Static Techniques

By design we mean to create a plan for how to implement an idea and technique is a method or way for performing a task. So, Test Design is creating a set of inputs for given software that will provide a set of expected outputs. The idea is to ensure that the system is working good enough and it can be released with as few problems as possible for the average user.

Broadly speaking there are two main categories of Test Design Techniques. They are:

1. Static Techniques
2. Dynamic Techniques

Below is the tree structure of the testing techniques:

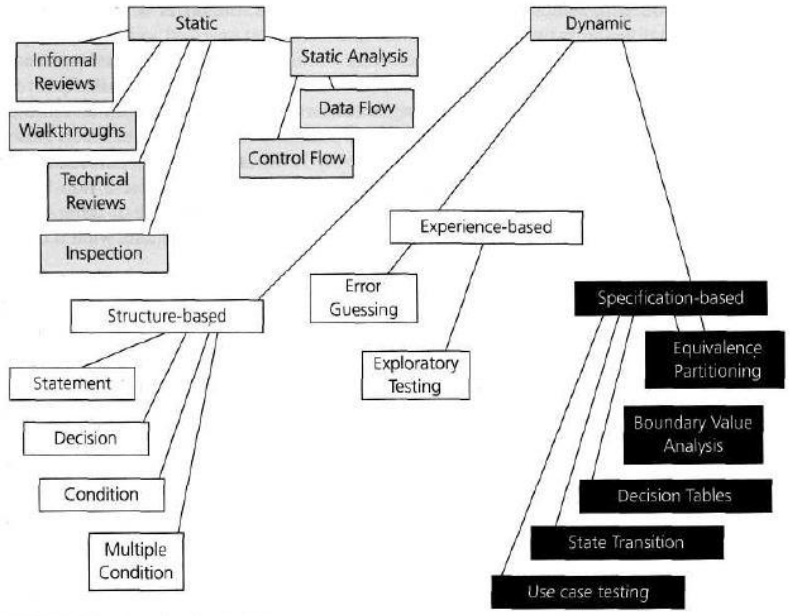

FIGURE 4.1 Testing techniques

What is a static test technique?

Static test techniques provide a great way to improve the quality and productivity of software development. It includes the reviews and provides the overview of how they are conducted. The primary objective of static testing is to improve the quality of software products by assisting engineers to recognize and fix their own defects early in the software development process.

What is static Testing?

- Static testing is the testing of the software work products manually, or with a set of tools, but they are **not executed**.
- It starts early in the Life cycle and so it is done during the verification process.
- It does not need computer as the testing of program is done without executing the program. For example: reviewing, walk through, inspection, etc.

What are the uses of Static Testing?

The uses of static testing are as follows:

- Since static testing can start early in the life cycle so early feedback on quality issues can be established.
- As the defects are getting detected at an early stage so the rework cost most often relatively low.
- Development productivity is likely to increase because of the less rework effort.
- **Types of the defects that are easier to find during the static testing** are: deviation from standards, missing requirements, design defects, non-maintainable code and inconsistent interface specifications.
- Static tests contribute to the increased awareness of quality issues.

What is Informal reviews?

Informal reviews are applied many times during the early stages of the life cycle of the document. A two person team can conduct an informal review. In later stages these reviews often involve more people and a meeting. The goal is to keep the author and to improve the quality of the document. The most important thing to keep in mind about the informal reviews is that they are **not documented.**

Formal reviews follow a formal process. It is well structured and regulated.
A formal review process consists of six main steps:

1. Planning
2. Kick-off
3. Preparation
4. Review meeting
5. Rework
6. Follow-up

1. Planning: The first phase of the formal review is the Planning phase. In this phase thereview process begins with a request for review by the author to the moderator (or inspection leader). A moderator has to take care of the scheduling like date, time, place and invitation of the review. For the formal reviews the moderator performs the entry check and also defines the formal exit criteria. The **entry check** is done to ensure that the reviewer's time is not wasted on a document that is not ready for review. After doing the entry check if the document is found to have very little defects then it's ready to go for the reviews. So, the **entry criteria** are to check that whether the document is ready to enter the formal review process or not. Hence the entry criteria for any document to go for the reviews are:

- The documents should not reveal a large number of major defects.
- The documents to be reviewed should be with line numbers.
- The documents should be cleaned up by running any automated checks that apply.
- The author should feel confident about the quality of the document so that he can join the review team with that document.

Once, the document clear the entry check the moderator and author decides that which part of the document is to be reviewed. Since the human mind can understand only a limited set of pages at one time so in a review the maximum size is between 10 and 20 pages. Hence checking the documents improves the moderator ability to lead the meeting because it ensures the better understanding.

2. Kick-off: This kick-off meeting is an optional step in a review procedure. The goal of this step is to give a short introduction on the objectives of the review and the documents to everyone in the meeting. The

relationships between the document under review and the other documents are also explained, especially if the numbers of related documents are high. At customer sites, we have measured results up to 70% more major defects found per page as a result of performing a kick-off, [van Veenendaal and van der Zwan, 2000].

3. Preparation: In this step the reviewers review the document individually using the related documents, procedures, rules and checklists provided. Each participant while reviewing individually identifies the defects, questions and comments according to their understanding of the document and role. After that all issues are recorded using a logging form. The success factor for a thorough preparation is the number of pages checked per hour. This is called the **checking rate.** Usually the checking rate is in the range of 5 to 10 pages per hour.

4. Review meeting: The review meeting consists of three phases:

- **Logging phase:** In this phase the issues and the defects that have been identified during the preparation step are logged page by page. The logging is basically done by the author or by a **scribe.** Scribe is a separate person to do the logging and is especially useful for the formal review types such as an inspection. Every defects and it's severity should be logged in any of the three severity classes given below:

  – Critical: The defects **will cause** downstream damage.
  – **Major:** The defects **could cause** a downstream damage.
  – **Minor:** The defects are **highly unlikely to cause** the downstream damage.

During the logging phase the moderator focuses on logging as many defects as possible within a certain time frame and tries to keep a good logging rate (number of defects logged per minute). In formal review meeting the good logging rate should be between one and two defects logged per minute.

- **Discussion phase:** If any issue needs discussion then the item is logged and then handled in the discussion phase. As chairman of the discussion meeting, the moderator takes care of the people issues and prevents discussion from getting too personal and calls for a break to cool down the heated discussion. The outcome of the discussions is documented for the future reference.

- **Decision phase:** At the end of the meeting a decision on the document under review has to be made by the participants, sometimes based on formal **exit criteria. Exit criteria** are the average number of critical and/or major defects found per page (for example no more than three critical/major defects per page). If the number of defects found per page is more than a certain level then the document must be reviewed again, after it has been reworked.

5. **Rework:** In this step if the number of defects found per page exceeds the certain level then the document has to be reworked. Not every defect that is found leads to rework. It is the author's responsibility to judge whether the defect has to be fixed. If nothing can be done about an issue then at least it should be indicated that the author has considered the issue.

6. **Follow-up:** In this step the moderator check to make sure that the author has taken action on all known defects. If it is decided that all participants will check the updated documents then the moderator takes care of the distribution and collects the feedback. It is the responsibility of the moderator to ensure that the information is correct and stored for future analysis.

During a review four types of participants take part. They are:

1. **The moderator:**

- Also known as review leader

- Performs entry check
- Follow-up on the rework
- Schedules the meeting
- Coaches other team
- Leads the possible discussion and stores the data that is collected

2. **The author:**

    - Illuminate the unclear areas and understand the defects found
    - Basic goal should be to learn as much as possible with regard to improving the quality of the document.

3. **The scribe:**

- *Scribe is a separate person to do the logging of the defects found during the review.*

4. **The reviewers:**

- *Also known as checkers or inspectors*
- *Check any material for defects, mostly prior to the meeting*
- The manager can also be involved in the review depending on his or her background.

5. **The managers:**

- *Manager decides on the execution of reviews*
- *Allocates time in project schedules and determines whether review process objectives have been met*

The main review types that come under the static testing are mentioned below:

1. **Walkthrough:**

- It is not a formal process
- It is led by the authors
- Author guide the participants through the document according to his or her thought process to achieve a common understanding and to gather feedback.
- Useful for the people if they are not from the software discipline, who are not used to or cannot easily understand software development process.
- Is especially useful for higher level documents like requirement specification, etc.

The goals of a walkthrough:

- To present the documents both within and outside the software discipline in order to gather the information regarding the topic under documentation.
- To explain or do the knowledge transfer and evaluate the contents of the document
- To achieve a common understanding and to gather feedback.
- To examine and discuss the validity of the proposed solutions

2. **Technical review:**

- It is less formal review
- It is led by the trained moderator but can also be led by a technical expert
- It is often performed as a peer review without management participation

- Defects are found by the experts (such as architects, designers, key users) who focus on the content of the document.
- In practice, technical reviews vary from quite informal to very formal

The goals of the technical review are:

- To ensure that an early stage the technical concepts are used correctly
- To access the value of technical concepts and alternatives in the product
- To have consistency in the use and representation of technical concepts
- To inform participants about the technical content of the document

3. Inspection:

- It is the most formal review type
- It is led by the trained moderators
- During inspection the documents are prepared and checked thoroughly by the reviewers before the meeting
- It involves peers to examine the product
- A separate preparation is carried out during which the product is examined and the defects are found
- The defects found are documented in a logging list or issue log
- A formal follow-up is carried out by the moderator applying exit criteria

The goals of inspection are:

- It helps the author to improve the quality of the document under inspection
- It removes defects efficiently and as early as possible
- It improve product quality
- It create common understanding by exchanging information
- It learn from defects found and prevent the occurrence of similar defects

Walkthrough:

- It is not a formal process/review
- It is led by the authors
- Author guide the participants through the document according to his or her thought process to achieve a common understanding and to gather feedback.
- Useful for the people if they are not from the software discipline, who are not used to or cannot easily understand software development process.
- Is especially useful for higher level documents like requirement specification, etc.

The goals of a walkthrough:

i. To present the documents both within and outside the software discipline in order to gather the information regarding the topic under documentation.
ii. To explain or do the knowledge transfer and evaluate the contents of the document
iii. To achieve a common understanding and to gather feedback.
iv. To examine and discuss the validity of the proposed solutions

What is Technical review in software testing?

Technical review:

- It is less formal review
- It is led by the trained moderator but can also be led by a technical expert
- It is often performed as a peer review without management participation
- Defects are found by the experts (such as architects, designers, key users) who focus on the content of the document.
- In practice, technical reviews vary from quite informal to very formal

The goals of the technical review are:

- To ensure that an early stage the technical concepts are used correctly
- To access the value of technical concepts and alternatives in the product
- To have consistency in the use and representation of technical concepts
- To inform participants about the technical content of the document

What is Inspection in software testing?

Inspection:

- It is the most formal review type
- It is led by the trained moderators
- During inspection the documents are prepared and checked thoroughly by the reviewers before the meeting
- It involves peers to examine the product
- A separate preparation is carried out during which the product is examined and the defects are found
- The defects found are documented in a logging list or issue log
- A formal follow-up is carried out by the moderator applying exit criteria

The goals of inspection are:

i. It helps the author to improve the quality of the document under inspection
ii. It removes defects efficiently and as early as possible
iii. It improve product quality
iv. It create common understanding by exchanging information
v. It learn from defects found and prevent the occurrence of similar defects

What is static analysis?

- Performed on requirement design or code without actually executing the software or before the code is actually run.
- Goal of static analysis is to find the defects whether or not they may cause failure.
- Static analysis find defects rather than failures.

What is Static analysis tools in software testing?

- **Static analysis tools** are generally used by developers as part of the development and component testing process. The key aspect is that the code (or other artefact) is not executed or run but the tool itself is executed, and the source code we are interested in is the input data to the tool.

- These tools are **mostly used by developers.**
- Static analysis tools are an extension of compiler technology – in fact some compilers do offer static analysis features. It is worth checking what is available from existing compilers or development environments before looking at purchasing a more sophisticated static analysis tool.
- Other than software code, static analysis can also be carried out on things like, static analysis of requirements or static analysis of websites (for example, to assess for proper use of accessibility tags or the following of HTML standards).
- Static analysis tools for code can help the developers to understand the structure of the code, and can also be used to enforce coding standards.

Features or characteristics of static analysis tools are:

- *To calculate metrics such as cyclomatic complexity or nesting levels (which can help to identify where more testing may be needed due to increased risk).*
- *To enforce coding standards.*
- *To analyze structures and dependencies.*
- *Help in code understanding.*
- *To identify anomalies or defects in the code.*

What is traceability in Software testing?

Test conditions should be able to be linked back to their sources in the test basis, this is known as **traceability.** Traceability can be horizontal through all the test documentation for a given test level (e.g. system testing, from test conditions through test cases to test scripts) or it can be vertical through the layers of development documentation (e.g. from requirements to components).

Now, the question may arise is that Why is traceability important? So, let's have a look on the following examples:

- The requirements for a given function or feature have changed. Some of the fields now have different ranges that can be entered. Which tests were looking at those boundaries? They now need to be changed. How many tests will actually be affected by this change in the requirements? These questions can be answered easily if the requirements can easily be traced to the tests.

- A set of tests that has run OK in the past has now started creating serious problems. What functionality do these tests actually exercise? Traceability between the tests and the requirement being tested enables the functions or features affected to be identified more easily.

Before delivering a new release, we want to know whether or not we have tested all of the specified requirements in the requirements specification. We have the list of the tests that have passed – was every requirement tested?