Lecture Overview

- What is a review? Why do it?
- 4 types of review method
- A generic inspection process
- Personal Software Process Reviews
- Review Success Factors and Pitfalls
- Defect Metrics
What is Formal Technical Review?

- A method involving a structured encounter in which a group of technical personnel analyzes or improves the quality of the original work product as well as the quality of the method.
Why review? We test!

- Reviews improve schedule performance

![Diagram showing the comparison between reviews and no reviews in terms of requirements (Req), design (Design), code (Code), and test (Test)].

- Reviews reduce rework.
  - Rework accounts for 44% of dev. cost!
  - Reqs (1%), Design (12%), Coding (12%), Testing (19%)

- Reviews are *pro-active* tests.
  - Find errors not possible through testing.

- Reviews are training.
  - Domain, corporate standards, group
Why review? Who benefits?

Formal technical review provides:

- Defect information to the author
- Information on work product and development to peers
- Fault likelihood data to testers
- Product status to management.
- Process status to SPI (software process improvement) group
True FTR is well-defined

- Well-defined process
  - Phases (orientation, etc.)
  - Procedures (checklists, etc.)

- Well-defined roles
  - Moderator, Reviewer, Scribe, Author, etc.

- Well-defined objectives
  - Defect removal, requirements elicitation, etc.

- Well-defined measurements
  - Forms, consistent data collection, etc.
FTR is effective quality improvement

- Reviews can find 60-100% of all defects.
- Reviews are technical, not management.
- Review data can assess/improve quality of:
  - work product
  - software development process
  - review process
- Reviews reduce total project cost, but have non-trivial cost (~15%)
- Upstream defect removal is 10-100 times cheaper.
- Reviews disseminate domain knowledge, development skills, and corporate culture.
Industry Experience with FTR

- **Aetna Insurance Company:**
  - FTR found 82% of errors, 25% cost reduction.

- **Bell-Northern Research:**
  - Inspection cost: 1 hour per defect.
  - Testing cost: 2-4 hours per defect.
  - Post-release cost: 33 hours per defect.

- **Hewlett-Packard**
  - Est. inspection savings (1993): $21,454,000

- **IBM (using Cleanroom)**
  - C system software
  - No errors from time of first compile.
Who, What, and When

- Who decides what should be reviewed?
  - Senior technical personnel, project leader

- What should be reviewed?
  - Work products with high impact upon project risks.
  - Work products directly related to quality objectives.
  - “Upstream” work products have higher impact.

- When should review be planned?
  - Specify review method and target work products in software development plan/quality plan.
The range of review practice

Development Method
- Non-Cleanroom
  - inFTR
    - Walkthrough (Yourdon89)
    - Code Reading (McConnell93)
  - FTArm (Johnson94)
  - Scrutiny (Gintell93)
  - CAIS (Mashayekhi94)
- Tool-Based
  - ICICLE (Brothers90)
  - Phased Insp. (Knight93)
- Manual
  - Code Inspection (Fagan76)
  - 2-Person Inspection (Bisant89)
  - N-Fold Inspection (Martin90)
- Cleanroom
  - Verification-based Inspection (Dyer92)

Software Review (Humphrey90)
Inspection (Gilb93)
TekInspect
Four Types of Review Method
Art Gallery and Visibility Room

- Dwayne Phillips
- SW Project
- Managers’ Handbook
- stick documents on the wall (public space)
- anyone can comment
- remove and review after for 3 days
Desktop Checklist

- A set of questions to stimulate critical appraisal of all aspects of the system.
- Questions are usually general in nature and thus applicable to many types of system.
- Author & Reviewer work through a document, checking items from the list.
- 2 people better than 1; Microsoft, Motorola: 4 better than 2.
- See [www.processimpact.com/pr_goodies.shtml](http://www.processimpact.com/pr_goodies.shtml) for sample checklists.
- Pitfall: If it’s not on the list it doesn’t get checked.
A Code Walk-Through is an informal meeting where the programmer leads the review team through his/her code and the reviewers try to identify faults.

You can do this with your own code but it is more effective if the reviewer is not the same person as the programmer (cf. egoless programming).
Formal Inspections

- Inspections are a formal, efficient and economical method of finding faults in design and code [Fagan, 76]
- Code inspection amounts to “executing the code in your head”
- Code inspections are very effective at finding faults [statistics from Pfleeger p.291]
  - detect 67% of faults [Fagan 76]
  - detect 85% of faults [Jones 77]
  - detect 93% of errors [Ackerman et al 86]
Personal SW Process Personal Reviews
What is PSP?

- PSP is the “Personal Software Process”
- PSP was invented by Watts Humphrey and is currently promoted by the Software Engineering Institute at CMU
- PSP is a technique to support individual, empirically-guided process improvement.
PSP Review Characteristics

- PSP reviews are:
  - Conducted on designs and code.
  - Have a well-defined process.
  - Always conducted prior to first compile.
  - Based upon checklists created from personal defect history.
  - Measured and evaluated for process improvements.

- PSP reviews satisfy all requirements for FTR except that they are not a *group* process.
PSP Review Process

- **Checklist Generation:**
  - Use defect history to generate checklist.

- **Design/Code review:**
  - Apply checklist at end of design/code phases to eliminate defects.

- **Evaluation/Improvement:**
  - Calculate time-savings of reviews.
  - Calculate effectiveness of checklist items.
  - Improve checklist to track defects created.
PSP Reviews vs. Generic Inspection

- Individual process
- Bottom-up driven
- No “roles”
- No group synergy
- Producer responsible for process improvement

- Group process
- Top-down and bottom up
- Moderator, Scribe, etc.
- Group synergy/learning
- Software engineering process group (SEPG) responsible for process improvement.
Inspection Success Factors and Pitfalls
Critical Success Factor: Checklists

- Checklists guide reviewers to areas prone to defects.
- Checklists may be stated as a yes/no question:
  - “Are all strings null terminated?”
- Checklists can also stimulate mental modelling:
  - “After a fork, what happens if a child exits immediately?”
- Checklists should be combined with general analysis.
  - Don’t trust checklists to be comprehensive!
- Checklists are specific to work product type and development phase.
Critical Success Factor: Effective Preparation

- Effective preparation requires both:
  - Comprehension: the nature of the entire document.
  - Analysis: inter-document consistency and adequacy.

- Focus on:
  - What is present but not adequate.
  - What is missing but should be there.
  - What unique skills and experiences can you bring to bear on the work product?

- Allocate enough time to prepare!
  - Make multiple passes over document.
  - Let it “sit overnight”.
  - Don’t prepare right before the review.
Critical Success Factor: Measurement

- The goal of Inspection is to detect and remove all defects efficiently and completely.

- We measure:
  - Time spent on each phase.
  - Number of issues of each type discovered.
  - Utility of review meeting, checklists, etc.

- Analysis over time suggests:
  - New and better checklist items.
  - Improvements to inspection process, by identifying poor quality review.
  - Improvements to software development process, by identifying poor quality work products.
  - Improvements to standards.
Critical Success Factor:
The moderator

- Indicators of effective inspection moderators:
  - Work products are inspected when ready.
  - Meeting dates are aggressive but do-able.
  - Author overviews are useful or omitted.
  - Checklists and reference materials are useful.
  - Review meeting focuses on issue detection.
  - Author does not feel threatened.
  - Rework is verified carefully.
  - Improvements to inspection and software development process are discovered.
  - Participants feel the method effectively improved quality.
  - Everyone wants to do it again!
Further reading

- *The WWW FTR Archive* [http://www2.ics.hawaii.edu/~johnson/FTR/](http://www2.ics.hawaii.edu/~johnson/FTR/)
Acknowledgements

Material on inspections is included from the review lecture notes of Philip Johnson, University of Hawaii
www.ics.hawaii.edu/~johnson
1998