***ASSIGNMENT 4.***

**1. Explain the various phases of SDLC.**

Given below are some phases, which are common in every SDLC model. Let us go through them to get an idea.

* **Requirement Gathering and Analysis:**

Requirement gathering and analysis is the first stage of any SDLC model. This phase is basically the brainstorming phase and often consists of sub-stages like Feasibility Analysis to check how much of the idea can be put into action.

If any particular software needs to be modified, the underlying problem(s) of that software is sorted out along with finding ways to solve it. If a brand new software is going to be developed, then every minute requirement regarding that software is looked in to. This implies that this stage involves maximum research and inputs from both the company that is developing the software and the client.

* **System Analysis:**

This is the second phase of SDLC where the entire system is defined in detail. In fact, it this stage wherein a detailed blueprint of various processes of the software is developed. If needed the system is divided into smaller parts to make it easier more manageable for the developers, designers, testers, project managers and other professionals who are going to work on the software in the latter stages.

* **System Design:**

In this phase, the physical system is designed with the help of the logical design prepared by system analysts. The analysts and designers work together and use certain tool and software to create the overall system design, include the probable output.

One stop solution for mobile management with MobileGo

* **Coding:**

As the name implies, in the stage software is coded with precision. A team of programmers are assigned by the company to work on software. More often than not, the work is sub-divided under a sub-phase called task allocation,where each developer is assigned a part of the work on his or her skill set(s). this helps complete the coding efficiently.

* **Testing:**

When the software is ready, it is sent to the testing department where Quality Analysts test it thoroughly for different error by forming various test cases. They either test the software manually or using automated testing tools and ensure that each and every component of the software works fine. Once the QA makes sure that the software is error-free, it goes to the next stage. Which is implementation.

* **Implimentation:**

Ths is the final stage of the software development life cycle. In this stage, if the software is run on various system by user. If it runs smoothly on these system without any flaw. Then it is considerd ready to be launched.

**2.Explain the various phases of STLC.**

* Requirements phase
* Planning Phase
* Analysis phase
* Design Phase
* Implementation Phase
* Execution Phase
* Conclusion Phase
* Closure Phase
* [**Requirement Phase**](http://www.softwaretestinghelp.com/rview-srs-document-and-create-test-scenarios-software-testing-training-course-day-2/)**:**

During this phase of STLC, analyze and study the requirements. Have brain storming sessions with other teams and try to find out whether the requirements are testable or not. This phase helps to identify the scope of the testing. If any feature is not testable, communicate it during this phase so that the mitigation strategy can be planned.

* [**Planning Phase**](http://www.softwaretestinghelp.com/how-to-write-test-plan-document-software-testing-training-day3/)**:**

In practical scenarios, Test planning is the first step of the testing process. In this phase we identify the activities and resources which would help to meet the testing objectives. During planning we also try to identify the metrics, the method of gathering and tracking those metrics.

* **Analysis Phase:**

This STLC phase defines “WHAT” to be tested. We basically identify the test conditions through the requirements document, product risks and other test basis. The test condition should be traceable back to the requirement. There are various factors which effect the identification of test conditions:

– Levels and depth of testing  
– Complexity of the product  
– Product and project risks  
– Software development life cycle involved.  
– Test management  
– Skills and knowledge of the team.  
– Availability of the stakeholders.

* **Design Phase:**

This phase defines “HOW” to test. This phase involves the following tasks:

– Detail the test condition. Break down the test conditions into multiple sub conditions to increase coverage.  
– Identify and get the test data  
– Identify and set up the test environment.  
– Create the requirement traceability metrics  
– Create the test coverage metrics.

* **Implementation Phase:**

The major task in this STLC phase is of creation of the detailed test cases. Prioritize the test cases also identify which test case will become part of the regression suite. Before finalizing the test case, It is important to carry out the review to ensure the correctness of the test cases. Also don’t forget to take the sign off of the test cases before actual execution starts. If your project involves automation, identify the candidate test cases for automation and proceed for scripting the test cases. Don’t forget to review them!

* [**Execution Phase**](http://www.softwaretestinghelp.com/test-execution-software-testing-qa-training-on-a-live-project-day-5/)**:**

As the name suggests, this is the Software Testing Life Cycle phase where the actual execution takes place. But before you start your execution, make sure that your entry criterion is met. Execute the test cases, log defects in case of any discrepancy. Simultaneously fill your traceability metrics to track your progress.

* [**Conclusion Phase**](http://www.softwaretestinghelp.com/bug-tracking-test-metrics-and-test-sign-off-free-qa-training-day-6/)**:**

This STLC phase concentrates on the exit criteria and reporting. Depending on your project and stakeholders choice, you can decide on reporting whether you want to send out a daily report of weekly report etc. There are different types of reports ( DSR – Daily status report, WSR – Weekly status reports) which you can send, but the important point is, the content of the report changes and depends upon whom you are sending your reports. If Project managers belong to testing background then they are more interested in the technical aspect of the project, so include the technical things in your report ( number of test cases passed, failed, defects raised, severity 1 defects etc.). But if you are reporting to upper stakeholders, they might not be interested in the technical things so report them about the risks that have been mitigated through the testing.

* **Closure Phase:**

Tasks for the closure activities include the following:

– Check for the completion of the test. Whether all the test cases are executed or mitigated deliberately. Check there are no severity 1 defects opened.  
– Do lessons learnt meeting and create lessons learnt document.( Include what went well, where are the scope of improvements and what can be improved)

**3. Explain Traditional Software Development Model.**

# Waterfall model:-

The waterfall model is a popular version of the systems development life cycle model for software engineering. Often considered the classic approach to the systems development life cycle, the waterfall model describes a development method that is linear and sequential. Waterfall development has distinct goals for each phase of development. Imagine a waterfall on the cliff of a steep mountain. Once the water has flowed mountain, it cannot turn back. It is the same with waterfall development. Once a phase of development is completed, the development proceeds to the next phase and there is no turning back.

**Advantages:**

* testing is inherent to every phases of the waterfall model.
* It is an enforced disciplined approached

**Disadvantages :**

* it only incorporation indirectly, thus changes may cause considerable confusion as the project progresses.
* As the client usally only has a vague idea of exactly where is required from the software product, is model has difficult accommodating the natural uncertain that exists at the beginning of the project.

**4. Explain Prototype Software development Model.**

The basic idea in **Prototype model** is that instead of freezing the requirements before a design or coding can proceed, a throwaway prototype is built to understand the requirements. This prototype is developed based on the currently known requirements. Prototype model is a [**software development model**](http://istqbexamcertification.com/what-are-the-software-development-models/). By using this prototype, the client can get an “actual feel” of the system, since the interactions with prototype can enable the client to better understand the requirements of the desired system.  Prototyping is an attractive idea for complicated and large systems for which there is no manual process or existing system to help determining the requirements.

**Advantage**:

* generates working software quickly and early during the software life cycle
* more flexible- less costly to changes scope and requirements.
* Each iteration is an easily managed milestone.

**Disadvantage:**

* Each phase of an iteration is rigid and do not overlap each other.
* Problems may arise pertaining to system architecture because not all requirements are gathered up front for the entire software life cycle.

**5. Explain Iterative enhancement life cycle model.**

The iterative enhancement life cycle model counters the third limitation of the [waterfall model](http://www.freetutes.com/systemanalysis/sa2-waterfall-software-life-cycle.html) and tries to combine the benefits of both [prototyping](http://www.freetutes.com/systemanalysis/sa2-prototyping-model.html) and the [waterfall model](http://www.freetutes.com/systemanalysis/sa2-waterfall-software-life-cycle.html). The basic idea is that the software should be developed in increments, where each increment adds some functional capability to the system until the full system is implemented. At each step extensions and design modifications can be made. An advantage of this approach is that it can result in better testing, since testing each increment is likely to be easier than testing entire system like in the waterfall model. Furthermore, as in [prototyping](http://www.freetutes.com/systemanalysis/sa2-prototyping-model.html), the increments provides feedback to the client which is useful for determining the final requirements of the system.

In the first step of iterative enhancement model, a simple initial implementation is done for a subset of the overall problem. This subset is the one that contains some of the key aspects of the problem which are easy to understand and implement, and which forms a useful and usable system. A project control list is created which contains, in an order, all the tasks that must be performed to obtain the final implementation. This project control list gives an idea of how far the project is at any given step from the final system.

**6. Explain various types of Maintenance phase.**

After implementation of the system, maintenance phase looks for all the activity that is mandatory to keep the system operational. There are two major types of maintenance corrective maintenance and adaptive maintenance.

**Corrective Maintenance:**

Even after putting extra efforts and expertise on board for developing flawless software, it is practically not possible to have fault free system. We do look into the system even after installed in production for defects. The defects popped up during the usage of the system need to be fixed and this type of fixing will fall under corrective maintenance.

**Adaptive Maintenance:**

It has been observed that user come up with some uncovered requirement which was not a part of requirement analysis phase, but they got to know new things only after using the system. Usage of the system helps the user to define their requirement more accurately.  This requirement may cause the modification is accepted and installed system, this type of medication fall under adaptive maintenance.

**Effective Maintenance required following things.**

* Understand Existing code and related document
* Making the changes in code and related documents
* Testing the changes were done during adaptive and corrective maintenance
* Retesting all old parts, those were not the part of changes.

**7. What is the difference between High level design and Low level design?**

# High Level Design or System Design (HLD)

High  level Design gives the overall System Design in terms of **Functional Architecture and Database design**. This is very useful for the developers to understand the flow of the system. In this phase design team, review team (testers) and customers plays a major role. For this the entry criteria are the requirement document that is SRS. And the exit criteria will be HLD, projects standards, the functional design documents, and the database design document.

**Low  Level Design (LLD)**

During the detailed phase, the view of the application developed during the high level design is broken down into modules and programs. Logic design is done for every program and then documented as **program specifications**. For every program, a **unit test** plan is created.

The entry criteria for this will be the HLD document. And the exit criteria will the program specification and unit test plan (LLD).

**8. Explain V- Shaped Model.**

V- model means Verification and Validation model. Just like the [**waterfall model**](http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/), the V-Shaped life cycle is a sequential path of execution of processes. Each phase must be completed before the next phase begins. **V-Model** is one of the [**many software development models**](http://istqbexamcertification.com/what-are-the-software-development-models/).Testing of the product is planned in parallel with a corresponding phase of development in **V-model**.

**Advantage:**

* simple and easy to use.
* Each phase has specific deliverable
* Works well for small projects where requirement are easily understood.

**Disadvantage:**

* very rigid, like waterfall model.
* Less flexibility and adjusting scope is difficult and expensive.

**9. what is difference between functional specification or business requirement specification?**

While both business and functional requirements related to the same project, there are major differences between the two. Both sets of requirements contribute to a common goal, although functional requirements are much more specific and detailed. While business requirements deal with mainly business goals and stakeholder expectations, functional requirements outline exactly how a project will support business requirements. A business requirement tells us what the future state of a project is and why the objective is worthwhile, while functional requirements tell us how we will get there. Functional requirements outline specific steps and outline how the project will be delivered. As a result, they help ensure a project is on track and are used for measuring performance.

**10. What is the difference between testing and debugging?**

|  |  |
| --- | --- |
| Testing | Debugging |
| 1. Testing always starts with known conditions, uses predefined methods, and has predictable outcomes too. | 1. Debugging starts from possibly un-known initial conditions and its end cannot be predicted, apart from statistically. |
| 2. Testing can and should definitely be planned, designed, and scheduled. | 2. The procedures for, and period of, debugging cannot be so constrained. |
| 3. It proves a programmers failure. | 3. It is the programmer’s vindication. |
| 4. It is a demonstration of error or apparent correctness. | 4. It is always treated as a deductive process. |
| 5. Testing as executed should strive to be predictable, dull, constrained, rigid, and inhuman. | 5. Debugging demands intuitive leaps, conjectures, experimentation, and some freedom also. |
| 6. Much of the testing can be done without design knowledge. | 6. Debugging is impossible without detailed design knowledge. |
| 7. It can often be done by an outsider. | 7. It must be done by an insider. |
| 8. Much of test execution and design can be automated. | 8. Automated debugging is still a dream for programmers. |
| 9. Testing purpose is to find bug. | 9. Debugging purpose is to find cause of bug. |