Quality Assurance: Building Quality In

The role of “testing”*
Active reviews

What

• Narrow view:
  – Testing is executing a program and comparing actual results to expected results
• Wider view:
  – “Testing” is shorthand for a variety of activities:
    anything we can do to check for defects
  – Dynamic program testing is the most common activity when the artifact is program code
  – Also, reviews, analysis of models, automated checks; we usually need several

Why test?

• Goal: develop software meeting the stakeholders’ functional and quality requirements at the lowest cost
• Imperfect development means some costs must be devoted to detecting and correcting errors
• Fundamental question: how is that money best spent?
  – When should testing be done?
  – What kinds of testing are most cost effective?
    • For reducing development costs
    • For avoiding costs of defects in final product
  – How much testing is sufficient?

Implications for Control

• What does “control” really mean?
• Cannot get everything under control then run on autopilot
• Rather, control requires continuous feedback
  1. Define ideal
  2. Make a step
  3. Measure deviation from idea
  4. Correct direction or redefine ideal and go back to 2
Feedback in the Product Development Cycle

Errors, Detection, and Repairs

- Basic observation:
  - Cost of a defect grows quickly with time between making an error and fixing it
    - Step function as defects cross scope walls: From programmer to sub-team, from close colleagues to larger team, from module to system, from developers to independent testers and from development to production
  - “Early” errors are the most costly
    - Misunderstanding of requirements, architecture that does not support a needed change, ...

Costs: Importance of Early Defect Detection

1. The majority of software errors are introduced early in software development
2. The later that software errors are detected, the more costly they are to correct

When

- As early as possible
  - Reduce the gap between making an error and fixing it
    - Ideally to “immediately” ... which we call “prevention” or “syntactic checking”
    - E.g., error detection/correction in Eclipse, other programming environments
- Throughout development
  - People make mistakes in every activity, so every work product should be tested as soon as possible
  - But should continue: different activities better detect different kinds of errors
Choosing What

• For every work product, we ask: How can I find defects as early as possible?
  – Ex: How can I find defects in software architecture before we’ve designed all the modules? How can I find defects in my module code before it’s integrated into the system?

• Considerations
  – Every approach has limitations
  – Generally impractical to prove correctness
  – Must correlate the project’s QA goals to the methods used

Summary: Quality is Cumulative

• Are the requirements valid?
• Complete? Consistent? Implementable?
• Testable?

• Does the design satisfy requirements?
• Are all functional capabilities included?
• Are qualities addressed (performance, maintainability, usability, etc.)?

• Do the modules work together to implement all the functionality?
• Are likely changes encapsulated?
• Is every module well defined

• Implement the required functionality?
• Race conditions? Memory leaks? Buffer overflow?

QA in Your Projects

• How do you plan to establish quality?
  – Capture QA planning in assembla pages

• Reviews (describe one)
  – What will be reviewed?
  – What kinds of reviews will be conducted and by whom?
  – What are the results

• Test plans
  – What is the testing strategy?
  – How will tests be created and by whom?
    • Module tests, system tests, etc.
  – Which testing strategies will be used and why?
    • Black box, white box, coverage, etc.

Active Reviews

Overview of peer review
More effective review
Peer Reviews

- Peer Review: a process by which a software product is examined by peers of the product’s authors with the goal of finding defects
- Why do we do peer reviews?
  - Review is often the only available verification method before code exists
  - Formal peer reviews (inspections) instill some discipline in the review process
- Particularly important for distributed teams
  - Supports communication and visibility
  - Provides feedback on both quality and understanding
  - A good review shows communication is working!

Effectiveness of Peer Reviews

- Generally considered most effective manual technique for detecting defects
  - Analysis of 12,000 development projects showed defect detection rate of 60-65% for formal inspection 30% for testing
  - Bell-Northern found 1 hour code inspecting saves 2 to 4 hours code testing
  - Effect is magnified in earlier inspections (e.g., 30 times for requirements in one study)
- Means that you should be doing peer reviews, but...
  - Doesn’t mean that manual inspections cannot be improved
  - Doesn’t mean that manual inspections are the best way to check for every properties (e.g., completeness)

Peer Review Problems

- Tendency for reviews to be incomplete and shallow
- Reviewers typically swamped with information, much of it irrelevant to the review purpose
- Reviewers lack clear individual responsibility
- Effectiveness depends on reviewers to initiate actions
- Large meeting size hampers effectiveness, increases cost
- No way to cross-check unstated assumptions
Qualities of Effective Review

• Ensures adequate coverage of artifact in breadth and depth
• Reviewers review only issues on which they have expertise
• Review process is active: i.e., performing the review produces visible output
• Individual responsibilities are clear and fulfilling them is evidence of a job well done
• Review process focuses on finding specific kinds of errors
• Limit meetings to focused groups and purposes requiring common understanding or synergy

Active Reviews

Goal: Make the reviewers think hard about what they are reviewing
1) Identify several types of review each targeting a different type of error
2) Identify appropriate classes of reviewers for each type of review
3) Assign reviews to achieve coverage: each applicable type of review is applied to each part of the specification

Active Reviews (2)

4) Design review questionnaires (key difference)
   – Define questions that the review must answer by using the specification
   – Target questions to bring out key issues
   – Phrase questions to require “active” answers (not just “yes”)
5) Review consists of filling out questionnaires defining
   – Section to be reviewed
   – Properties the review should check
   – Questions the reviewer must answer
6) Review process: overview, review, meet
   – One-on-one or small, similar group
   – Focus on discussion of issues identified in review
   – Purpose of discussion is understanding of the issue, not necessarily agreement

Examples

• In practice: an active review asks a qualified reviewer to check a specific part of a work product for specific kinds of defects by answering specific questions:
  – Ask a designer to check the functional completeness by showing the calls sequences sufficient to implement a set of use cases
  – As a developer to check the data validity of a module’s specification by showing what the output would be for in-range and out-of-range values
  – Ask a technical writer to check the SRS for grammatical errors
• Can be applied to any kind of artifact from requirements to code
### Conventional vs. Active Questions

- **Goal:** Make the reviewer(s) think hard about what they are reviewing
  - Define questions that the review must answer by using the specification
  - Target questions to bring out key issues
  - Phrase questions to require "active" answers

<table>
<thead>
<tr>
<th>Conventional Design Review Questions</th>
<th>Active Design Review Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Are exceptions defined for every program?</td>
<td>For each access program in the module, list the exceptions that can occur?</td>
</tr>
<tr>
<td>Are the right exceptions defined for every program?</td>
<td>What is the range or set of legal values?</td>
</tr>
<tr>
<td>Are the data types defined?</td>
<td>For each data type, what are: an expression for a literal value of that data type; a declaration statement to declare a variable for that type; the greatest and least values in the range of that data type?</td>
</tr>
<tr>
<td>Are the programs sufficient?</td>
<td>Write a short pseudo-code program that uses the design to accomplish (some defined task).</td>
</tr>
</tbody>
</table>

### Exploiting Use Cases

- Use cases or scenarios can be effectively used in active review
- Apply requirements scenarios to verify design against requirements
  - "Show the sequence of program calls that would implement use case C"
- Conversely, can check properties ask the reviewer to construct scenarios
  - "What sequence of calls would result in an exception E?"

### Why Active Reviews Work

- Focuses reviewer’s skills and energies where they have skills and where those skills are needed
  - Questionnaire allows reviewers to concentrate on one concern at a time
  - No one wastes time on parts of the document where there is little possibility of return.
- Largest part of review process (filling out questionnaires) is conducted independently and in parallel
- Reviewers must participate actively but need not risk speaking out in large meetings
- Downside: much more work for QA staff (but can be done in parallel)

### Summary

- Need to do reviews to find defects
- Active reviews are more efficient and effective but may take more effort
Questions?