Code Review

Exercise Session
Reviews

• Review: examination of (software) artifacts, in order to find mistakes and improve the quality of the artifact

• Examples for artifacts
  – Source Code
  – Design Documents (SRS Doc, API-Design Doc)
Purpose of code reviews

• Ensure that code has sufficient quality to be released / committed
  – Review formatting / documentation of code
  – Review code correctness and efficiency

• Teach (new) developers how to improve code w.r.t.
  – Code quality
  – Consistency
  – Maintainability
Why review?

• Reviewing is effective to find mistakes

• Developers (hopefully) write better code when others will look at it
  – “You don’t want to be the guy with the ugly code...“
Effectiveness of Reviews

• Inspection of design & code [Fagan 1976]
  – 67% - 82% of all faults were found by inspections
  – 25% time saved on programmer resources (despite inspections)

• [Fagan 1986]
  – 93% of all faults were found by inspections

• Cost reduction for fault detection (compared with testing)
  – [Acerman+ 1989]: 85%
  – [Fowler 1986]: 90%
  – [Bush 1990]: 25,000 US$ saved PER inspection
How to review?

Many different approaches, e.g.

• Personal review
  – Author reviews her own code → Not objective, but available to every developer

• Over-the-shoulder
  – Other developer looks „over the shoulder“ of the code author; read / discuss code together
How to review?

Many different approaches (continued)

• Walkthroughs
  – Scheduled meeting, chaired by a moderator
  – Participants prepare for the meeting in advance
  – Author presents and provides information
    • Advantage: author knows the code best
    • Disadvantage: author might feel attacked personally; tries to defend the code

• Inspections (more formal than walkthroughs)
  – Code presenter different from code author (author is present but has passive role; only answers specific questions)
  – Stronger focus on specific aspects (checklist approach)

Result of reviews: Review Report (protocol of the meeting, records all errors found)
Best practices

• Restrict to review to max. 60 – 90 min
  – Reviewers performance in finding defects drops after that time span as human concentration declines

• Review fewer than 200 - 400 LOC/hour
  – With more LOC, the ability to find defects diminishes
  – Reviewing needs time if code should be fully understood by reviewers
Best practices

• Authors should prepare the source code for review
  – Format and document code
  – Put special review- comments on sections which should be reviewed in depth

• Establish quantifiable goals for code review and capture metrics so you can improve your process
Best practices

• Use checklists for authors and reviewers
  – Helps to limit discussion and focus on important aspects

• Verify that defects are actually fixed
  – Follow up on the Review Report produced in the last meeting

• Create a good code review culture in which finding defects is viewed positively
  – Review the product, not the producer
  – Ask questions instead of making accusations
  – Stick to the review agenda
  – Raise issues, don’t resolve them
  – Limit discussions
  – Stick to technical correctness; avoid style discussions
What to review

Include sections...
• of complicated logic
• where defects severely damage essential system capability
• dealing with new environments
• designed by new or inexperienced team members

Omit sections...
• which are „straightforward“ (no complications)
• of a type already reviewed by the team in a similar past project
• that, if faulty, are not expected to effect functionality
• Reused code
• Repeated parts of code
Summary

• Very effective techniques to ensure higher quality of code (and other software artifacts)

• Low technology (“paper and pencil”)
  – Many supporting tools are available (search for them)

• Use reviews in your own projects
Sources

Slides based on material from:

- G. Engels; Slides: Software Quality Assurance – Chapter V; University of Paderborn; 2008; http://is.uni-paderborn.de/fachgebiete/fg-engels/lehre/ss08/software-quality-assurance/lecture-notes.html