**Testing link- http://istqbexamcertification.com/what-are-software-testing-levels/**

**Ad hoc testing**- it is a form of black box or behavioral testing performed without any formal process in place.

This is mainly done with the aim of trying to uncover defects or flaws which cannot be captured through traditional or formal processes followed during the testing cycle.

**Configuration management** determines clearly about the items that make up the software or system. These items include source code, test scripts, third-party software, hardware, data and both development and test documentation. Configuration management is also about making sure that these items are managed carefully, thoroughly and attentively during the entire project and product life cycle.

**Agile development model** is also a type of [Incremental model](http://istqbexamcertification.com/what-is-incremental-model-advantages-disadvantages-and-when-to-use-it/). Software is developed in incremental, rapid cycles. This results in small incremental releases with each release building on previous functionality. Each release is thoroughly [tested](http://istqbexamcertification.com/why-is-testing-necessary/) to ensure [software quality](http://istqbexamcertification.com/what-is-software-quality/) is maintained. It is used for time critical applications. Extreme Programming (XP) is currently one of the most well known agile [development life cycle model](http://istqbexamcertification.com/what-are-the-software-development-models/).

**Advantages of Agile model-**

- Working software is delivered frequently (weeks rather than months).

- Close, daily cooperation between business people and developers.

- Even late changes in requirements are welcomed

- Regular adaptation to changing circumstances.

- Face-to-face conversation is the best form of communication.

- Customer satisfaction by rapid, continuous delivery of useful software.

-- Unlike the [waterfall model](http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/) in agile model very limited [planning](http://istqbexamcertification.com/what-is-the-purpose-and-importance-of-test-plans/) is required to get started with the project. Agile assumes that the end users’ needs are ever changing in a dynamic business and IT world. Changes can be discussed and features can be newly effected or removed based on feedback. This effectively gives the customer the finished system they want or need.

**Disadvantages of Agile model:**

-In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.

- There is lack of emphasis on necessary designing and documentation

- The project can easily get taken off track if the customer representative is not clear what final outcome that they want.

1. few types of **integration testing:**
   * Big bang integration testing
   * Top down
   * Bottom up
   * Functional incremental

- Only senior programmers are capable of taking the kind of decisions required during the development process. Hence it has no place for newbie programmers, unless combined with experienced resources.

**CMM- Capability Maturity Model** is a bench-mark for measuring the maturity of anorganization’s software process. It is a methodology used to develop and refine an organization’s software development process. CMM can be used to assess an organization against a scale of five process maturity levels based on certain Key Process Areas (KPA). It describes the maturity of the company based upon the project the company is dealing with and the clients.

five maturity levels-

1. **Initial-** processes unpredictable, poorly controlled and reactive. Company has no standard process for software development. Nor does it have a project-tracking system that enables developers to predict costs or finish dates with any accuracy
2. **Managed-** Company has installed basic software management processes and controls. But there is no consistency or coordination among different groups.
3. **Defined-** Company has pulled together a standard set of processes and controls for the entire organization so that developers can move between projects more easily and customers can begin to get consistency from different groups.
4. **Quantitatively Managed-** In addition to implementing standard processes, company has installed systems to measure the quality of those processes across all projects.
5. **Optimizing-**Company has accomplished all of the above and can now begin to see patterns in performance over time, so it can tweak its processes in order to improve productivity and reduce defects in software development across the entire organization.

**test approaches** or **test strategy** is one of the most powerful factor in the success of the test effort and the accuracy of the test plans and estimates.

Types-

1) **Analytical:**

**2) Model Based**

**3) Methodical:**

**4) Process or standard compliant**

**5)** **Dynamic**

**6) Consultative or directed:**

**7) Regression-averse:**

**How do you know which strategies to pick or blend for the best chance of success?** There are many factors to consider, but let us highlight a few of the most important:

* **Risks:** Risk management is very important during testing, so consider the risks and the level of risk. For a well-established application that is evolving slowly, regression is an important risk, so regression-averse strategies make sense. For a new application, a risk analysis may reveal different risks if you pick a risk-based analytical strategy.
* **Skills:** Consider which skills your testers possess and lack because strategies must not only be chosen, they must also be executed. . A standard compliant strategy is a smart choice when you lack the time and skills in your team to create your own approach.
* **Objectives:** Testing must satisfy the needs and requirements of stakeholders to be successful. If the objective is to find as many defects as possible with a minimal amount of up-front time and effort invested – for example, at a typical independent test lab – then a dynamic strategy makes sense.
* **Regulations:** Sometimes you must satisfy not only stakeholders, but also regulators. In this case, you may need to plan a methodical test strategy that satisfies these regulators that you have met all their requirements.
* **Product:** Some products like, weapons systems and contract-development software tend to have well-specified requirements. This leads to synergy with a requirements-based analytical strategy.
* **Business:** Business considerations and business continuity are often important. If you can use a legacy system as a model for a new system, you can use a model-based strategy.

**Performance testing tools** are basically for system level testing, to see whether or notthe system will stand up to a high volume of usage. A **load testing** is tocheck thatthe system can handle its expected number of transactions. A **volume testing** is basicallytocheck that the system can handle a large amount of data, e.g. manyfields in a record, many records in a file, etc. A **stress testing** is one that goesbeyond the normal expected usage of the system (to see what would happenoutside its design expectations), with respect to load or volume.

**Load testing- Jmeter- How to user JMeter-**

<https://www.digitalocean.com/community/tutorials/how-to-use-apache-jmeter-to-perform-load-testing-on-a-web-server>

* Add a Thread Group- Add **Number of Threads (users)**:, **Number of Threads (users)**:, **Loop Count**:
* Add an HTTP Request Defaults- fill in the Server Name or IP field with the name or IP address of the web server you want to test.
* Add an HTTP Cookie Manager- If your web server uses cookies, you can add support for cookies
* Add an HTTP Request Sampler- fill in the Path with the item that you want each thread (user) to request
* Add a View Results in Table Listener- listeners are used to output the results of a load test
* Run the Basic Test Plan

Performancetesting- **http://www.guru99.com/performance-testing.html**

**The focus of Performance testing is checking a software program's**

* Speed - Determines whether the application responds quickly
* Scalability - Determines maximum user load the software application can handle.
* Stability - Determines if the application is stable under varying loads

Memory Leak testing-

**MEMOry leak testing is one of the whitebox testing  
  
tools  
1 boundary checker  
2 insure++  
3 electric fense  
4  glowcode  
5 ratonal purify**

On Android phones, memory leaks in Apps are one of the major reasons why Apps crash with an Out-of-memory exception,

Generally speaking, a memory leak in Java happens when you hold on to an Object long after its purpose has been served. The Java Garbage collector collects objects that don’t have any references to them, but if you’re holding on to some Object, the GC can’t clean it up.

. A common pattern is when you add Objects into a collection, but forget to remove them.

For android phone- Tool- Little Eye

► Ad Hoc Testing Testing contrived for only the specific purpose or problem at hand; testing not carefully planned in advance. ► Scenario Detailed description (specific instance) of a use case, including rules, exceptions, boundaries, limits, etc. ► Test Case A specific set of test data along with expected results for a particular test objective. ► Test Coverage Describes how much of a system has been tested. ► Test Design Describes how a feature or function shall be tested. ► Test Plan Test Strategy

► Test Procedure Describes the steps for executing a set of test cases and analyzing their results. ► Test Script Step by step description for specific tests. ► Test Strategy Describes the scope, approach, resources and schedule for the testing activities of the project. This includes defining what will be tested, who will perform testing, how testing will be managed, and the associated risks and contingencies. Also referred to as a Test Plan. ► Use Case Describes a sequence of interactions between a system and an external actor that results in the actor accomplishing a task that provides benefit to someone. An actor is a person or other entity external to the software system being specified who interacts with the system to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the customer community that will use the product.

Security testing links-

<http://www.softwaretestinghelp.com/security-testing-of-web-applications/>

<http://www.softwaretestinghelp.com/penetration-testing-guide/>

<http://www.softwaretestinghelp.com/penetration-testing-tools/>

**Traceability Matrix** is mapping the number of testcases that   
serves to cover the number requirements. Its nothing but   
Requirement versus no. of test cases. Its in bit high level.  
  
where as **Test coverage matrix** is no. of test cases covered   
for a all test scenario or for a screen/page.  
It is very much in low level trying to cover as much as   
possible(testcases covered).