Inspection

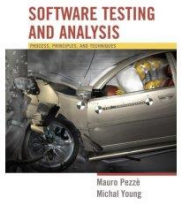
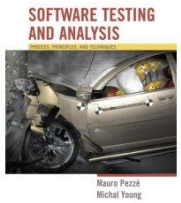


Photo credit jurvetson on Flickr.com;
creative commons attribution license

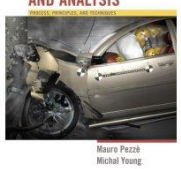
Learning objectives

- Understand how and why manual inspection fits in a modern quality process
- Introduce some common ways of organizing inspection, and basic principles on which they are based
  - Including contemporary activities such as pair programming
- Appreciate the role of checklists as a form of process feedback



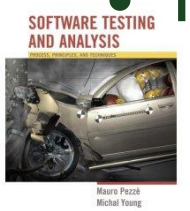
What is Inspection?

- A whole family of systematic review techniques
  - ranging from classical “Fagan” inspections to pair-programming, and many others
- Common threads
  - Systematic, detailed review
    - More than just “look this over”
  - Peer review
    - Not just the author; not just the boss
  - Inherently manual
    - Though some automated support is possible
- Widely practiced



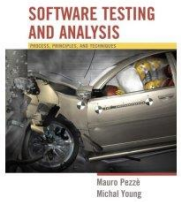
Why inspection?

- Applicable to any artifact
  - Not just code. Also design documents, specifications, test plans, ...
- Finds defects that testing misses
  - Complementary: Different faults
- Social and educational benefits
  - Encourages quality work (to avoid embarrassment)
  - Spreads quality norms
  - Educates new team or organization members
- Required by some standards (ISO, CMM)



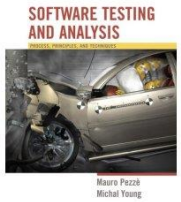
Inspection: Features (2)

- Downsides:
  - Expensive
  - Not incremental (repeating costs as the first one)
- An inspection process is characterized by:
  - Roles (who are the inspectors?)
  - Process (how do the inspectors organize their work?)
  - Reading techniques (how are artifacts examined?)



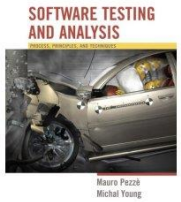
The inspection team

- Inspection is not full-time!
  - Productivity drops quickly during a session
  - Inspectors are usually borrowed from other roles
  - Tradeoff between cost, background and perspective
    - More expert = more valuable, but more costly
    - More perspectives = bigger teams
- Different levels of inspections
  - Junior engineers can do simpler checks while self-training on standards and practices
  - Senior engineers can be paired with junior engineers when checking more complex properties
  - Larger groups when special expertise is required



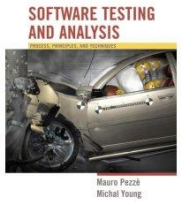
Classical inspection teams

- Classic ("Fagan") inspection postulates from 4 to 6 inspectors
  - A senior engineer moderates the inspection
  - Software and testing engineers, both junior and senior, read and comment the artifacts
  - The developer is usually present to provide detailed knowledge that cannot be easily acquired



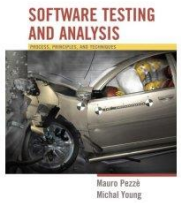
Involvement of developers

- Developers must be motivated in collaborating constructively in inspection
  - The relationship between inspectors and developers is delicate
  - Inspectors are usually borrowed from other project
  - Typical roles:
    - Software engineers
    - Test engineers
    - Project and quality managers
    - Technical writers
    - Software analysts and architects
  - Reward mechanisms can have adverse effects



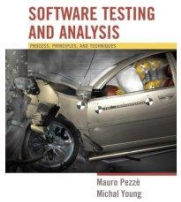
The inspection process (1)

- Scheduling is essential to optimize costs
  - Too early = detected faults will be likely reintroduced
  - Too late = correction of faults may become expensive
- Example tactics:
  - Avoid checking compliance to coding standards until testing, when most of the code has been produced
  - Check conformance of application semantics after testing, to reduce distraction by simpler faults



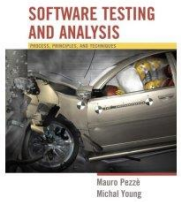
The inspection process (2)

- Usually structured in three phases:
  - Preparatory
  - Review
  - Followup



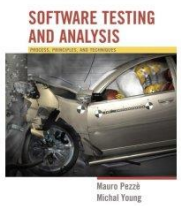
The preparatory phase

- Inspectors detect and check the readiness of the artifacts to be inspected
- Inspection roles are assigned
- Necessary support information is gathered
- Individual inspection activities are assigned
- Inspection meetings are planned



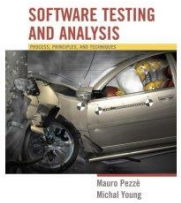
Checklists in the Review Phase

- A checklist is a set of questions that help identifying defects and in an inspected artifact and verifying its compliance to company standards
- Checklists must be structured to facilitate their use during review sessions
  - Length and complexity should match a session's planned time (not longer than 2 hrs)
  - Length/complexity ratio should be tuned on the kind of review:
    - Many simple questions for syntactic reviews
    - Few complex questions for semantic reviews



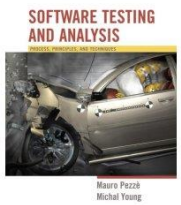
Structure of checklists

- All kinds of checklists share a similar structure:
  - Preface:
    - Type of artifact to be checked
    - Type of inspection that can be done with the checklist
    - Required level of expertise
  - List of artifact's features to be checked
    - Helps to organize the work
  - For each feature, a list of items to be checked
- The item to be checked asks whether some property holds (positive answer = compliance)
- A comment to each item can be added by the inspector



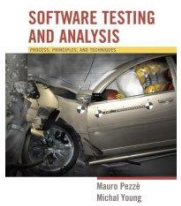
The followup phase

- The result of the review are summarized in a report
- The report is delivered to developers
- Further follow-up checks may be scheduled:
  - For simple checks, the report can be used as a checklist
  - For more complex ones, a new review may be planned



Pair programming

- A variant of program inspection associated with agile development methods
- Two programmers work side-by-side at a computer: one types, the other reviews and inspects the newly typed code
- Features:
  - No checklist
  - Fine grain
  - Tied to practices facilitating teamwork and concentration
  - Requires a constructive attitude
    - “egoless programming”
    - responsibility is up to programmers



Summary

- Systematic review is (still) a key quality activity
  - In various forms, from classic “Fagan” inspections to staged reviews to pair programming
- Key values
  - Applicable to any work product, not just code
  - Complementary to other testing and analysis techniques
  - Side benefits in education, communication, motivation