Review Processes

- Quality and Reviews
  - efficacy of Q/A techniques
  - benefits of reviews
  - types of reviews
- Formal Reviews
  - the process
  - the roles
  - risks and how to manage them
- Informal Review Processes
  - differences and trade-offs
- Introduction to Project 1c

Ways to improve quality

- Try to be careful
  - follow established best practices
  - reduce number of mistakes we make
- Peer Reviews
  - get other skilled people to check our work
  - before we do further work based on it
- Testing
  - test for all the problems we can think of
  - try to find the mistakes after we make them
  - go back and fix them before we ship

Reviews

- get other sets of eyes to review our work
  - to find errors and omissions
  - to encourage developers to do better work
- review each major completed work product
  - fix requirements before we do architecture
  - fix architecture before we do the design
  - fix design before we write the code
  - understand how to test code before we write it
  - fix code before we test and ship it
- enabling us to ship better products
  - on-time, with lower development & support costs

Benefits of Reviews

- They can be better than testing
  - finds more problems than testing
  - finds problems sooner and more efficiently
- They are excellent training tools
  - process, methodology, standards, technique
- They improve information dissemination
  - reviewers learn other parts of the product
- They improve programming skills
  - as people learn from others' mistakes

Requirements Reviews

- Ensuring we are building the right thing
- user-level requirements
  - clear and well justified, widely agreed to
  - traceable and prioritized
  - relatively complete and stable
  - do we believe we can satisfy them?
- validate component-level requirements
  - reasonable, complete, consistent, testable
  - do they add up to the user-level requirements

Architectural Reviews

- Review architecture prior to design
- Is it capable of meeting requirements?
  - embraces all applicable standards
  - no performance or robustness issues
- Will it be practical to build & support?
  - all components well specified, look doable
  - reasonable use of off-the-shelf technology
  - good modularity, well abstracted interfaces
- Is there anything here we'll regret later?

Design Reviews

- Review Design prior to implementation
- Is the design reasonable?
  - it will satisfy all component requirements
  - no major concerns about it working
  - complete, correct, and relatively simple
- Is it clear how to build this component?
  - clearly achievable with existing technology
  - no significant open design questions
- Is the design testable?
  - adequately observable and controllable

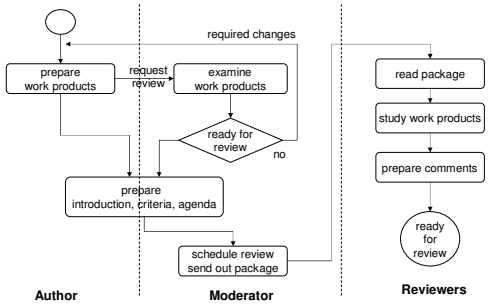
Test Plan Reviews

- Review proposed test cases
  - each clearly and adequately described
  - sufficient to cover all likely problems
  - no redundant or useless test cases
- Review proposed testing strategy
  - enables code to be tested as developed
  - clear how all tests will be implemented
  - good use of standard automation technology
- How much confidence will it give us?

Code Reviews

- Review Code prior to testing
- Does this code implement the design?
  - implements all specified functionality
  - appropriately handles all reasonable cases
- Is this code obviously correct
  - unobviousness often hides incorrectness
- Does it conform to applicable standards?
  - naming, commenting, layout conventions
  - portability, tool enabling conventions, etc.

Preparation for a Review



Author

- person who created the work product
- Preparatory tasks
  - prepare the work product for review
    - all known problems should already be addressed
  - prepare introductory & background materials
- During the review
  - author is a passive observer of the review
  - may answer questions

Review Materials - Introduction

- background
  - what project/component are we discussing
  - what do reviewers need to know about it
    - history, key problems, important decisions, etc.
  - where can they find additional information
    - requirements, designs, issue analyses
- goals of this review
  - specific work products will be reviewed
  - scope of this review (what is in/out of bounds)
  - what approval means

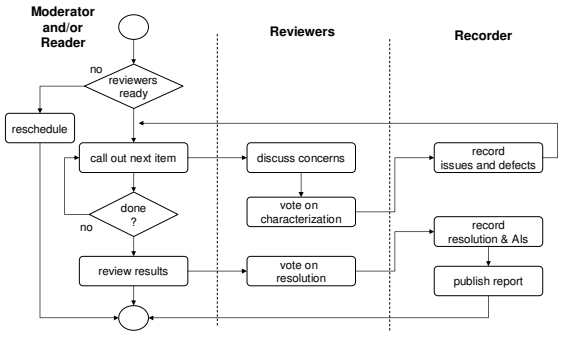
Materials - the Work Products

- the work products to be reviewed
  - specifications, designs, code, test plans, etc.
  - these must speak for themselves
  - wasteful to review them before they're ready
- a plan to structure the review
  - a table-of-contents for the work product
  - what will be reviewed, in what order
    - correct order is often critical to understanding
  - what types of issues will be covered when
  - this is the basis for the review agenda

Review Materials - Criteria

- requirements to be satisfied
  - customer, organizational, standards
- review check-lists
  - many organizations have review check-lists
    - questions to asked, problems to consider
  - they are evolved based on experience
    - e.g. at the end of McConnell's chapters
  - these can help the reviewers
    - by reminding them of things to consider
  - they can't substitute for thought/experience

The Review Process



The Review Moderator

- experienced person, other than the author
- Preparatory tasks
  - schedule the review
  - review & distribute the preparatory materials
  - prepare and distribute an agenda
- During the review
  - keep review moving per the agenda (w/o digressions, rat-holes, scope-excursions)
  - ensure all voices heard, no key points lost
  - ensure decisions & action items are recorded

Reader

- experienced person, other than the author
  - could be moderator
- during the review
  - guide the discussion through the code
  - following the prepared review materials
  - calling out each interesting element
  - asking for observations and issues

Reviewers (2-6)

- adequate technical experience
  - all reviewers must understand work products
  - others may attend for training purposes
    - to learn the technology or review process
    - but these people are not there as reviewers
- breadth of relevant expertise
  - people familiar with the problem domain, related products, or components
- take the process seriously
  - do the preparation, fully participate

Recorder

- take notes during the review
  - record all defects discovered
    - it is useful to assign a severity to each
  - record all issues raised
    - questions, suggestions, escalations, etc.
  - record decision and action items
    - accepted, major/minor revisions, further review
- publish a report of the review
  - recorder is often a Q/A or process person, observing process & collecting metrics

Potential Problems

- scope issues
  - digressions and rat-holes
  - revisiting past decisions
- productivity issues
  - materials difficult to understand
  - reviewers haven't done the preparation
  - reviewer burn-out
- ego issues
  - discussing people rather than problems
  - telling the author how to write his/her code
  - author defends his/her decisions

Less Formal Processes

- 90% of bugs are found during preparation
  - is the formal meeting a price performer?
  - how much value do moderator and scribe add?
- Structured Walk Through
  - conducted by author, perhaps w/o preparation
  - no check lists, moderator, or written report
- Code Reading
  - give code to reviewers (similar to review)
  - reviewers send feedback directly to author
  - no meeting, moderator, or written report
- Pair Programming
  - code reviewed by partner, as it is written

How do they work?

- they can work well with a good author
- advantages
  - fewer people, less overhead, faster, cheaper
  - has potential to find most of the problems
- disadvantages
  - no moderator may mean poor agenda control
  - no formal discussion may mean lost input
  - no scribe may mean lost issues and concerns
  - less opportunity for emergent insights
  - less opportunity for training and learning

Project 1c – Architectural Review

- group project, due Tue 11/13/07 (plan due 10/29)
- developing team
  - package project 1B for review
  - deliver it to a reviewing team
  - revise architecture to address all raised issues
- reviewing team
  - (individually) study supplied package, prepare notes
  - conduct a formal architectural review
  - write up report and recommendations
- write up a project post-mortem

For Next Lecture

- McConnell, chapter 8
  - excellent overview of defensive programming
- Kampe: Brief Glossary of Availability Terminology
  - brief introduction to key concepts
- Kampe: What every programmer needs to know about Software High Availability
  - overview of error management methodology

Supplementary Slides

Review Scope

- review must be kept at designated level
  - requirements … don't design the system
  - architectural … don't design the components
  - design … don't over-constrain implementor
- avoid descending to lower level issues
  - may be needed to illustrate potential issues
  - suggestions can be made outside the review
- avoid re-specifying/designing the system
  - escalations can be included in the report

Too much ego involvement

- author is too defensive to get input
  - hence author can't moderate/review/record
  - ensure senior engineers set a good example
  - exclude management from review meetings
- reviewers focusing on the wrong things
  - different approaches, style, personality
  - clearly defined review scope
  - written standards and check-lists
  - moderator must manage the egos & scope