Chapter 3 - Static techniques

Lan Anh Tran

March 25, 2025

Contents

1 Introduction

Software testing is a critical process that ensures the reliability and quality of software systems. Among the various testing methodologies, static techniques play a fundamental role in identifying defects early in the development cycle. Unlike dynamic testing, which requires code execution, static testing involves the manual examination and automated analysis of code and documentation without running the software. This chapter explores different static techniques, their integration into the test process, and the tools that facilitate static analysis. Additionally, it delves into the review process, discussing its significance, types, and best practices to enhance software quality and maintainability.

2 Static techniques and the test process

2.1 Important terms

Dynamic testing requires the execution of software under test.
Static testing is manual examination and automated analysis of the code or documentation without execution of the software under test.
Reviews is a way of testing software products (including code) and can be performed well before dynamic test execution.
Static analysis tools analyze program code.

2.2 Static techniques

Static and dynamic testing have the same objective: identifying defects. They are complementary, which means that they are useful together. Compared to dynamic testing, static techniques find causes of failures (defects) rather than the failures themselves.

3 Static analysis by tools

Static analysis is performed without actually executing the software being examined by the tool. Static analysis tools analyze program code, as well as generated output such as HTML and XML.

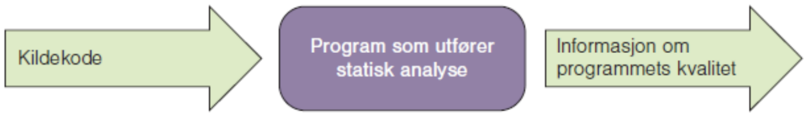
Kildekode
Program som utfører statisk analyse
Informasjon om programmets kvalitet

Figure 1: Static analysis tool

3.1 Objectives of static analysis

Find defects and increase the quality of hte software source code and software models.

Note 3.1. *Static analysis finds defects rather than failures.*

3.2 Typical defects discovered by static analysis tools

- Referencing a variable with an undefined value
- Inconsistent interface between modules and components
- Variables that are never used
- Unreachable (dead) code
- Syntax violations of code and software models
- Programming standards violations
- Security vulnerabilities

3.3 When should we use static analysis tools?

Static analysis tools can be used by both developers and designer.

Developers
Use static analysis before and during component testing and integration testing.

Designers
Use static analysis during software modeling.

3.4 Why is static analysis valuable?

- Early detection of defects prior to test execution
- Early warning about suspicious aspects of the code or design, by the calculation of metrics, such as a high complexity measure.
- Identification of defects not easily found by dynamic testing
- Detecting dependencies and inconsistencies in software models, such as links
- Improved maintainability of code and design
- Prevention of defects, if lessons are learned in development

3.5 Code structure

We have three different code structures during static analysis by tools.

Control flow structure which is the sequence in which the instructions are executed.

Data flow strucutre follows the trail of data item as it is accessed and modified by the code.

Data structure which is the organization of the data itself, independent of the program (array, list, stack, queue, tree, graph, ...)

3.6 Coding standards

Recommended that existing standards should be adopted to save a lot of effort.

- Naming conversions, e.g. class name should start with a Capital letter.
- Access conversions, e.g. public/private
- Layout specifications, e.g indents
- Set of programmign rules, e.g. always check boundaries on an array when using it
- Checking tools supports code standards

3.7 Code metrics, e.g.

- Comments frequency
- Depth of nesting
- Cyclomatic complexity/complecity metrics

3.8 Practical side

Static analysis tools may produce a large number of warning messages, which need to be well managed to allow the most effective use of the tool.

4 Review process

Reason to make reviews

- Defects deteced during reviews early in the life cycle are cheaper to remove than those detected while running tests.
- Reviews can find omissions, for example, in requirements, which are unlikey to be found in dynamic testing.

Tools (manual + tool support)
The main manual activity is to examine a work product and make comments about it.

4.1 The review process

Object of reviews
Any software work product can be reviewed, e.g.

- Requirements specifications
- Design specifications
- Code
- Test plans, test specifications, test cases, test scripts

- User guides
- Web pages

Benefits

- Early defects detections and correction
- Development productivity improvements
- Reduced development timescales
- Reduced testing cost and time
- Lifetime cost reductions
- Fewer defects
- Improved communication

Typical defects

- Deviations from standards
- Requirement defects
- Design defects
- Insufficient maintainability
- Incorrect interface specifications
- inconsistencies, ambiguities, contradictions, omissions, inaccuracies, and redundacies in requirements

These defects are easier to find in reviews than in dynamic testing

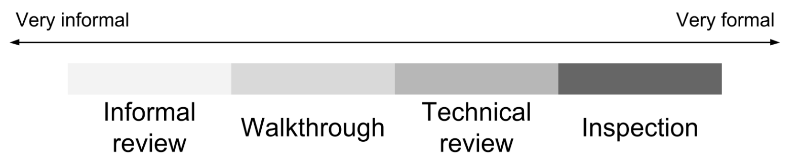

Figure 2: Different types of revies vary from: very informal to very formal

4.2 Review Process - Background

The formality of a review process is related to factors like:

- Risk
- Size of the project
- The maturity of the development process
- Any legal or regulatory requirements
- The need for an audit trail

The way a review is carried out depends on the agreed objective of the review:

- Find defects and omissions
- Gain understanding
- Discussion and decision by consensus

4.3 Activiteis of a formal review

Phases of a formal review

1. Planning
2. Initiate review - kickoff
3. Individual review - individual preparation
4. Issue communication and analysis - review meeting
5. Fixing and reporting - rework and follow-up

Planning focuses on selecting the personnel, allocate roles, define the entry and exit criteria for more formal review types (e.g. inspection), then selecting which parts of documents to look at.

Initiate review focuses on disitributing documents, explainting the objectives of the review and the review process, explaining the documents to the participants and checking and discuss entry/exit criteria.

Individual review/preparation works done by each of the participants on their own before the review meeting, noting potential defects, questions and comments. Each participants proposes the severity of the defects. Severity classes: critical, major or minor.

Issue communication and analysis usually has a review meeting that include logging and discussion, with documented results or minutes. The meeting participants may simply note defects, make recommendations for handling the defects, or make decision about the defects. Decisions based on the exit criteria. And examining, evaluate and recording.

Fixing and reporting where rework and follow-up is about fixing defects found, typically done by the author. Checking that defects have been addressed. And gather metrics, e.g number of defects found, time spent checking per page, total review effort etc.

4.4 Roles and responsibilites

Note 4.1. *One person may take one or more roles!*

The author
The writer or person with chief responsibility for the documents to be reviewed and the rework to be done.

The management
Decides on the execution of reviews. Assigning resources: staff, budget and time. determines if the review objectives has been met.

The review leader
Taking the overall responsibilites for the review. Deciding who will be involved. Works closely with both the management and the facilitator (moderator).

The facilitator or moderator
Leads the review of the document(s). Planning the review. Running the meetings. And follow-up after the meeting. And if necessary, the faciliator og moderator may mediate between the various points of view and is often the person upon whom the success of the review rests.

The reviewers
Individuals with specific technical or business background. Identify and describe the findings in the product under review.

Note 4.2. *Reviewers should be chosen to represent different perspectives and roles in the review process.*

Note 4.3. *Reviewers should take part in the review meeting.*

Scribe (or recorder)
Documents all the issues, problems and open points that wew identified during the meeting.

4.5 Types of reviews

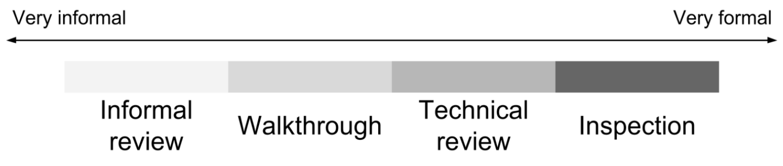

Figure 3: Types of reviews

1. Informal review
Purpose: Inexpensive way to get some benefit.
Form: Pair reviews; e.g. pair programming or a technical lead reviewing designs and code

Note 4.4. *No formal process and optinally may be documented*

2. Walktrough
Purpose: Learing, gaining understanding, defect finding and feedback.
Form: Meeting led by author, may vary in practice from quite informal to very formal. And stakeholders may participate.

3. Technical review
Purpose: Discuss, make decisions, evaluate alternatives, find defects, solve technical problems, check conformance to specifications and standards.
Form: May vary from very formal to informal peer review without management participation. Techinal review is ideally led by trained facilitator or moderator. Documented, defined defect-detection process; includes peers and technical experts. Pre-meeting preparation. Optionally the use of checklists, review report, list of findings and management.

4. Inspection
Purpose: Find defects
Form:

- Usually peer examination led by trained faciliator or moderator (not the author)
- Formal process based on rules and checklists with entry and exit criteria
- Pre-meeting preparation
- Defined roles
- Includes metrics
- Inspection report, list of findings

4.6 Review techniques

- Ad hoc reviewing
- Checklist-based reviewing
- Scenario-based reviewing and dry runds
- Role-based reviewing
- Perspective-based reviewing

4.7 Success factors for reviews

We have to different success factors: organizational and people-related.

Organizational success facotors

- Have a clear objective
- Pick the right review type and technique
- Review material need to be keept up to date
- Limit the scope of review
- Enough time!
- Management support is critical

People-related succsess factors

- Pick the right reviewers
- Use testers
- Each reviewers does their review work well
- Limit the scope of the review and pick things that really count
- Defects found should be welcomed
- Review meeting are well managed.
- Trust is critcal
- How you communicate is important
- Follow the rules, but keep it simple
- Train participants
- Continuously improve process and tools
- Just do it!

Approach
Defects found are welcome and expressed objectively. Apply suitable review techniques for the type and level of software products. Use checklists or roles if appropriate to increase effectiveness of defect identification. Management supports a good review process (e.g. by incorporating adequate time for review activites.)

Training and learing
Training is given in review techniques, especially the more formal techniques, such as inspections. There is an emphasis on learing and process improvement.

5 Conclusion

Static techniques provide an essential layer of quality assurance by detecting defects before code execution, reducing the cost and effort required for later-stage debugging. Through static analysis tools and structured review processes, software teams can identify issues related to coding standards, security vulnerabilities, and maintainability concerns. By incorporating these techniques effectively, organizations can enhance software reliability, improve collaboration, and streamline development workflows. Ultimately, static testing serves as a complementary approach to dynamic testing, reinforcing the overall goal of delivering robust and high-quality software products.