Lecture Notes
On
SRS and Software Testing

UNIT-IV

Prepared by,
Alok Haldar
Assistant Professor,
Department of Computer Science and BCA,

Kharagpur College

What is Software Requirement Specification - [SRS]

A software requirements specification (SRS) is a document that captures complete description about how the system is expected to perform.

Qualities of SRS:

- Correct
- Unambiguous
- Complete
- Consistent
- Ranked for importance and/or stability
- Verifiable
- Modifiable
- Traceable

Types of Requirements:

The below diagram depicts the various types of requirements that are captured during SRS.

Following are the characteristics of a good SRS document:

1. **Correctness:**
   User review is used to ensure the correctness of requirements stated in the SRS. SRS is said to be correct if it covers all the requirements that are actually expected from the system.
2. **Completeness:**
   Completeness of SRS indicates every sense of completion including the numbering of all the pages, resolving the to be determined parts to as much extent as possible as well as covering all the functional and non-functional requirements properly.
3. **Consistency:**
   Requirements in SRS are said to be consistent if there are no conflicts between any set of requirements. Examples of conflict include differences in terminologies used at separate places, logical conflicts like time period of report generation, etc.**Unambiguousness:**
   A SRS is said to be unambiguous if all the requirements stated have only 1 interpretation. Some of the ways to prevent unambiguousness include the use of modelling techniques like ER diagrams, proper reviews and buddy checks, etc.
4. **Ranking for importance and stability:**
   There should a criterion to classify the requirements as less or more important or more specifically as desirable or essential. An identifier mark can be used with every requirement to indicate its rank or stability.
5. **Modifiability:**
   SRS should be made as modifiable as possible and should be capable of easily accepting changes to the system to some extent. Modifications should be properly indexed and cross-referenced.
6. **Verifiability:**
   A SRS is verifiable if there exists a specific technique to quantifiably measure the extent to which every requirement is met by the system. For example, a requirement stating that the system must be user-friendly is not verifiable and listing such requirements should be avoided.
7. **Traceability:**
   One should be able to trace a requirement to design component and then to code segment in the program. Similarly, one should be able to trace a requirement to the corresponding test cases.
8. **Design Independence:**
   There should be an option to choose from multiple design alternatives for the final system. More specifically, the SRS should not include any implementation details.
9. **Testability:**
   A SRS should be written in such a way that it is easy to generate test cases and test plans from the document.
10. **Understandable by the customer:**

An end user maybe an expert in his/her specific domain but might not be an expert in computer science. Hence, the use of formal notations and symbols should be avoided to as much extent as possible. The language should be kept easy and clear.

11.**Right level of abstraction:**
If the SRS is written for the requirements phase, the details should be explained explicitly. Whereas, for a feasibility study, fewer details can be used. Hence, the level of abstraction varies according to the purpose of the SRS.

Software Testing

Software testing is widely used technology because it is compulsory to test each and every software before deployment.

Software testing includes

1. Black Box Testing,

2. White Box Testing,

3. Visual Box Testing

4. Gray Box Testing.

Levels such as

I. Unit Testing,

II. Integration Testing,

III. Regression Testing,

IV. Functional Testing.

V. System Testing,

VI. Acceptance Testing,

VII. Alpha Testing,

VIII.Beta Testing,

IX. Non-Functional testing,

X. Security Testing,

XI. Portability Testing.

What is Software Testing

Software testing is a process of identifying the correctness of software by considering its all attributes such as

I. Reliability,

II. Scalability,

III. Portability,

IV. Re-usability,

V. Usability

and evaluating the execution of software components to find the software bugs or errors or defects.

Testing is mandatory because it will be a dangerous situation if the software fails any of time due to lack of testing.

So, without testing software cannot be deployed to the end user.

Software Testing can be classified into two categories:

1. **Black Box Testing** is a software testing method in which the internal structure/ design/ implementation of the item being tested is not known to the tester

2. **White Box Testing** is a software testing method in which the internal structure/ design/ implementation of the item being tested is known to the tester.

Differences between Black Box Testing vs White Box Testing:

Black Box Testing	White Box Testing
It is a way of software testing in which the internal structure or the program or the code is hidden and nothing is known about it.	It is a way of testing the software in which the tester has knowledge about the internal structure or the code or the program of the software.
It is mostly done by software testers.	It is mostly done by software developers.
No knowledge of implementation is needed.	Knowledge of implementation is required.
It can be referred as outer or external software testing.	It is the inner or the internal software testing.
It is functional test of the software.	It is structural test of the software.
This testing can be initiated on the basis of requirement specifications document.	This type of testing of software is started after detail design document.
No knowledge of programming is required.	It is mandatory to have knowledge of programming.
It is the behavior testing of the software.	It is the logic testing of the software.
It is applicable to the higher levels of testing of software.	It is generally applicable to the lower levels of software testing.
It is also called closed testing.	It is also called as clear box testing.
It is least time consuming.	It is most time consuming.
It is not suitable or preferred for algorithm testing.	It is suitable for algorithm testing.
Can be done by trial and error ways and methods.	Data domains along with inner or internal boundaries can be better tested.
Example: search something on google by using keywords	**Example:** by input to check and verify loops

Types of Black Box Testing:

- A. Functional Testing
- B. Non-functional testing
- C. Regression Testing

Types of White Box Testing:

- A. Path Testing
- B. Loop Testing
- C. Condition testing

Types of Testing:-

1. Unit Testing

It focuses on the smallest unit of software design. In this, we test an individual unit or group of interrelated units. It is often done by the programmer by using sample input and observing its corresponding outputs.
Example:

a) In a program we are checking if loop, method or function is working fine

b) Misunderstood or incorrect, arithmetic precedence.

c) Incorrect initialization

2. Integration Testing

The objective is to take unit tested components and build a program structure that has been dictated by design. Integration testing is testing in which a group of components is combined to produce output.

Integration testing is of four types: (i) Top-down (ii) Bottom-up (iii) Sandwich (iv) Big-Bang
Example:

(a) Black Box testing:- It is used for validation. In this we ignore internal working mechanism and focuse on **what is the output?**.

(b) White Box testing:- It is used for verification. In this we focus on internal mechanism i.e. **how the output is achieved?**

3. Regression Testing

Every time a new module is added leads to changes in the program. This type of testing makes sure that the whole component works properly even after adding components to the complete program.
Example :

In school record suppose we have module staff, students and finance combining these modules and checking if on integration these module works fine is regression testing.

4. Smoke Testing

This test is done to make sure that software under testing is ready or stable for further testing
It is called a smoke test as the testing an initial pass is done to check if it did not catch the fire or smoke in the initial switch on.
Example:

If project has 2 modules so before going to module make sure that module 1 works properly.

5. Alpha Testing

This is a type of validation testing. It is a type of *acceptance testing* which is done before the product is released to customers. It is typically done by QA people.
Example:
When software testing is performed internally within the organization

6. Beta Testing

The beta test is conducted at one or more customer sites by the end-user of the software. This version is released for a limited number of users for testing in a real-time environment
Example:

When software testing is performed for the limited number of people.

7. System Testing

This software is tested such that it works fine for the different operating systems. It is covered under the black box testing technique. In this, we just focus on the required input and output without focusing on internal working.
In this, we have security testing, recovery testing, stress testing, and performance testing
Example:

This include functional as well as non functional testing

8. Stress Testing

In this, we give unfavorable conditions to the system and check how they perform in those conditions.
Example:

(a) Test cases that require maximum memory or other resources are executed
(b) Test cases that may cause thrashing in a virtual operating system
(c) Test cases that may cause excessive disk requirement

9. Performance Testing

It is designed to test the run-time performance of software within the context of an integrated system. It is used to test the speed and effectiveness of the program. It is also called load testing. In it we check, what is the performance of the system in the given load.
Example:

```
Checking number of processor cycles
```

10. Object-Oriented Testing

This testing is a combination of various testing techniques that help to verify and validate object-oriented software. This testing is done in the following manner:

- Testing of Requirements,
- Design and Analysis of Testing,
- Testing of Code,
- Integration testing,
- System testing,
- User Testing.