Lecture Notes
November 9, 2017

Black-box unit testing and completion of first iteration due tonight (includes “official” demo for first iteration)

Second Iteration Plan due November 14

(Optional) Panel of professional software engineers next Friday November 17, 11:45-12:45, Davis Auditorium - we only have the room for one hour so arrive/leave promptly

Please sit with your teams today

Code Review

Code review is a form of “static testing”, where the code is not executed

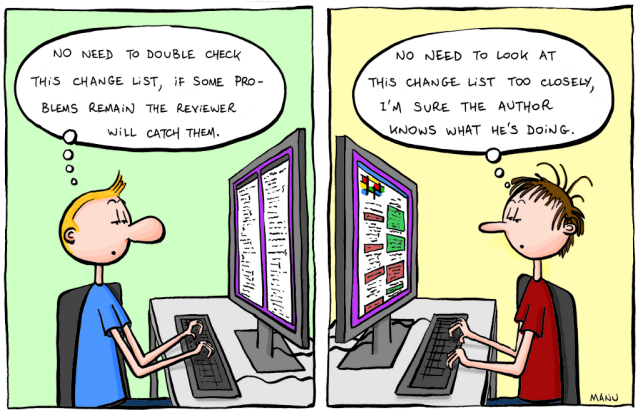
We already covered requirements and design review, before code is written, and static analyzers that automatically detect style violations, code smells, and generic bugs

In code review, one or more *humans* read the code

Informal code review happens on the fly during the pair programming, where the navigator reviews while the driver writes/edits

Many teams require developers to submit their code for independent (offline) review by a peer or designated reader prior to committing to the shared code repository

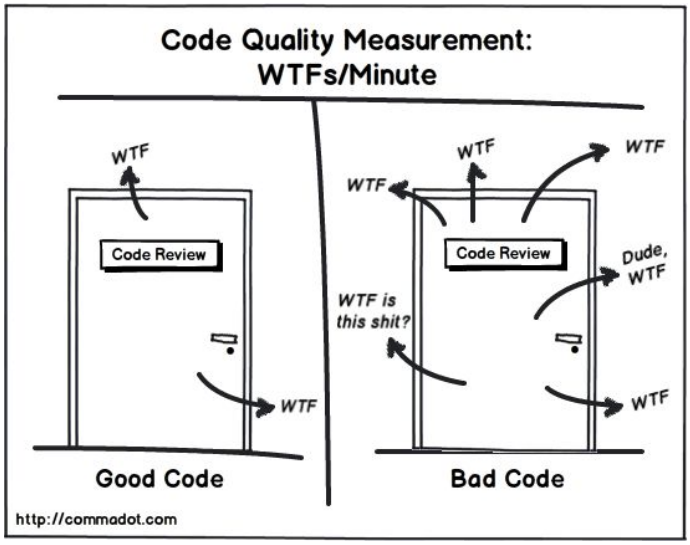
> May happen before, after or concurrently with the developer’s testing
>
> In theory, the developer and the reviewer (and the tester, if different) make independent mistakes



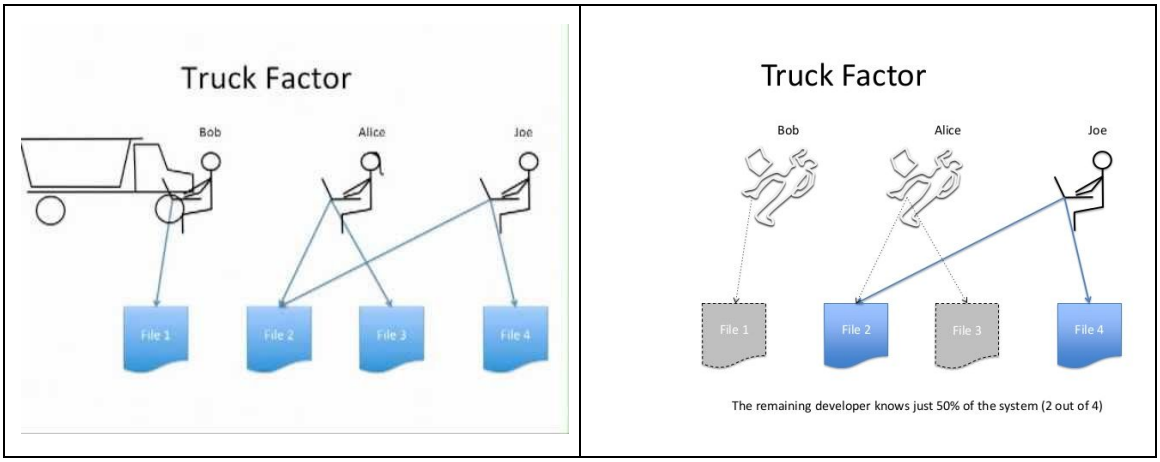
Why (statically) read/review code, why not just run (dynamic) tests?

Like automated static analysis, code review can potentially consider all inputs whereas testing can only consider some inputs (probably a very small subset of all inputs)

Code review can find problems that dynamic testing and static analyzers cannot, such as code that fulfills style guidelines and passes all its test cases, but is still unreadable (or will be unreadable by future engineers tasked with fixing bugs and adding features)



Helps cross-train developers on code written by other people (truck factor)



“Code inspection”, sometimes known as formal code review or Fagan inspection, involves a *team* in the review process not just a single reader checking the code

Predefined time limit: Fagan recommended maximum two hours

> Goal is to find problems, *not* to fix them in real time - if necessary schedule separate meeting with relevant participants

Predefined roles:

> Moderator - runs meeting, often from outside development team such as another team or a separate quality assurance group
>
> Recorder - takes notes
>
> Reader - sometimes author, sometimes intentionally not the author, reads the code aloud, walkthrough line by line, explaining along the way
>
> Everyone else is called an Inspector - attendees might include customer representatives, end-users, testers, customer/technical support, ...

Multiple stages:

> Schedule meeting and distribute specific code to be reviewed in advance - code should already compile and pass static analysis (don’t waste human time on indentation)
>
> Inspectors are supposed to prepare before the meeting by reading on their own
>
> Actual meeting
>
> Write report afterwards summarizing problems found, in addition to bug reports and change requests (can help identify problem code or common problems across code)

Problems classified as minor, moderate, severe

> Minor - author is trusted to fix on own
>
> Moderate - moderator checks
>
> Severe - need another review meeting

Class Exercise

Your code inspection: Second Iteration Development and Code Inspection due November 30 (with preliminary demo for second iteration)

Let's get started during class