Software Inspections

Peer Reviews

Objectives

- Today
  - Overview of inspections
  - Why inspections can help improve your software quality
  - Mini inspection in preparation for actual inspections
- Coming up after next week
  - Conduct actual inspections (or heuristic evaluation)
  - Same order as presentations

Presentation Order

- Subject to change

- Thursday, March 26
  - Jazon Burnell
  - Chris Ochap
- Tuesday, March 31
  - Ian Roskam
  - Dmitry Korobov
  - Michael Burnham
- Thursday, April 2
  - Julian Bertmaring
  - Shawn Rivera / Matt Rykaczewski
  - Collin Schroeder

Why inspections?

- Inspections can be applied to many different things by many different groups
- Inspections are a "Best Known Method" (BKM) for increasing quality
  - Developed by Michael Fagan at IBM, paper published 1976
  - Estimates: Inspections of design and code usually remove 50-90% of defects before testing
  - Very economical compared to testing
- Formal inspections are more productive than informal reviews

Formal Inspections

- By formalizing the process, inspections become systematic and repeatable
  - Each person in the inspection process must understand their role
  - Use of checklists focus concentration on detection of defects that have been problematic
- Metrics
  - Feedback and data collection metrics are quantifiable
  - Feed into future inspections to improve them
- Designers and developers learn to improve their work through inspection participation

More reasons to use inspections

- Inspections are measurable
- Ability to track progress
- Reduces rework and debug time
- Cannot guarantee that a deadline will be met but can give early warning of impending problems
- Information sharing with other developers, testers

Definition

- What is an inspection?
  - A formal review of a work product by peers. A standard process is followed with the purpose of detecting defects early in the development lifecycle.
- Examples of work products
  - Code, Specs, Web Pages
  - Presentations, Guides, Requirements,
  - Specifications, Documentation

When are inspections used?

- Possible anytime code or documents are complete
  - Requirements: Inspect specs, plans, schedules
  - Design: Inspect architecture, design doc
  - Implementation: Inspect technical code
  - Test: Inspect test procedure, test report

Defects

- Inspections are used to find **defects**

- A defect is a deviation from specific or expected behavior

- Something wrong
- Missing information
- Common error
- Standards violation
- Ambiguity
- Inconsistency
- Perception error
- Design error

A defect is a defect

- A defect is based on the opinion of the person doing the review
  - This means that any defect that is found **IS** a defect
  - Not open to debate
  - Not all defects are necessarily bugs
  - Many defects may not be "fixed" in the end
- No voting or consensus process on what is a defect
- How to fix a defect should be debated later, not when the defects are logged

Other Review Methods

	Presentation	Walkthrough	Inspection
What	Present idea or proposal	Technical presentation of work	Formal review by peers
Audience	Mgmt/Tech	Tech	Tech
Objective	Provide Info, Evaluate specs or plan – Give Status	Explain work, may find design or logic defect - Give context	Find defects early - Find defects

Other Defect Detection Methods

	Buddy	Testing	Inspection
What	Developers work in pairs	Formal testing	Formal review by peers
Audience	Tech	Tech	Tech
Objective	Develop, explain work, find defects	Find defects by **symptom**, usability, performance	Find defects **where they occur**

Why a formal review?

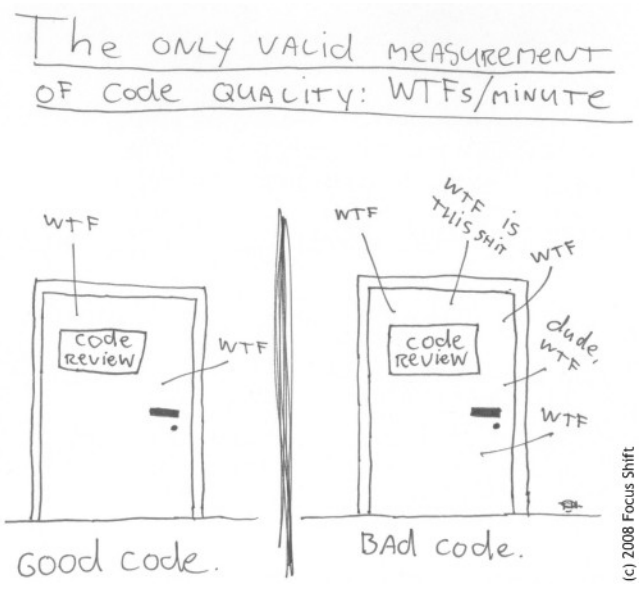
- Provides a well-defined process
  - Repeatability, measurement
  - Avoids some scenarios with less formal processes
    - "My work is perfect"
      - Point is not to criticize the author
    - "I don't have time"
      - Formal process proceeds only when all are prepared, have inspected code in advance
  - We'll actually do a mix of inspection an walkthrough

Walkthrough vs. Inspection

	Walkthrough	Inspection
Focus	**Improve product**	**Find defects**
Activities	Find defects Examine alternatives **Forum for learning** **Discussion**	**Find defects** Only defect explanation allowed **Learning through defects and inspection**
Process	Informal	**Formal**
Quality	Variable; personalities can modify outcome	Repeatable with fixed process
Time	**Preparation ad-hoc, less formal**	Preparation required, efficient use of time

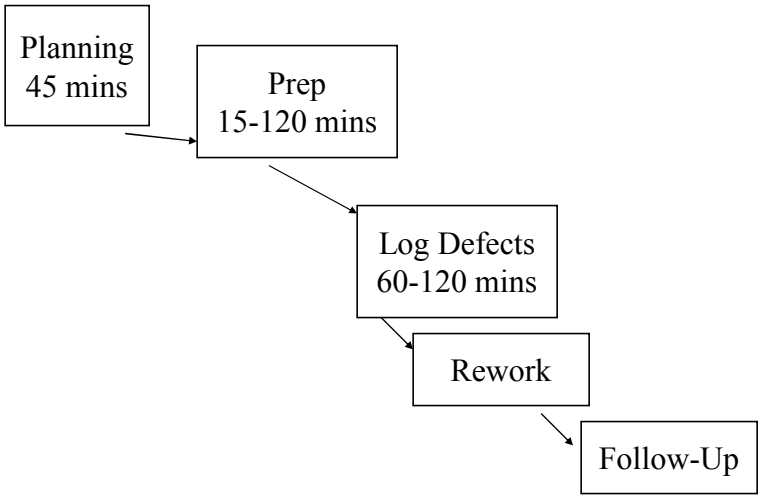
What should be inspected?

- For existing code or documentation, select
  - The most critical piece to the program's operation
  - Most used section
  - Most costly if defects were to exist
  - Most error-prone
  - Least well-known
  - Most frequently changed
- For new code or documentation
  - 20% <= inspect <= 100%

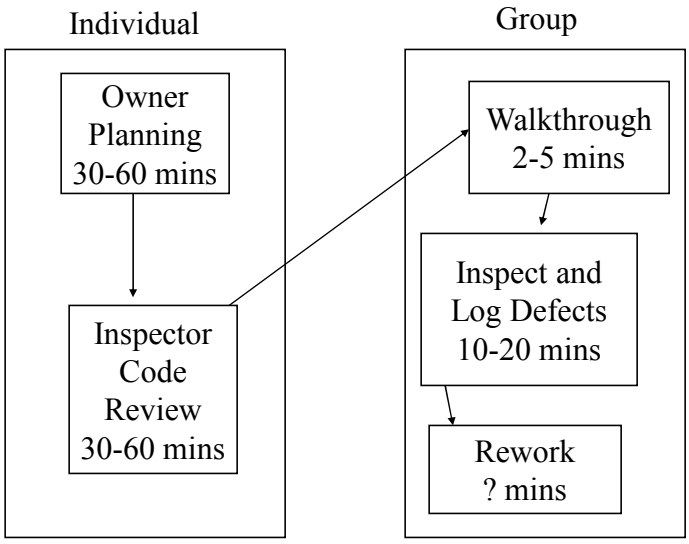


Inspection Process

Typical Inspection Process



Our Inspection Exercise



Roles

Moderator

Inspectors

Scribe

Work Owner

Owner Planning

- Owner decides what code/documents to review
  - Should include relevant requirements
  - Common-errors list
    - One should be provided by the moderator
    - Owner can include more specific common errors
    - For you: coding techniques posted on website, bug list as well
  - Copy of code listing for everyone
    - Send me code at least two days before the inspection date and I'll post it on the calendar page for everyone to get
    - Not all code, just the selected code (see previous slide on "What should be inspected?")
    - Up to owner's discretion as to what/how much, but we will stop after 25 minutes
      - Probably about 2-3 pages

Preparation

- Each inspector should have the materials to inspect in advance
  - Identify defects on their own to ensure independent thought
  - Note defects and questions
  - Complete a defect log
    - High/Medium/Low
  - Without this preparation, group review might find only 10% of defects that could otherwise be found (Fagan)
- Rules of thumb
  - 2 hours for 10 full pages of text

Preparation – Our Exercise

- Too many projects/would take too much time to perform detailed inspection for all our projects
- Compromise – Shorter inspection plus a walkthrough - enough to give you a flavor of inspections, still be useful
  - Everyone should prepare by examining code before the class and noting defects
  - Owner will provide brief walkthrough
    - May spur inspectors to note new defects
  - Scribe will log defects in real-time after walkthrough is made

Common Defects

- See handouts on web
  - C++
  - Java
- Anything we discussed in class
  - Code techniques
    - E.g. variable names, location, initialization, refactoring, defensive programming, error checking, magic numbers, loop length, etc.
  - Security
  - Usability
  - Etc.
- Similar issues apply to other languages

Walkthrough

- Prior to walkthrough
  - Owner sends me the selected code, relevant requirements or other docs
  - Code posted online
  - Inspectors have prepared by inspecting the code and noting their defects
- Process
  - Owner provides walkthrough for code
  - Inspectors search for defects
  - Round-robin where each inspector describes a defect found
  - Total of 15-25 minutes in our exercise

Walkthrough Example

- Requirement: Support authentication based upon user@host using regular expressions

```
1  /*********************************************************
2   * Returns a 1 if the user is on the ops list, and
3   * returns a 0 if the user is not on the ops list.
4   *********************************************************/
5  int Authorized(char *user)
6  {
7   FILE *f;
8
9   f=fopen(OPSPATH,"r");                          /* open authorized file */
10  while (fgets(tempstr,80,f)!=NULL)
11  {
12   tempstr[strlen(tempstr)-1]='\0';              /* annoying \r at end */
13   if (!fnmatch(tempstr,user,FNM_CASEFOLD)) { fclose(f); return(1); }
14  }
15  fclose(f);
16  return(0);
17 }
```

Open file Containing operators

Returns true if wildcards match

Defect Logging

- Performed by the scribe; leaves work owner free to concentrate on other tasks
- Moderator leads meeting and facilitates process
  - Keeps discussions to a minimum
  - Defect understanding only
  - No criticisms
  - No “rat holes”
  - Limited discussion
  - Moderator has the right to stop discussion at any time
  - May use round-robin for each inspector
- Focus is on finding defects
  - A defect is a defect

Defect Log

	Severity: H M L Q	Location	Description
1			
2			
3			
4			
5			
6			

Defect Logging

- High, Medium, Low, or Question
- Brief description should be ~7 words or less, or until the owner understands
- If possible, resolve questions: defect or not
- Also log defects found in
  - Parent document, e.g. requirements
  - Common errors list
  - Work product guidelines
- Will be up to the work owner whether or not to fix a defect

Causal Analysis Meeting

- We won't hold these, but in general they are a good idea
- Purpose – Brainstorming session on the root cause of specific defects
  - This meeting supports the continuous improvement
  - Initiate thinking and action about most common or severe defects
  - Can help prevent future defects from occurring
    - Specific action items may be achieve this goal

Rework

- Purpose: Address defects found during the logging process
- Rules
  - Performed by product owner
  - All defects must be addressed
    - Does not mean they are fixed, but that sufficient analysis/action has taken place
  - All defects found in any other documents should be recorded
  - Owner should keep work log

Follow-Up

- Purpose: Verify resolution of defects
  - Work product redistributed for review
  - Inspection team can re-inspect or assign a few inspectors to review
  - Unfixed defects are reported to the team and discussed to resolution
- We'll skip this as well, but can be useful for many projects