- Earned Value Analysis's objective is to measure project performance in terms of scope, cost and time.
- EVA is used to evaluate project health and project performance.
- Earned Value Analysis is also used for monitoring the progress of a software project / team which involves tasks allocated to the Project Schedule.
- Total time required to complete the project is calculated and each task is given an Earned Value based on its estimated (%) out of the total.

#### 3.8.2(B) Need for EVA

- EVA provides the measures of project process of different tasks associated with project, so it is a single way of measuring each and every point in the project.
- It also provides a signal for corrective action in the project. The types of signals can be the following :

##### 1. Time to recover

If the project is not going as per plan and it is found that there is some delay in the project. Then at this time, situation is needed to be taken care by finding out the reasons that are causing delay and by taking the required corrective actions.

##### 2. Request for additional funds/resources

While there is time to recover, the need for extra resources or money can be calculated with an early warning.

Software Engineering (MU - Sem. 6 - Comp)

#### Key Elements of EVA

##### 1. Planned Value (PV)

The allocated cost for the project which is approved. It was known as Budgeted Cost of Work Scheduled (BCWS).

##### 2. Earned Value (EV)

The budgeted value of the completed work packages. It used to be known as Budgeted Cost of Work Performance at a specified point (BCWP).

##### 3. Actual Cost (AC)

The actual cost involved during the execution of project work. It was previously called Actual Cost of Work Performed (ACWP).

#### Tools and techniques

There are several software packages available which are listed are as follows :

1. Schedule maker
2. Planisware OPX2
3. DIAL TOOL

#### ✓ Earned Value Analysis – Example

- Consider you are handling a software development project. The project deadline of the project is eight months with costing of \$10,000 per month.
- After two months, you get understood that the project work is thirty percent completed with costing of \$40,000. Now there is need to determine whether the status of project will be on-time and on-budget after two months.

**Step 1 :** Calculate the Planned Value (PV) and Earned Value (EV)

From the scenario,

1. Budget at Completion (BAC) =  $\$10,000 \times 8 = \$80,000$
2. Actual Cost (AC) =  $\$40,000$
3. Planned Completion =  $2/8 = 25\%$
4. Actual Completion =  $30\%$

Therefore,

$$\text{Planned Value} = \text{Planned Completion (\%)} \times \text{BAC} = 25\% \times \$80,000 = \$20,000$$

$$\text{Earned Value} = \text{Actual Completion (\%)} \times \text{BAC} = 30\% \times \$80,000 = \$24,000$$

**Step 2 :** Compute the Cost Performance Index (CPI) and Schedule Performance Index (SPI)

$$\text{Cost Performance Index (CPI)} = \text{EV} / \text{AC} = \$24,000 / \$40,000 = 0.6$$

$$\text{Schedule Performance Index (SPI)} = \text{EV} / \text{PV} = \$24,000 / \$20,000 = 1.2$$

## Module 5: Risk Analysis

- A risk table provides a project manager with a simple technique for risk projection
- It consists of five columns :
  - o **Risk Summary** : Short description of the risk.
  - o **Risk Category** : One of seven risk categories.
  - o **Probability** : Estimation of risk occurrence based on group input.
  - o **Impact** : (1) catastrophic (2) critical (3) marginal (4) negligible.
  - o **RMMM** : Pointer to a paragraph in the Risk Mitigation, Monitoring, and Management Plan.

Risk Summary	Risk Category	Probability	Impact (1-4)	RMMM

#### ✓ Developing a Risk Table

- List all risks in the first column (by the help of the risk item checklists). Mark the category of each risk.
- Estimate the probability of each risk occurring.
- Assess the impact of each risk based on an averaging of the four risk components to determine an overall impact value.
- Sort the rows by probability and impact in descending order.
- Draw a horizontal cutoff line in the table that indicates the risks that will be given further attention.

The overall risk exposure formula is  $RE = P \times C$  and for how long the impact will be felt.

**P** = the probability of occurrence for a risk

**C** = the cost to the project should the risk actually occurs

#### Example

**P** = 80% probability that 18 of 60 software components will have to be developed

**C** = Total cost of developing 18 components is \$25,000

$$RE = 0.80 \times \$25,000 = \$20,000$$

## 5.5 RMMM

→ (MU - May 15)

Q. 5.5.1 Write note on RMMM. (Ref. sec. 5.5)

May 15, 5 Marks

Q. 5.5.2 Explain the steps involved in setting up or generating RMMM plan. (Ref. sec. 5.5)

(5 Marks)

- The main aim of all of the risk analysis related activities is to guide the project team in forming a strategy for dealing with risk.
- In an effective strategy there are three important issues : risk avoidance (mitigation), risk monitoring, and risk management and contingency planning.
- If proactive approach to risk is accepted by a software team, avoidance is considered as always the best strategy.
- This is accomplished by setting a plan for risk mitigation.
- For example, consider that high staff turnover is marked as a project risk  $r_1$ . As per previous history and management instinct, the likelihood  $l_1$  of high turnover is guessed to be 0.70 (70 percent, rather high) and the impact  $x_1$  is projected as critical.
- It indicates that there will be critical impact of high turnover on project cost and schedule.
- To mitigate this risk, a strategy can be developed to decrease turnover.

SQA

## 5.8 Software Quality Assurance

Q. 5.8.1 Explain the concept of Software Quality Assurance in details. (Ref. sec. 5.8)

(5 Marks)

- Software Quality Assurance (SQA) is a group of activities to guarantee that the quality in software engineering processes is maintained.
- It make sure that developed software project would meet and complies with the predefined or standardized quality specifications.
- SQA is an ongoing activity in the Software Development Life Cycle (SDLC) that consistently checks the developed software to guarantee that it should meet the desired quality standards.



- SQA practices are adapted in most types of software development industries.
- SQA incorporates and implements software testing methodologies to test the software. Rather than every time checking for quality after completion of each project, SQA processes test the quality in each phase of development, throughout the software development process.
- With SQA, the software development activities moves into the next level only if the present/prior level complies with the required quality standards.
- SQA generally works on one or more engineering standards that assist in building software quality guiding principle and execution strategies.
- Once the processes have been defined and implemented, Quality Assurance has the following tasks :
  - (a) To identify the weaknesses in the processes,
  - (b) Correct those weaknesses to constantly improve the process.

It includes the following activities

- (1) Process definition and accomplishment
- (2) Auditing
- (3) Guidance

Processes could be

- 1) Software Development method
- 2) Project Management
- 3) Configuration Management
- 4) Requirements Development/Management
- 5) Estimation
- 6) Software Design
- 7) Testing, etc.

### 5.15 Formal Technical Review (FTR)

- Q. 5.15.1 What is FTR in SQA? What are its objectives? Explain steps in FTR. (Ref. sec. 5.15) → (MU - Dec. 15, May 16, May 17, May 18)
- Q. 5.15.2 Explain FTR. (Ref. sec. 5.15) Dec. 15, May 17, 10 Marks  
May 16, May 18, 3 Marks

- A formal technical review is a process performed to give assurance of software quality.
- This testing activity is done by software engineers and other persons.
- The objectives of the Formal Technical Review are :
  - (1) To find out errors in function, logic, or coding in the software.
  - (2) To check that the software fulfils the requirements for which it is built.
  - (3) To give assurance that software has been designed as per customer requirements.
  - (4) To create software developed in uniform order.
  - (5) To create more manageable project.
- The Formal Technical Review is used while providing training and it is also used as an example for junior developers to study different ways for software analysis, design, and development.
- The Formal Technical Reviews is a class of reviews that contains walkthroughs, inspections, round-robin reviews and other small technical assessments of software.
- Every Formal Technical Review is performed as a meeting.
- If Formal Technical Reviews is planned, controlled, and attended in correct way then only it becomes successful.

## Module 6: Software Testing

### 6.1 Software Testing

- Software testing is a procedure to verify whether the actual results are same as of expected results.
- Software testing is performed to provide assurance that the software system does not contain any defects.
- Defects means errors which are detected by tester when actual result of our software module is not identical as expected result.
- Software testing helps us to find out whether all user requirements are fulfilled by our software or not.
- Software quality depends on; at what extend software fulfills user requirements and number of defects occurred in software.
- Software testing is used to give assurance that we deliver quality product to customer.
- To perform software testing we create test cases and test data. Test Case is a collection of actions which applied on our software product to check specific feature or functionality of it.
- Collection of test cases is called as test suit.
- Test data is the input provided to modules which are present in our software product. Test data represents the data which effects execution of the particular module. Sometime test data is used for positive testing. That means it is used to check that a provided set of input for given function generate expected result or not. Sometime test data is used for negative testing. That means test data is used to test the capability of our software modules to handle unexpected input.

#### 6.1.1 Advantages of Software Testing

(5 Marks)

Q. 6.1.1 What are the advantages of Software Testing ? (Ref. sec. 6.1.1)

There are several advantages of software testing :

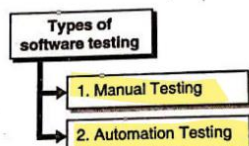
1. Testing reduces the possibility of software failure by removing errors which leads to software failure.
2. Testing process removes maximum possible errors from software and helps to deliver qualitative software to customer.
3. Testing ensures correctness and completeness of software along with quality.
4. Testing allows us to verify and validate the software and ensures that software will satisfy all of the user's requirements.
5. Testing enables the software to produce expected outcome by removing errors which prevent system from producing expected outcome.

#### 6.1.2 Types of Software Testing

(10 Marks)

Q. 6.1.2 Write note on Manual and Automation Testing. (Ref. sec. 6.1.2)

Software Testing is divided in to two categories :



## → 1. Manual Testing

- Manual Testing is one of the type of Software Testing in which Tester treat himself as end user and use every module of application from login module to log out module to check application behavior according to requirement provided by user.
- In manual testing, tester manually runs all the test cases without help of any automation testing tools.
- Manual Testing is the very basic type of all testing types and supports to search the bug (error) in the software system.
- First we do manual testing for any new application before going for automation testing. We perform feasibility study of automation testing in manual testing i.e. do the analysis of benefits of performing automation testing for our project to take the decision that whether to do automation testing or not.
- For performing manual testing, we do not need knowledge of any testing tool.

List of manual testing is as follow :

- Unit testing
- Integration testing
- System testing
- Acceptance testing

## 2. Automation Testing

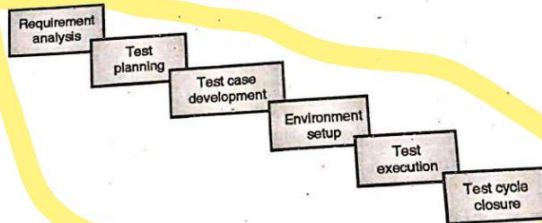
- Automation Testing is a process in which we use automation tools to run our test cases to find out bugs from our software product.
- The automation software is able to enter test data into the software on which we perform testing and it also matches the expected and actual results to create test reports.
- Automation testing requires considerable amount of money and resources such as employees, testing tools etc.
- After change in requirement or occurrence of error, we need to make changes in code and we require executing same test suite again and again.
- We can record test suite by using test automation tools and re-play it when needed.
- Automation testing does not require human interaction to perform testing like manual testing.
- Aim of Automation testing is to decrease number of test cases to be executed manually but not replace Manual Testing completely.
- To perform automation testing we require generation of test cases, test data and test script.
- Automation testing is performed for project if requirements of projects are stable at some extent i.e. requirements are not frequently changing.
- List of some automation tools
  - Selenium
  - Mentis
  - Quality Test Professional (QTP)
  - Buxila
  - HP ALM (Application Lifecycle Management)



### 6.1.6 Software Testing Process

Q. 6.1.6 Explain the software testing process in detail. (10 Marks)  
(Ref. sec. 6.1.6)

- Software testing process is a sequence of various activities which helps to certify software system.
- Software testing process activities are : Requirement analysis, Test planning, Test case development, environment set up, Test execution, Test cycle closures.
- Each of the activity of software testing process has defined entry and exit criteria.



### Unit Testing

- Unit Testing is a level of software testing where individual units / components of a software are tested.
- The purpose is to validate that each unit of the software performs as designed.
- A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.
- In procedural programming, a unit may be an individual program, function, procedure, etc.
- In object-oriented programming, the smallest unit is a method, which may belong to a base / super class, abstract class or derived/ child class. (Some treat a module of an application as a unit.)

→ (MU - Dec. 15, Dec. 16)

Q. 6.4.1 Explain integration testing. (Ref. sec. 6.4)

Dec. 15, Dec. 16, 5 Marks

- Integration testing is a method of software testing in which all units of software under test are integrated and tested as a single group.
- It is always done after unit testing is applied.
- It mainly focuses on testing data communication among all units of system.

Scanned by CamScanner

Software Engineering (MU - Sem. 6 - Comp) 6-15

Software Testing and Maintenance

and Maintenance

and we need so developer module A. This

Module B will return

Marks)

It makes use of methodologies as Big bang approach and incremental approach.

Incremental approach can be either top-down, bottom-up or combination of both.

The main objective of integration testing is to determine faults in communication between, integrated units.

Fig. 6.4.1

Following are reasons for performing Integration testing :

- A Module is developed by software developer whose level of understanding and programming logic may not be same as of other software developers from project team.
- So, while performing Integration Testing, we required to check whether software module works efficiently with other modules or not.
- While developing single module, there is risk of changes in requirements by the clients.
- We may not perform unit testing for this new requirements and so performing integration testing of software is needed.

### 6.5.3 Alpha and Beta testing

Q. 6.5.2 Write note on Alpha and Beta Testing. (Ref. sec. 6.5.3)

(10 Marks)

#### Alpha testing

- Alpha Testing is software testing which is done to find out bugs before deploying the software application to end user.
- Alpha testing is a type of acceptance testing.
- This testing is called as an alpha testing since it is performed when development phase of software application is near to Beta Testing.
- The objective of alpha testing is to refine the software product by finding (and fixing) the bugs that were not discovered through previous tests.
- Alpha testing is typically performed by in-house software engineers or QA staff.
- It is the final testing stage before the software is released into the real world.
- Alpha testing is performed in two phases :
  - (i) In first phase software is tested by development team members. They perform debugging of software to catch bugs quickly.
  - (ii) In second phase software is tested by software quality analyst team for additional testing in actual user's environment setup.

#### (2) Beta testing

- Beta testing is testing which is performed at the location of customer. In this testing actual as well as intended users will test the software to determine whether the software is satisfying their needs and expectations.
- Beta testing allows users to test software before it is released to public.
- Beta testing minimizes the product failure risks and delivers qualitative product through customer validation. It gives direct feedback from the user.
- It ensures reliability, security, robustness etc. from user's perspective.
- Types of Beta testing are : Traditional, public, technical, focused and past release.
- Beta testing is also called as User Acceptance Testing, Customer acceptance Testing, Customer Validation testing and Pre-Release testing.
- Beta testing gives assurance that we deliver quality software to our customers by testing the product under various situations in real world environment that can't be created in a lab setting.

Fig. 6.6.1 : Software testing methods

#### → 1. Black Box Testing

In this type of testing, we check working of our software project by running it. We do not need to check code of our software in this testing type. System testing is included in black box testing

#### → 2. White Box Testing

In white box testing we check internal working of our code. So in this type of testing tester need deep knowledge of programming language which is used in our software product.



**1.11.1 Agility Principles**

→ (MU - May 15, Dec. 17)

**Q. 1.11.6** Explain principles of Agile Methodology. (Ref. sec. 1.11.1)

May 15, 4 Marks

**Q. 1.11.7** Describe and discuss characteristics of Agile Process Model.  
(Ref. sec. 1.11.1)

Dec. 17, 10 Marks

The Manifesto for Agile Software Development is based on twelve principles :

1. Customer satisfaction by early and continuous delivery of valuable software.
2. Welcome changing requirements, even in late development.
3. Working software is delivered frequently (weeks rather than months).
4. Close, daily cooperation between business people and developers.
5. Projects are built around motivated individuals, who should be trusted.
6. Face-to-face conversation is the best form of communication (co-location).
7. Working software is the primary measure of progress.
8. Sustainable development, able to maintain a constant pace.
9. Continuous attention to technical excellence and good design.
10. Simplicity is essential.
11. Best architectures, requirements, and designs emerge from self-organizing teams.
12. Regularly, the team reflects on how to become more effective, and adjusts accordingly.