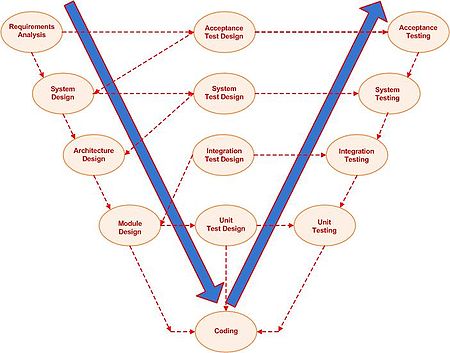
**Software Engineering Interview Questions**

**Imp Note: you have search for more information about these topics and to make sure about these information**

1. **V-Model (software development)**

**[](http://en.wikipedia.org/wiki/File:V-model.JPG)**

The V-model is a software development process.

The **V-model** is a [software development process](http://en.wikipedia.org/wiki/Software_development_process) which can be presumed to be the extension of the [waterfall model](http://en.wikipedia.org/wiki/Waterfall_model). Instead of moving down in a linear way, the process steps are bent upwards after the [coding](http://en.wikipedia.org/wiki/Source_code) phase, to form the typical V shape. The V-Model demonstrates the relationships between each phase of the development life cycle and its associated phase of [testing](http://en.wikipedia.org/wiki/Software_testing).

The V-model deploys a well-structured method in which each phase can be implemented by the detailed documentation of the previous phase. Testing activities like test designing start at the beginning of the project well before coding and therefore saves a huge amount of the project time.

1. **Software Requirements are Descriptions and specifications of a system**

**Requirements Engineering**

- Eliciting, organizing, and documenting the requirements of the system.

**Software requirements should be**

•Clear

•complete

•Unambiguous

•Quantitative

**Functional Requirements** are the Expected functionality or services from the system.  
**Non - Functional Requirements** are System property and constraints.

1. **Different types of Architectures in Software Engineering**:

\*Object-Oriented/Abstract Data Style  
\* Layered Hierarchies  
\* Blackboard architecture  
- involving shared access to data with loosely coupled agents.  
\* Client/Server  
\* Three tier Client/Server architectures  
\* Peer-to-Peer Architecture

1. **Sequence Diagrams**: They are pictorial representation of event happening according to the time line.

**Collaboration Diagrams**: Are an alternative presentation of a sequence diagram. We use boxes to describe objects.

**Package Diagram:** Is complete set of sequence Diagrams or Collaboration Diagrams of the system.

1. **Object Oriented Design Tools** - Rational Rose, Rhapsody, Telelogic Tau G2 UI Design Tools - Rapid/Paper prototypes, Simple  
   SUMI (Software Usability Measurement Inventory).
2. What is **SDLC**  
   a software cycle deals with various parts and phases from planning to testing and deploying. All these activities are carried out in different ways, as per the needs. Each way is known as a **Software Development Lifecycle Model** (SDLC).

Some SDLC models:

\* The Linear model (Waterfall)   
- Separate and distinct phases of specification and development.  
- All activities in linear fashion.  
- Next phase starts only when first one is complete.  
\* Evolutionary development  
- Specification and development are interleaved (Spiral, incremental, prototype based, Rapid Application development).  
- Incremental Model (Waterfall in iteration)  
- RAD(Rapid Application Development)

- Focus is on developing quality product in less time.  
- Spiral Model

- We start from smaller module and keeps on building it like a spiral.

- It is also called Component based development.  
\* Formal systems development  
- A mathematical system model is formally transformed to an implementation.  
\* Agile Methods  
- Inducing flexibility into development  
\* Reuse-based development  
- The system is assembled from existing components.

1. What is **testing** in Software Engineering  
   Testing is running the program (or product) under various circumstances and conditions to find errors and bugs in it.

**Unit Testing**   
- The Tools used in Unit Testing are debuggers, tracers and is done by Programmers. Unit testing verifies the functioning in isolation of software pieces which are separately testable.

**Integration Testing**  
- According to IEEE, Integration Testing is an orderly progression of testing in which software elements, hardware elements, or both are combined and tested, to evaluate their interactions, until the entire system has been integrated. It Test against system design and Focuses on communication between modules start with one module, and then add incrementally.

Various Types of Integration Testing are:

\* Big bang approach (Usage Model testing)

- Integrate Everything at once.

- Can be used in both software and hardware integration testing.

Advantages

- Saving time in the integration testing process.

- Suitable for small-sized and simple systems with limited number of components.

- Suitable when modules have been heavily unit tested

- Helps for showing the overall system behavior early.

Disadvantages

- If the test cases and their results are not recorded properly, the entire integration process will be more complicated and may prevent the testing team from achieving the goal of integration testing.

- Risky for systems with large number of components.

- Complicates the debugging process.

- Integration cannot start before all system modules are ready.

\* Top-down approach

- Keep on breaking the system in parts one by one and then test each part.

Advantages

- It is easier to find a missing branch link.

- Easier to logically understand and helps for seeing the big picture first.

Disadvantages

- In most cases, it defers the bulk of the complex processing until later.

\* Bottom-up approach

- Test the small parts first and then keep on integrating the system and keep on testing the bigger module of it.

Advantages

- Is that bugs are more easily found.

- In most cases, it checks for the bulk of the complex processing at the beginning.

Disadvantages

- Harder to logically understand and harder to see the big picture until later.

\* Mixed approach (Sandwich Testing)

- Is an approach to combine top down testing with bottom up testing.

**Functional Testing**- It checks that the functional specifications are correctly implemented. - Can also check if non functional behavior is as per expectations.

\*Stress testing (aka endurance testing).

**Configuration Testing**  
- It is used when software meant for different types of users. It also checks that whether the software performs for all users.   
Smoke (Sanity Test).   
- It is used to verify whether the build is ready for feature/requirement (based testing).   
Recovery Testing  
- It is used in verifying software restart capabilities after a “disaster”   
(Security Testing).  
- It is used to verify proper controls have been designed.

**System Testing** (Found in the slides of testing course we have taken)

Some of its types:

Regression Testing  
- Regression Testing according to IEEE is "selective retesting of a system or component to verify that modifications have not caused unintended effects". It is repetition of tests intended to show that the software’s behavior is unchanged, except insofar as required. It can be done at each test level.

Performance Testing  
- It is verifying that the software meets the specified performance requirements (response time, volume ...).

Usability Testing  
- It evaluates the Human Computer Interface.

- Verifies for ease of use by end-users.

- Verifies ease of learning the software, including user documentation. - Checks how effectively the software functions in supporting user tasks.

- Checks the ability to recover from user errors.

**Acceptance Testing** (Found in the slides of testing course we have taken)

One of its types:

Alpha / Beta testing- Probably one term which you must be aware of as we often hears this software is in Alpha phase and in Beta phase. Here Testing is done by representative set of potential users for trial use. Please Note - in-house (alpha testing) - external (beta testing).

1. Testing Tools for unit testing and integration testing.
2. **Black box** is a technical term for a device, system or object when it is viewed in terms of its input, output and [transfer characteristics](http://en.wikipedia.org/wiki/Transfer_function) without any knowledge required of its internal workings. Almost anything might occasionally be referred to as a black box: a [transistor](http://en.wikipedia.org/wiki/Transistor), an [algorithm](http://en.wikipedia.org/wiki/Algorithm), [humans](http://en.wikipedia.org/wiki/Humans), the [Internet](http://en.wikipedia.org/wiki/Internet).

The opposite of a black box is a system where the inner components or logic are available for inspection (such as a [free software](http://en.wikipedia.org/wiki/Free_software)/[open source](http://en.wikipedia.org/wiki/Open-source_software) program), which is sometimes known as a [white box](http://en.wikipedia.org/wiki/White_box_(software_engineering)), a glass box, or a clear box.

In [software engineering](http://en.wikipedia.org/wiki/Software_engineering) **white box**, in contrast to a [black box](http://en.wikipedia.org/wiki/Black_box_(systems)), is a [subsystem](http://en.wikipedia.org/wiki/Subsystem) whose internals can be viewed, but usually cannot be altered. This is useful during [white box testing](http://en.wikipedia.org/wiki/White_box_testing), where a system is examined to make sure that it fulfills its requirements.

Having access to the subsystem internals in general makes the subsystem easier to understand, but also easier to [hack](http://en.wikipedia.org/wiki/Hack_(technology_slang)); if a programmer, for example, can examine source code, weaknesses in an algorithm are much easier to discover. This makes white box testing much more effective than [black box testing](http://en.wikipedia.org/wiki/Black_box_testing), but considerably more difficult due to the sophistication needed on the part of the tester to understand the subsystem.

Also known as **glass box**, **clear box**, or **open box**.

White box also refers to a system that is visible, but which is so complex that it might as well be a [Black box](http://en.wikipedia.org/wiki/Black_box). This follows from a [Murphy's Law](http://en.wikipedia.org/wiki/Murphy%27s_law) of computer programming: Program complexity grows until it exceeds the capability of the programmer who must maintain it.

There is more information about black and white box in the slides