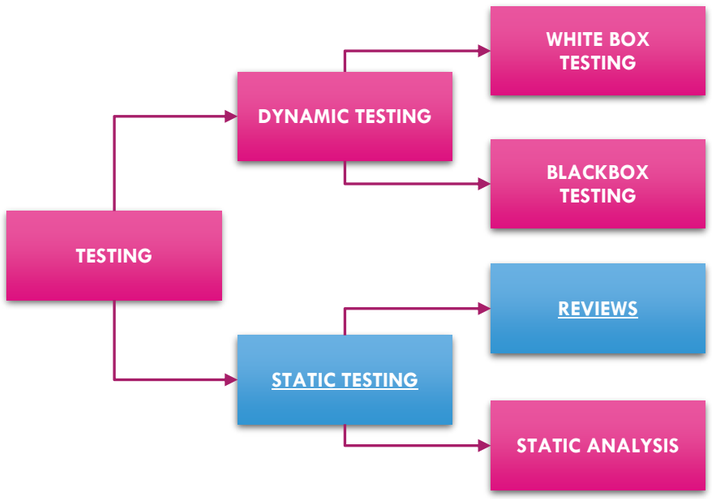
3. Static Testing



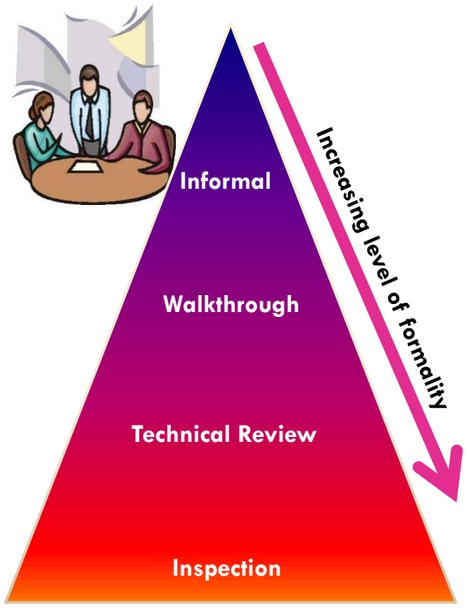
A. Benefits of static testing

Static testing (as opposed to dynamics testing) is a key principle of early testing. There less chance of defects, reduced time & cost, improved communications It is also an important part of security testing.
Testers need to have access to documentation as soon as possible.

B. Defects found are easier to fix using static testing:

Requirement defects (e.g., inconsistencies, ambiguities, omissions, inaccuracies and redundancies) NOTE: Therefore, allows early validation of user requirements.
Design defects (e.g., inefficient algorithms or database structures)
Coding defects (e.g. unreachable code, duplicate code)
Deviations from standards (e.g., lack of adherence to coding standards)
Incorrect interface specifications (e.g., different units of measurement used by the calling system than by the called system)
Security vulnerabilities (e.g., susceptibility to buffer overflows)
Gaps or inaccuracies in test basis traceability or coverage (e.g., missing tests for an acceptance criterion)
Maintainability defects (improper modularization, poor reusability of components, code that is difficult to analyse and modify without introducing new defects)

C. Review types



Informal – No formal process. Not usually documented. Inexpensive way to achieve limited benefits. May be implemented as Pair Programming. Checklists are optional
Walkthrough – Led by Author of document. Prior preparation by reviewers. Can vary widely in practice from formal to informal. Main purpose to enable learning. Scribe is mandatory. Checklists are optional
Technical Review – Gain consensus. Prior preparation by individuals. Scribe is mandatory (ideally not Author). Checklist are optional. Usually performed as a peer review. Led by facilitator/moderator. Checklists are optional. Can also vary widely from informal to formal.
Inspection – Based on rules, checklist, entry and exit criteria. Prior preparation is essentials. Main purpose to find defects but Secondary may be process improvement. Led by trained facilitator/moderator. Metric are collected and used for improvement of the software development process

D. Review meetings

(occur in tech/peer reviews and inspections)
Planning: Define review criteria, define exit criteria, checking entry criteria, define roles, select personnel
Initiate Review: Distributing documents, explain objectives, process, documents to
Individual review (AKA preparation): Actual review. Issues are recorded
Issue communication and analysis: Note defects, evaluate issues, make decisions on this information
Fixing and Reporting: Fix defects & record updated status of defects. Gather metrics, check if defects have been dealt with

E. Review Attendees

Review leader – Responsible for the review, decides who's going to be invited and where the meeting will take place
Facilitator/Moderator – mediates reviews, ensures effective running of review.
Management – Plans and decides on the execution of reviews. Determines whether review process objectives have been met. Assign staff and budget. Monitors costs and controls decisions.
Scribe – Collates potential defects found during the individual review activity. Records new potential defects, open points, and decisions from the review meeting (when held)
Author – of the document being reviewed. Helps understanding of the defects
Reviewers – May be SMEs, Stakeholders, etc. Identify potential defects in the work under review. May represent different perspectives (e.g. tester, programmer, BA etc)

F. Applying Review Techniques:

Ad hoc - Reviewers given little/no guidance. Little preparation is required. Go through work product sequentially to identify defects. Dependant on review's skills and may leads to duplicate issues being reported by different reviewers.

Checklist-based – Systematic coverage of typical defects as effects detected based on checklist. Checklist consists of set of questions based on potential defects, which may be derived from experience. Checklist is sent out at review initiation by facilitator.

Scenarios and dry runs - Reviewers are provided with structured guidelines on how to read through the work product. Supports reviewers in performing "dry runs" on the work product based on expected usage of the work product (if the work product is documented in a suitable format such as use cases). These scenarios provide reviewers with better guidelines on how to identify specific defect types than simple checklist entries. reviewers should not be constrained to the documented scenarios.

Role-Based – Reviewers evaluate the work product from the perspective of individual stakeholder roles. Typical roles include specific end user types (experienced, inexperienced, senior, child, etc.), and specific roles in the organization (user administrator, system administrator, performance tester, etc.).

Perspective-based – Similar to role-based. Requires the reviewers to attempt to use the work product under review to generate the product they would derive from it. E.g. a tester would attempt to generate draft acceptance tests if performing a perspective-based reading on a requirements specification to see if all the necessary information was included. Further, in perspective-based reading, checklists are expected to be used. *Most effective Review Technique.*

G. Success Factors for Reviews

Organisational SFs – Participants have time to prep, clear objectives, checklists updated, scheduled on time, management support, large docs reviewed in small pieces so author is provided with early and frequent feedback on defects (quality control exercised).

People-related SFs – Right people present, adequate training provided (especially for more formal reviews), tester valued as reviewers, culture of learning promoted