

CS 350, slide set 9

M. Overstreet
Old Dominion University
Spring 2005

Reading

- Appendix C: Software Inspections

Outline

- What are inspections?
- What makes inspections effective?
- Inspection methods
- Inspection data
- The inspection report: from INS

What are inspections?

- Two or more programmers review someone else's work to identify:
 - defects
 - problems
- The purpose is NOT to fix, just to identify.
 - Fixing too often becomes a distraction from inspecting.
- Can be used for almost any items produced in software development
 - Requirements, test designs, HLD, DLD, test data, user manuals, ...

Inspection phases

- Briefing
 - Often developer of item brings inspectors up to speed
- Review
 - Must be done before meeting, so inspectors are prepared
- Meeting
 - The formal review process
- Repair
 - Done after meeting
- More effective if done with a script

Why are inspections effective? - 1

- Reviewing decisions so you can be ready to explain them helps you improve your decisions
- Figuring out how to explain your work to someone else often helps you discover problems in it.

Effective? - 2

- Use collective knowledge
 - The group often has a broader experience base than the developer alone; some may know what the common problems are
 - Helps if inspectors can look at product from points of view:
 - Use, tester, coder, designer, ...
- Can give opportunity to look at entire item at once
 - Testing (the alternative) tends to look only at details – one test case at a time

Other observations

- Testing is still needed even when inspections are conducted effectively
- Only inspect reviewed products
 - Respect time of other people
 - Trivial errors can distract from serious, perhaps subtle, flaws
 - Saves embarrassment

Checklists

- For this kind of detailed activities, most of us work better from checklists.
- Checklists should be developed over time
 - Most of use tend to make the same kinds of errors repeatedly
 - Should be personal checklists

Inspection rates for preparation: guidelines only

- Requirements: fewer than 2 pages/hr.
- High level design: fewer than 5 pages/hr.
- Detailed design: fewer than 100 pseudocode lines/hr
- Code: fewer than 200 loc/hr
- All of these are probably optimistic
Assumes you're shooting for an A rather than a C

More rates:

- These things vary a lot;
 - Depends on complexity of product
 - Experience of developers and inspectors
- Plan to spend about 50% of design time in inspection

Inspection yield

- High inspection yield saves time
- Humphrey reports on Air Force project where with the use of inspections, testing time went from 22% of development schedule to 2.7%

Announcements

- Group assignments made
 - Sent to UNIX accounts
 - Let me know of problems, particularly contacting ALL members
- What's due from groups:
 - Group suggestion of what modules to implement
 - E-mail to cmo. Sent by Team Leader
 - Due by midnight Saturday.
 - Goals (individual, group)
 - Due Tuesday midnight
 - Weekly Reports.
 - Submitted once per week, starting this week
 - First due Monday by midnight

Testing Concepts Review - 1

- It a lot of work and can easily take more time, creativity than coding!
 - NASA build a simulator to plug the code into
- Concept (other approaches are sometimes better):
 - Each module uses inputs to produce outputs
 - The tester makes up inputs and determines the outputs a correct program should produce from them.
 - This can be very difficult!
 - The tester prepares a file of inputs and expected outputs

Testing Concepts - 2

- A test driver feeds the file inputs to a module, then compares the module's output with those in the file.
- The test driver produces a report
- Note: code to be tested can fail in lots of ways besides getting wrong answers!
 - May not compile, may not like, may loop, may abort so that no run report is produced by the test driver.
- So after you've completed testing, you have:
 1. Files of test cases, perhaps many
 2. Code for a test driver
 3. A collection of reports, one report for each test case
 - Some generated by the test driver, some written by the tester

Test Concepts - 3

- Problems:
 - How do you know what to test?
 - Requirements coverage
 - Programming experience
 - Special cases
 - Extremes
 - Critical functionality
 - How do you create expected outputs?

Consider building a testing tool

- For cp, a correct version packs data into a packet on byte boundaries
- You give cp data, then check to see if cp puts the right stuff in the right place. Checking isn't easy because of the packing.
- Consider using a version testalias.cpp to help build test data
 - For example, give it a double, then let it print the corresponding 4 shorts.
 - Sometimes the corresponding 6 shorts! Depends on byte boundaries used by the double.
 - How could this help build test data?
 - Point:
 - Does this save you time?
 - Does this improve your testing?

Traditional inspection roles

- Developer of item (code, test plan, requirements, user doc) – present to answer questions
- Presenter – present items being inspected, line by line
- Inspectors, usually at least two – the come prepared with a list of defects that they identify when presenter gets there
- Recording – takes notes, follows up
- Moderator – decides of inspectors are prepared, runs meeting, controls behavior

Typical Time script

- 1 week before inspection, developer briefs inspectors, providing them a complete set of items to be inspected
- Inspection should go no longer than 2 hours
- Moderator must stop inspection if inspectors are not prepared
- Follow-up options:
 - If simple, left to recorder to sign off that all defects have been removed
 - If not, may need to schedule a follow-up meeting

For TSP

- Follow inspection script in text (outline below)
- Use forms provided
- Use smaller group, maybe only 3 or 4 people with moderator covering role of presenter and recorder also

Estimating yield with 2 inspectors

- A: number of defects found by ins. 1
- B: number of defects found by ins. 2
- C: number of defects found by both
- Estimated defects: $A \times B / C$
- Found defects: $A + B - C$
- Est. remaining: $A \times B / C - (A + B - C)$
- Yield: $[100(A + B - C) \times C] / (A \times B)$

2 inspector report - 1

Name	Defects		Prep Time			Est. Yield
	Major	Minor	Size	Time	Rate	
George	7	3	380	85	268	58.3
Sue	5	4	380	68	335	41.7
	12	7	760	153	298	75.0

Sent to moderator; prepared before inspection

Inspection form - 2

No.	Defect Des.	Defects		Inspectors (finding maj defects)		
		Maj	Min	A	B	
1)=>	1				1
9	.->;	1			1	
12	Converter	1			1	
13	Test not ddl	1			1	
19	N = 1000	1			1	1
...	...					
Tot		9			7	5
	Unique defects			4	2	

Completed during inspection by moderator

Computations

- From previous slides:
 - A = 7 (for George)
 - B = 5 (for Sue)
 - C = 3 (found by both)
- Est. Defects: $(5 \times 7) / 3$ or 12
- Yield is Found/Est. or $9/12$
- George's yield: $7/12$
- Sue's yield: $5/12$

Inspection form - 3

Inspection summary	Product size	380	Size measure	LOC
Total defects for A: 7	Total defects for B: 5		Common	3
Total defects (AB/C) 12	Number found(A+B-C) 9		Number left	3
Meeting time: 43	Total inspection hrs 4.7		Overall rate	80.9

Mechanics - 1

- The person who has completed the product arranges the inspection
- The moderator is the quality/process manager
- Moderator assess readiness of product for the inspection
- Min group size to do this is 3: producer, moderator/inspector, inspector

Mechanics - 2

- Before meeting, moderator gets INS forms from inspections (top part)
- Moderator may defer meeting if inspectors aren't ready
 - No reason to waste everyone's time
- Moderator does a walk-through of inspection item

Mechanics - 3

- At each line, inspectors identify any defects for question related to that line
- Moderator decides if it is major or minor
- Product owner records major defects in LOGD form so they can be included in SUMP and SUMQ forms
- At the end, the moderator decides how defect corrections will be verified.

INS script - 1

Step	Activity	Description
1	Plan the inspection	<p>The developer (programmer):</p> <ul style="list-style-type: none"> ■ Arranges with the quality/process mgr to be moderator ■ Sets up and runs the meeting <p>The moderator</p> <ul style="list-style-type: none"> ■ Reviews the product to ensure it is ready to inspect (If not, has the developer fix problems before proceeding) ■ Selects the other inspection member(s)
2	Hold the ins. briefing	<p>The moderator describes the inspection process</p> <p>The producer familiarizes the team with the item</p> <p>The reviewers select viewpoints or areas for concentration (e.g., operation, recovery, installation, size, performance)</p> <p>In design inspection, reviewers also ensure that</p> <ul style="list-style-type: none"> ■ At least one reviewer will verify each segment of the design ■ At least one reviewer will use a trace table <p>The moderator sets the date for the meeting</p>

INS script - 2

Step	Activity	Description
3	Review the item	<ul style="list-style-type: none"> ■ Reviewers separately make detailed item reviews ■ They mark the defects found on item documentation ■ They record their preparation time
4	Open the ins. mtg	<p>Moderator opens the inspection meeting and</p> <ul style="list-style-type: none"> ■ If any reviewers are not prepared, reschedules meeting ■ Outlines the inspection meeting procedure
5	Conduct item walkthrough	<p>Moderator steps through the item sections, one by one, and</p> <ul style="list-style-type: none"> ■ Has the reviewers describe every defect found ■ Confirms with producer ■ Enters major defect data on INS form ■ Notes who found each major defect ■ The producer enters each major defect in LOGD

INS script - 3

Step	Activity	Description
6	Estimate remaining defects	<ul style="list-style-type: none">■ Moderator estimates the defects remaining after the inspection■ Moderator determines reviewer's yields■ Reviewers note any items to add to their review checklist
7	Conclude mtg	<p>Inspection team decides</p> <ul style="list-style-type: none">■ Whether another inspection is warranted, who should do it, and when■ How to verify the corrections <p>Moderator completes LOGD and INS forms</p>
8	Rework item, verify fixes	<p>Producer</p> <ul style="list-style-type: none">■ Makes repairs, updates documentation■ Holds needed reviews and/or reinspections■ Has the fixes verified as decided in step 7

Other estimation techniques

- If estimation of remaining defects is important, consider
- Capture/recapture method:
 - Experienced programmer inserts several "typical" errors into item to be inspected (called "salting")
 - Inspectors inspect.
 - Assume the ratio of "salted-found" to "salted-unfound" is the same as unsalted errors.
 - If 25 errors found, 5 of them salted:
 - Assume 5/25 of all errors were found.
 - So 100 errors remaining.
