

Software Reviews

17-654/17-754

Analysis of Software Artifacts

Jonathan Aldrich



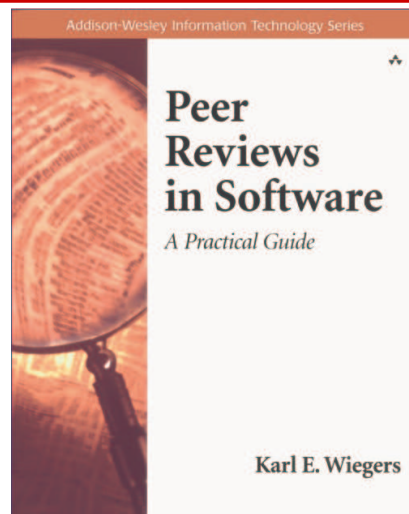
Analysis of Software Artifacts -
Spring 2006

Resources



My primary source:

- Peer Reviews in Software: A Practical Guide. Karl E. Wieggers.
 - Well-written, gives choices and rationale
 - Many other sources available



Analysis of Software Artifacts -
Spring 2006

15

Benefits of Software Reviews



- Get another perspective
 - Finding defects can be easier for someone who hasn't seen the artifact before and doesn't have preconceived ideas about its correctness
- Transfer of knowledge
 - About the software artifact, and about defect detection
- Find errors early
 - Can dramatically reduce cost of fixing them
- Reduce rework and testing effort
 - Can reduce overall development effort

Reviews vs. Testing



- Can evaluate properties testing can't
 - Maintainability, evolvability, reusability
 - Other properties tough to test
 - Scalability, efficiency
 - Security, integrity
 - Robustness, reliability, exception handling
- Can evaluate artifacts earlier
 - Requirements, design documents
 - Code before test harness is built, or when a fault prevents running a test

Statistics



- Raytheon
 - Reduced "rework" from 41% of cost to 20% of cost
 - Reduced effort to fix integration problems by 80%
- Paulk et al.: cost to fix a defect in space shuttle software
 - \$1 if found in inspection
 - \$13 during system test
 - \$92 after delivery
- IBM
 - 1 hour of inspection saved 20 hours of testing
 - Saved 82 hours of rework if defects in released product
- IBM Santa Teresa Lab
 - 3.5 hours to find bug with inspection, 15-25 through testing
- C. Jones
 - Design/code inspections remove 50-70% of defects
 - Testing removes 35%
- R. Grady, efficiency data from HP
 - System use 0.21 defects/hour
 - Black box 0.282 defects/hour
 - White box 0.322 defects/hour
 - Reading/inspect. 1.057 defects/hour
- Your mileage may vary
 - Studies give different answers
 - These results show what is possible

Inspections / Formal Technical Reviews



- Advance preparation by participants
 - Typically based on checklists
- Formal meeting to discuss artifact
 - Led by moderator, not author
 - Documented process followed
- Formal follow-up process
 - Written deliverable from review
 - Appraise product

Walkthroughs



- No advance preparation
- Author leads discussion in meeting
- No formal follow-up
- Low cost, valuable for education

Other Forms of Review



- Passaround
 - Just the preparation part of an inspection
- Peer deskcheck
 - Examination by a single reviewer
- Ad-hoc
 - Informal feedback from a team member
- Feel free to find your own variant
 - But understand the tradeoffs between techniques
 - Formal technical reviews will find more bugs, but also cost more
 - Ford: 50% more bugs with formal process

Review Roles: Moderator



- Organizes review
 - Keeps discussion on track
 - Ensures follow-up happens
- Key characteristics
 - Good facilitator
 - Knowledgeable
 - Impartial and respected
 - Can hold participants accountable and correct inappropriate behavior

Review Roles: Reader



- Presents material (different from author)
 - Provides point of comparison for author and other team members
 - Differences in interpretation provoke discussion
 - Reveals ambiguities vs. if author were to present
 - Wieggers, p. 48
- Alternative
 - Get comments section by section
 - Faster, but does not capture differing perspectives as effectively

Review Roles: Recorder



- Writes down issues

Review Roles: Author



- Not moderator or reader
 - Hard for author to be objective when presenting work or moderating discussion
 - Other inspectors can raise issues more comfortably
- Not recorder
 - Temptation to not write down issues the author disagrees with
- Significant benefits to attending
 - Gain insight from others' perspectives
 - Can answer questions
 - Can contribute to discussion based on knowledge of artifact; others' discussion may stimulate ideas
 - Only downside: meeting is more confrontational

Process: Planning



- Determine objectives
- Choose moderator
- Identify inspectors
 - Good to involve people with connection to artifact
 - e.g. depends on, interfaces with
- Schedule meeting(s)
 - General guideline: 150-200 SLOC/hour, or 3-4 pages/hour
- Prepare and distribute inspection package
 - Deliverable, supporting docs, checklists
 - Cross-reference specs, standards

Process: Overview Meeting



- Informal meeting
- Goal: go over features, assumptions, background, context
- Optional stage
 - May be able to use paper overview or shared context

Process: Preparation



- Inspectors examine deliverable
 - Defects: cause an error in the product
 - Non-defects: improvements, clarification, style, questions
 - May want to list typos/spelling/format/style separately and not discuss during the meeting
 - Conformance to standards & spec
 - Often use checklist
- General guideline
 - prep time ~ meeting time

Process: Meeting



- Reader describes one piece at a time
 - Inspectors respond: defects, questions, suggestions
- Recorder writes down each defect, suggestion, issue
 - This is the primary deliverable
- Moderator
 - Avoid problem solving, inappropriate behavior, lack of participation
- Appraisal of product
 - Accepted (minor changes, no follow up)
 - Accepted conditionally (minor changes, verification)
 - Reinspect following rework (major changes)
 - Inspection not completed
- Gather input on improving inspection process
- Moderator prepares report with appraisal and data
- Variant: reviewers make comments on electronic bulletin board
 - Cost is lower
 - Lose benefits of real meeting
 - Synergy - new bugs found (4%? 25%?)
 - Learning by participants
 - Communication about product

Process: Rework and Follow-up



- Author addresses each item
 - Ensure understands issue
 - Judge whether or not it is a defect
 - Fixes defects and makes improvements
 - Any uncorrected/unverified defects go into defect tracking system
- Deliverables
 - Corrected work product
 - Response to each issue and rationale for action
- Moderator (or verifier) meets with author
 - Check resolution of issues
 - Examine corrected deliverable
- Author checks in code

Process: Analysis



- Casual analysis
 - Analyze root causes of defects
- Make improvements to development and QA processes
 - Add issue to checklist
 - Change testing approach
 - Develop or purchase new static analysis
- Measuring effectiveness
 - % of bugs found during inspection
 - vs. found in inspection and afterwards (test, customer)
- Measuring efficiency
 - Defects per hour
 - Will decrease as your process improves

What to Inspect



- Requirements, design documents
 - hard to validate in other ways
 - especially important to get right
 - cheaper to fix earlier on in process
 - many different views are helpful
- Critical or uncertain pieces of code
- Initial work in large work segment
 - catches mistakes early, when easy to fix
 - allows rest of system to be built with knowledge gained
- Samples of larger body of work

Review Guidelines



- Build reviews into your schedule
 - Otherwise viewed as intrusion
 - Recognize that reviews can accelerate schedule by reducing other V&V activities
- Keep review team small
 - General guidelines: 3-7 participants
 - 3 is minimum for formal process to work
 - Below 3 you don't get enough perspectives besides author
 - Above 7, work goes more slowly, scheduling is difficult, moderating is hard
 - Smaller groups for code, larger groups for other documents
 - Knowledge is spread around more, more stakeholders
- Find problems, but don't try to solve them
 - Typically less expensive to address 1-on-1
 - Guideline: halt solution discussion after 1 minute
- Limit meetings to 2 hours maximum
 - Attention span gets lost beyond this
- Require advance preparation
 - Provides much of the value of a (formal) review

Checklists



- **Benefits**
 - Focus on likely sources of error
 - Form quality standard that aids preparers
 - Can bring up issues specific to a product
- **Should be short**
 - Rule of 7 : can only keep about 7 things in the head at once
 - If more than that, group and do multiple passes
- **Focus**
 - Most important issues
 - Issues unlikely to be found other ways
 - Historical problems
 - Issues specific to the document
- **Start with checklist from well-known source**
 - Refine based on experience

Social Aspects of Reviews



- **Reviews are challenging**
 - Authors invest self-worth in product
 - Encourages you to avoid letting others find errors
- **For Authors**
 - Recognize value of feedback
 - Place value in making code easy to understand
 - Don't take criticism of code personally
- **For reviewers**
 - Don't show off how much better you are
 - Be sensitive to colleagues
 - Bad: "you didn't initialize this variable"
 - Good: "I didn't see where this variable was initialized"

Review Pitfalls



- Letting reviewers be in charge of quality
 - Attitude: “why fix this, the reviewers will find it“
 - Responsibility for quality is with author, not reviewers
 - reviewers help in that goal
- Insisting on perfection before review
 - Makes harder to accept suggestions for change
- Using review statistics for evaluation
 - Real world example: manager decides "finding more than 5 bugs during an inspection would count against the author"
 - Weigers '02
 - Negative effects
 - Avoid submitting for inspection
 - Submit small pieces at a time
 - Avoid pointing out defects in reviews (thus missing them)
 - Holding "pre-reviews" that waste time and skews metrics