**Testing Concepts**

Lesson 05: Tool support for testing

Tool support for testing:

Test tools can be used for activities that support testing:

* Directly used in testing such as test execution
* Help in managing the testing process
* Used for exploration
* Aids in testing such as spreadsheet

Test Management Tools

* Provide interfaces for executing tests
* Track defects
* Manage requirements
* Support for quantitative analysis
* Reporting of the test objects
* Tracing the test objects to requirements

Requirements Management Tools

* Store requirement statements
* Store the attributes for the requirements
* Provide unique identifiers
* Support tracing the requirements to individual tests
* Help to identify inconsistent or missing requirements

Incident Management Tools (Defect Tracking Tools)

* Store and manage incident reports
* Help in managing the life cycle of incidents, optionally provide support for statistical analysis

Configuration Management Tools

* Not strictly test tools but are necessary
* storing information about versions and builds of the software and testware
* traceability between software and testware
* release management, baselining, and access control.

Review Tools

* Assist with review processes, checklists, review guidelines
* Used to store and communicate review comments
* Report on defects and effort
* Provides aid for online reviews for large or geographically dispersed teams.

Static Analysis Tools

* Help developers and testers find defects prior to dynamic testing
* Provide support for enforcing coding standards
* Help in planning or risk analysis by providing the metrics for the code (e.g., complexity)

Modeling Tools

* Used to validate software models (e.g., physical data model for a RDBMS)
* Help in finding defects
* Aid in generating some test cases based on the model

Test Design Tools

* Generate test inputs or executable tests
* Generate test oracles from requirements, graphical user interfaces, design models or code

Test Data Preparation Tools

* Manipulate databases, files or data transmissions
* Set up test data to be used during the execution of tests
* Ensure security through data anonymity

Test Design Tools

* Generate test inputs or executable tests
* Generate test oracles from requirements, graphical user interfaces, design models or code

Test Data Preparation Tools

* Manipulate databases, files or data transmissions
* Set up test data to be used during the execution of tests
* Ensure security through data anonymity

Test Comparators

* Determine differences between files, databases or test results
* A test comparator may use a test oracle, especially if it is automated

Coverage Measurement Tools

* Measure the percentage of specific types of code structures that have been exercised through intrusive or non-intrusive means

Security Testing Tools

* Evaluates the security characteristics of software
* Evaluates the ability of the software to protect data confidentiality, integrity, authentication, authorization, availability, and non-repudiation

Dynamic Analysis Tools

* Find defects only when software is executing, such as time dependencies or memory leaks
* Used in component and component integration testing and when testing middleware

Performance Testing/Load Testing/Stress Testing Tools

* Monitor and report on how a system behaves under a variety of simulated usage conditions
* The simulation of load is achieved by creating virtual users (VUsers) carrying out a selected set of transactions

Monitoring Tools

* Continuously analyze, verify and report on usage of specific system resources
* Give warnings of possible service problems

Potential Benefits of using Tools

* Reduction of repetitive work
* Greater consistency and repeatability
* Objective assessment
* Ease of access to information about tests or testing

Risks of using Tools

* Unrealistic expectations from the tool
* Under estimating the time, cost and effort while initial introduction of a tool
* Under estimating the time and effort needed to achieve significant and continuing benefits from the tool
* Under estimating the effort required to maintain the test assets

**Defect Density**

Total Defect density = (Total number of defects including both impact and non-impact, found in all the phases + Post delivery defects)/Size

**Average Defect Age**

Average Defect age = (Sum of ((Defect detection phase number – defect injection phase number) \* No of defects detected in the defect detection phase))/(Total Number of defects till date)

**Defect Removal Efficiency**

DRE = 100 \* No. of pre-delivery defects / Total No. of Defects

**Review Effectiveness**

Review Effectiveness = 100 \* Total no. of defects found in review / Total no. of defects

**Cost of finding a defect in review(CFDR)**

Cost of finding a defect in reviews = (Total efforts spent on reviews / No. of defects found in reviews)

**Cost of finding a defect in testing(CFDT)**

Cost of finding a defect in testing = (Total efforts spent on testing / defects found in testing)

**Cost of Quality**

Components of CoQ – Prevention Cost, Appraisal Cost, Failure Cost

**Prevention Cost: (Green Money)**

Cost of time spent in DP meetings

Cost of time spent by DPR/PM/TL on analysis of defect entries/discussions with team members

Cost of time spent by the team in implementing the preventive actions identified from project start date to till date

**Appraisal Cost: (Blue Money)**

Cost of time spent on review and testing activities from the project start date to till date

**Failure Cost: (Red Money)**

Failure costs include internal and external failure costs

Cost of time taken to fix the pre and post delivery defects

Expenses incurred in rework – Customer does not pay for this

**Cost of Quality**

* + % Cost of Quality = (Total efforts spent on Prevention + Total efforts spent on Appraisal + Total efforts spent on failure or rework)\*100/(Total efforts spent on project)
  + Failure cost = Efforts spent on fixing or reworking the pre-delivery defects + (3 \* efforts spent on fixing or reworking the post-delivery defects)

**Test Case Effectiveness**

Test Case Effectiveness = # of defects detected using the test cases \* 100/ total # of defects detected in testing

This metrics defines the effectives of the test cases which is measured in terms of the number of defects found in testing with using the test cases

Types of Metrics:

There are several types of metrics

* + Project Metrics
  + Process Metrics
  + Productivity Metrics
  + Closure Metrics