Prof. Philip Koopman

Carnegie Mellon University

Electrical & Computer ENGINEERING

EDGE CASE RESEARCH

Peer Reviews

18-642 / Fall 2020

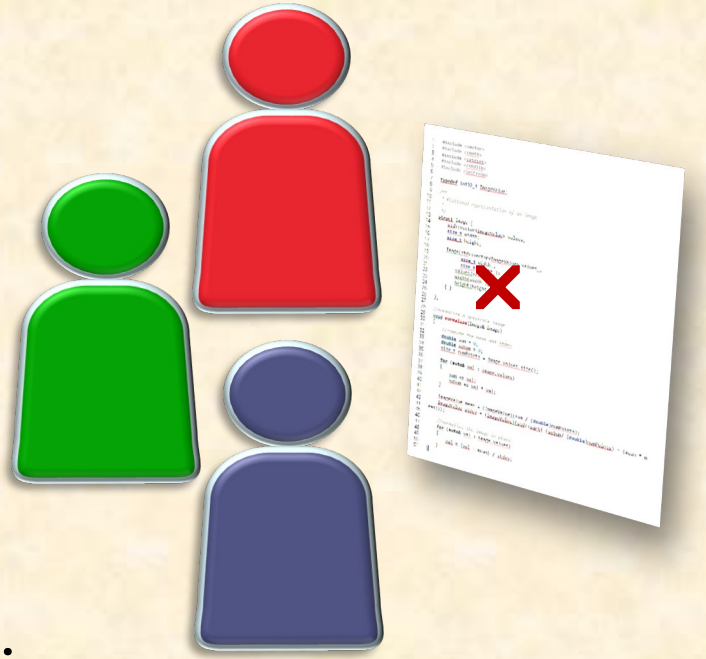
"The competent programmer is fully aware of the strictly limited size of his own skull; therefore he approaches the programming task in full humility, and among other things he avoids clever tricks like the plague.

– Edsger Dijkstra

Peer Reviews

EDGE CASE RESEARCH

- Anti-Patterns:
  - No peer reviews
  - Reviews too informal/too fast
  - Reviews find <50% of all bugs



- Fresh eyes find defects
  - Code and other document benefit from a second (and third) set of eyes
  - Peer reviews find more bugs/$ than testing
    - And, they find them earlier when bugs are cheaper to fix
  - Everything written down can benefit from a review

Most Effective Quality Practices

EDGE CASE RESEARCH

Ebert & Jones, "Embedded Software: Facts, Figures, and Future," IEEE Computer, April 2009, pp. 42-52

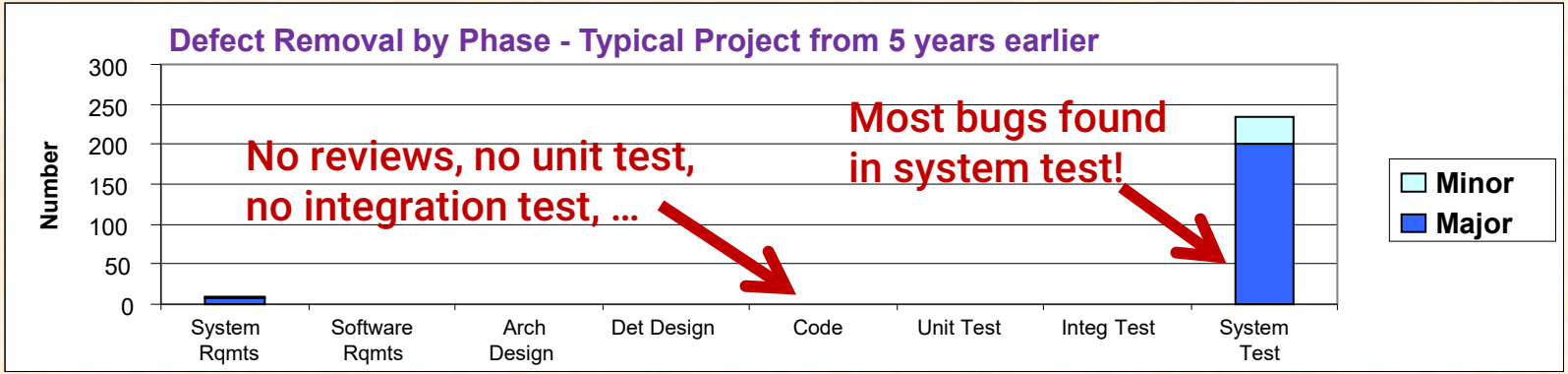
Ranked by defect removal effectiveness in percent defects detectable at that stage that are removed.

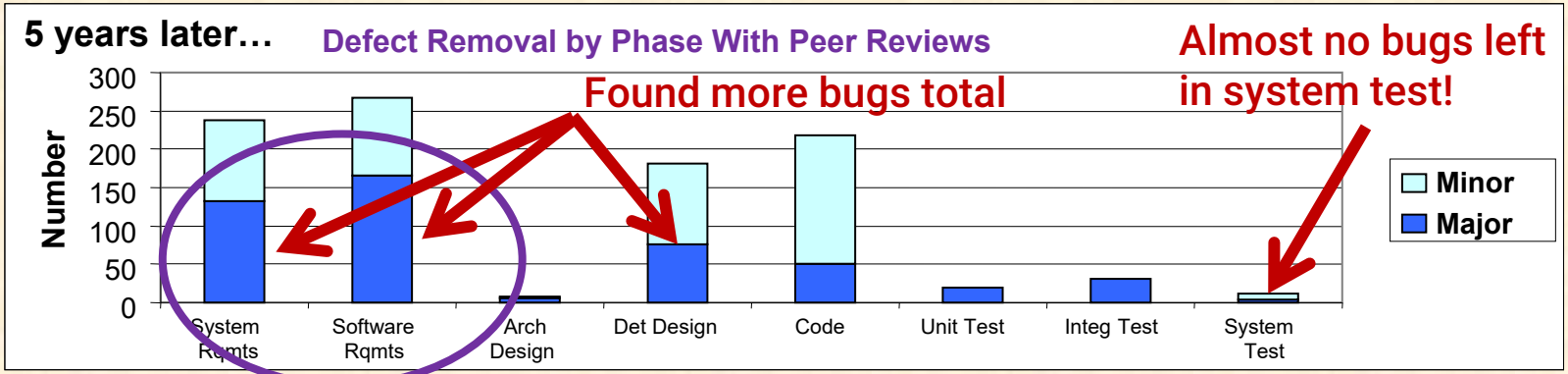
"*" means exceptionally productive technique (more than 750+ function points/month)

- * 87% static code analysis ("lint" tools, compiler warnings)
- 85% design inspection
- 85% code inspection
- 82% Quality Function Deployment (requirements analysis)
- 80% test plan inspection
- 78% test script inspection
- * 77% document review (other documents)
- 75% pair programming (informal on-the-fly review)
- 70% bug repair inspection
- * 65% usability testing
- 50% subroutine testing (unit test)
- * 45% SQA (Software Quality Assurance) review
- * 40% acceptance testing

Peer Reviews Are Effective + Efficient

EDGE CASE RESEARCH





Found many bugs up front, where fixes are cheaper

[Source: Roger G., Aug. 2005]

Gold Standard: Fagan Style Inspections

EDGE CASE RESEARCH

- Methodical, in-person review meetings
  - Pre-meeting familiarity with project
  - Producer explains item then leaves
  - Moderator keeps things moving
  - Reader (not author) summarizes as you go
  - Reviewers go over every line, using checklists (perspective-based)
  - Recorder takes written notes
  - Result: written list of defects. The Producer fixes code off-line
  - Re-inspection if the defect rate was too high
- Methodical reviews are the most cost effective
  - Important to measure bug discovery rate to ensure review quality

Rules for Successful Peer Reviews

EDGE CASE RESEARCH

- Inspect the item, not the author
  - Don't attack the author.
- Don't get defensive
  - Nobody writes perfect code. Get over it.
- Find but don't fix problems
  - Don't try to fix them; just identify them.
- Limit meetings to two hours
  - People are less productive after that point.
- Keep a reasonable pace
  - About 150 lines of code (or equivalent) per hour. Too fast and too slow are both bad.
- Avoid "religious" debates on style
  - Enforce conformance to your style guide. No debates on whether style guide is correct.
- Inspect, early, often, and as formally as you can
  - Keep records to document value (might take a while to mature).

Example Light-Weight Review Report

EDGE CASE RESEARCH

	Peer Review Template for Project X	
Date:	4/17/2011	
Artifact:	Xyzzy.cpp Functions: Foo(), Bar(), Baz()	
Reviewers:	Stella K., Joe B., Sam Q., Trish R.	
Size:	357	SLOC
Time Spent:	112	Minutes
# Issues:	3	
Outcome:	Re-Review of Bug Fixes Required	
Issue#	**Issue Description**	**Status**
1	Issue 1…..	Fixed
2	Issue 2….	Bugzilla
3	Issue 3….	Bugzilla
4	Issue 4….	Not a Bug
5		
6		
7		
8		
Status Key:	Fixed (trivial fix by author; no need to enter in defect list)	
	Bugzilla (entered into project defect system)	
	Not a Bug (false alarm)	

 # issues found is the most important item!

Free form text issue description

Just enter "fixed" if fixed within 24 hours

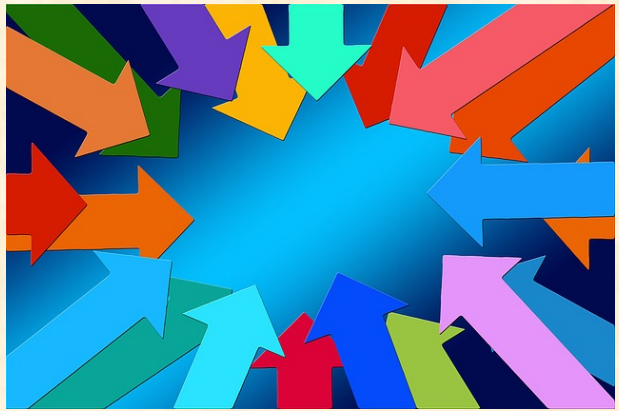
Perspective-Based Peer Reviews

EDGE CASE RESEARCH

- Perspective-based Peer Reviews are 35% more effective

[https://www.cs.umd.edu/projects/SoftEng/ESEG/papers/82.78.pdf]

- Mechanics of a Perspective-based review
  - Divide a peer review checklist into three sections
  - Assign each participant a different section of the checklist
    - OK to notice other things, but primary responsibility is that section
    - Multiple sets of eyes + perspective breadth



- Example perspectives for a review:
  - Control flow issues
  - Data handling issues
  - Style issues

Peer Review Checklist Template

EDGE CASE RESEARCH

Peer Review Checklist: Embedded C Code

- Customize as needed

Before Review:

0	____	Code compiles clean with extensive warning checks (e.g. MISRA C rules)

Reviewer #1:

1	____	Comm
2	____	Style c
3	____	Proper
4	____	No orp
5	____	Condit
6	____	Parent
7	____	All swi

Reviewer #2:

8	____	Single point
9	____	Loop entry a
10	____	Conditionals
11	____	All functions
12	____	Use const ar
13	____	Avoid use of
14	____	Strong typin
15	____	All variables

Reviewer #3:

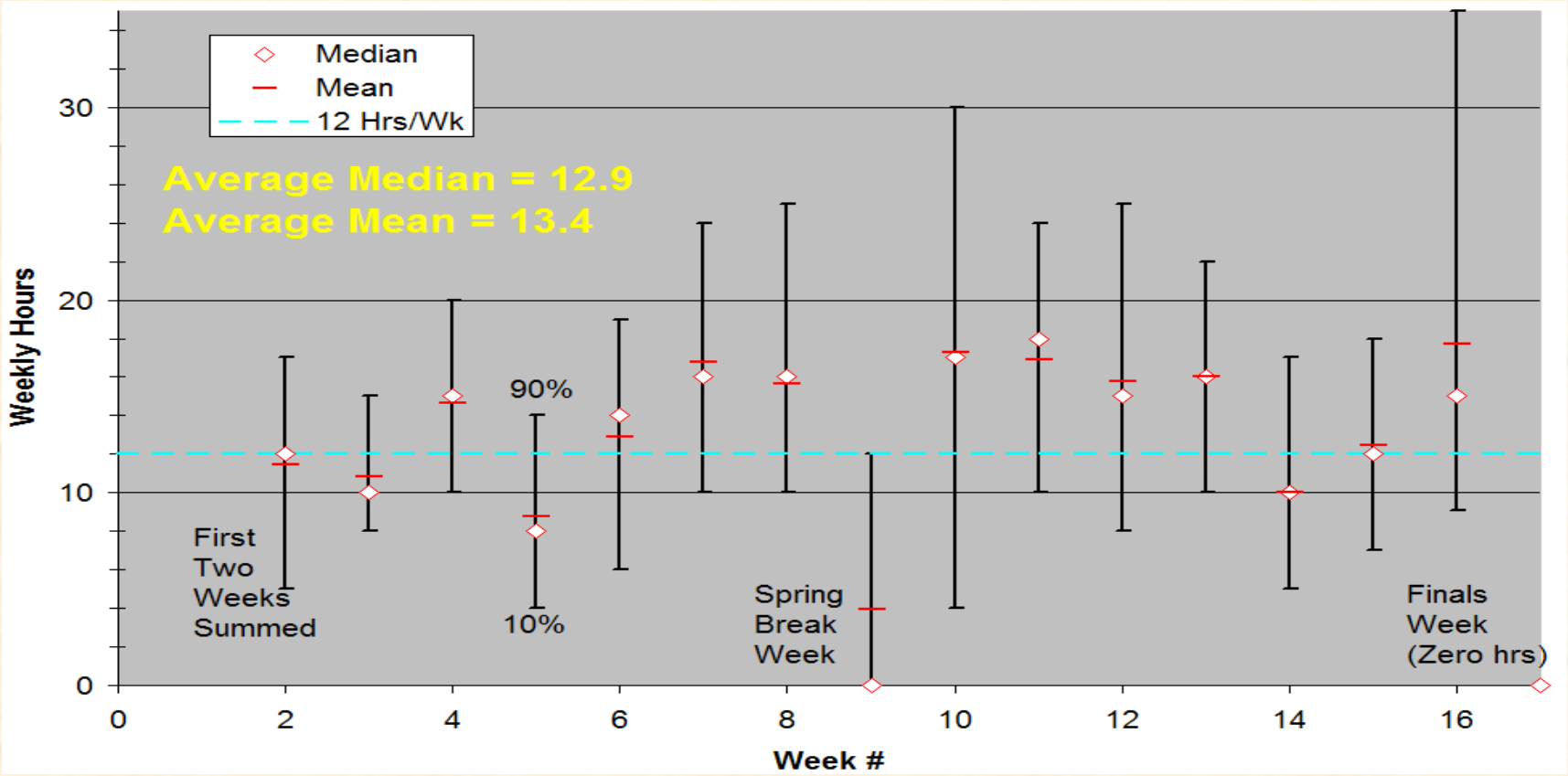
16	____	Minimum scope for all functions and variables; essentially no globals
17	____	Concurrency (locking, volatile keyword, minimize blocking time)
18	____	Input parameter checking (style, completeness)
19	____	Error handling for function returns
20	____	Handle null pointers, division by zero, null strings, boundary conditions
21	____	Floating point issues (equality, NaN, INF, roundoff); use of fixed point
22	____	Buffer overflow safety (bound checking, avoid unsafe string operations)

All Reviewers

23	____	Does the code match the detailed design (correct functionality)?
24	____	Is the code as simple, obvious, and easy to review as possible?
		For TWO Reviewers assign items: Reviewer#1: 1-11; 23-24 Reviewer#2: 12-24

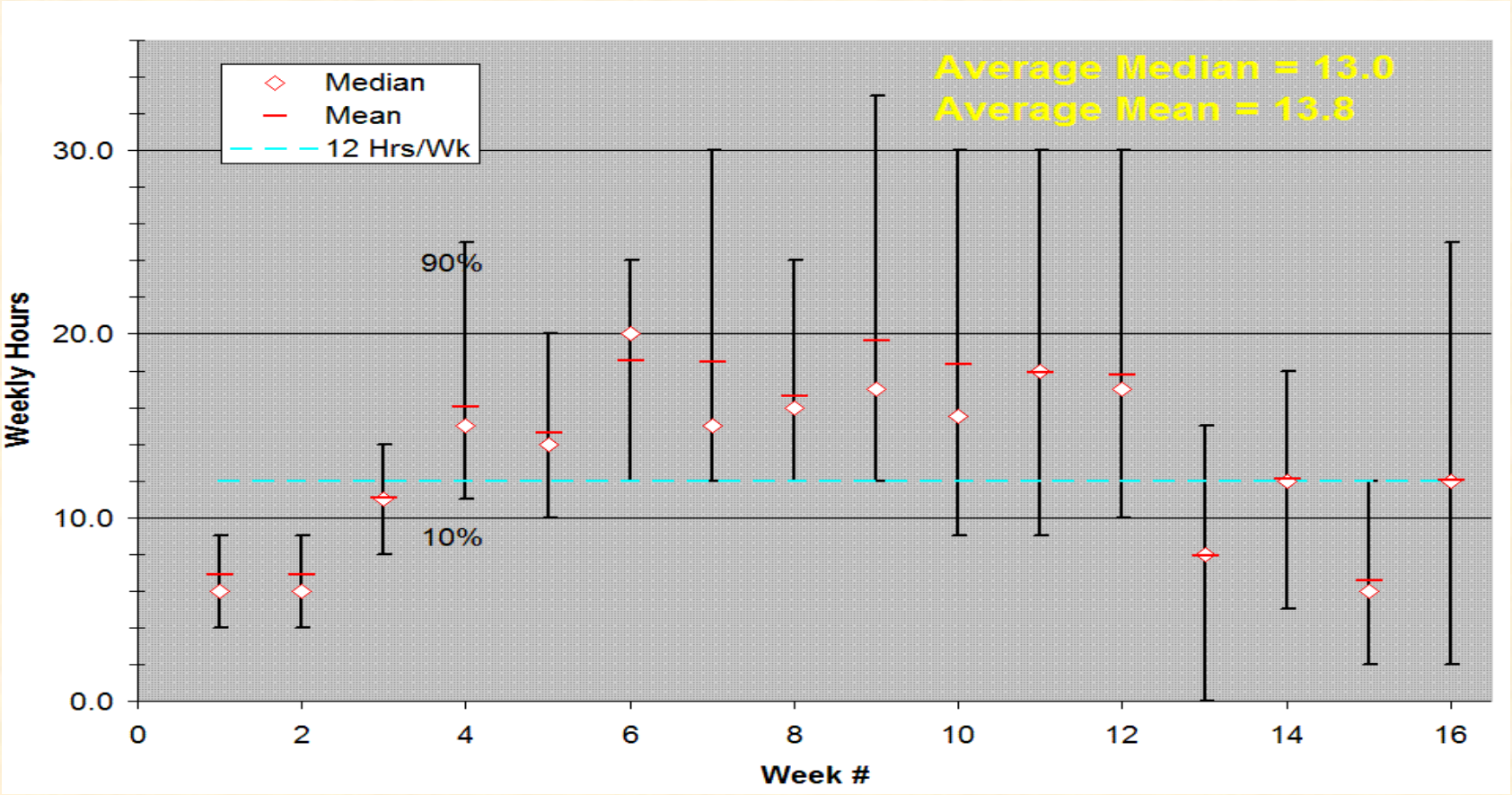
Before (Ineffective Reviews)

EDGE CASE RESEARCH



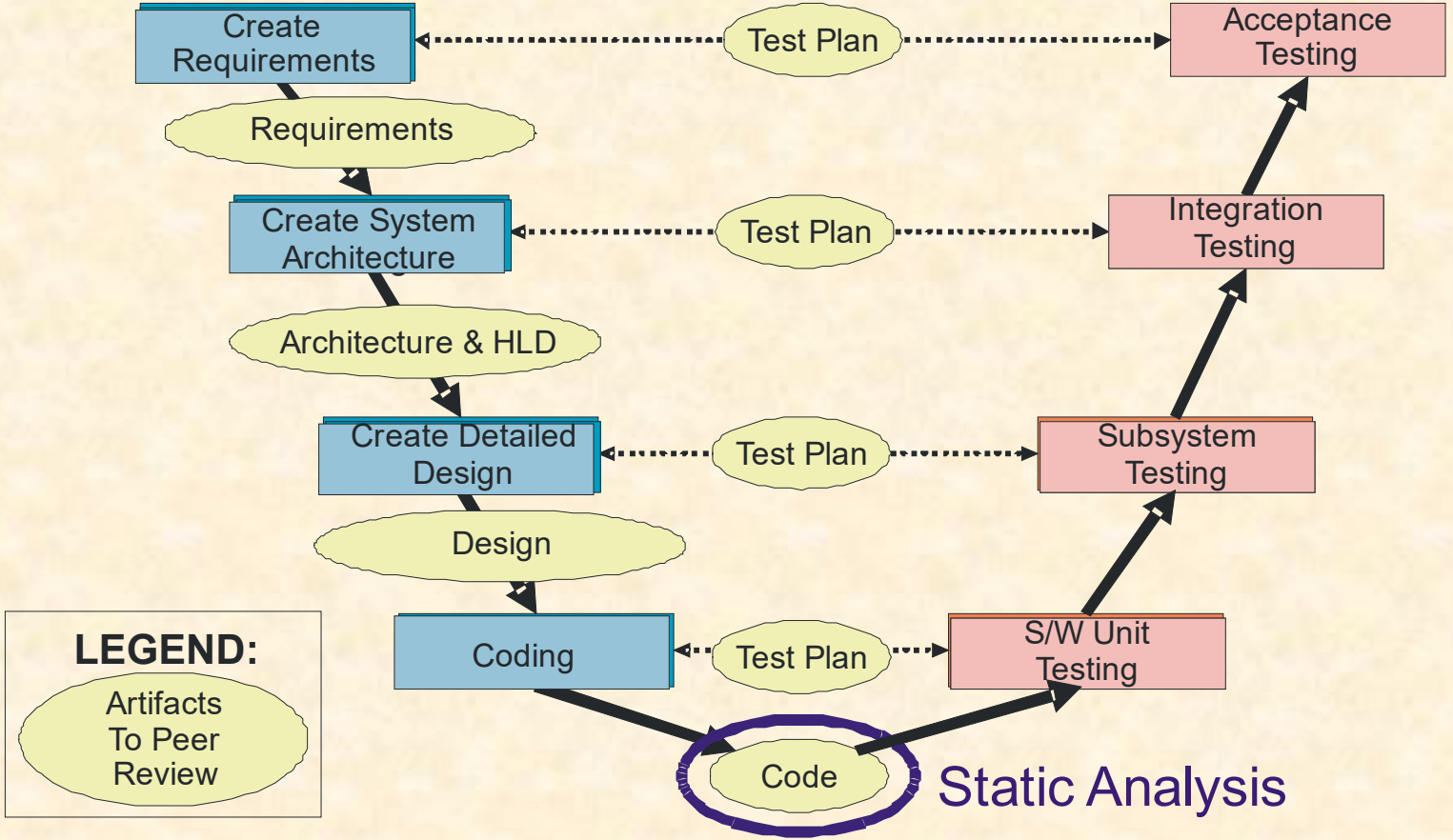
With Weekly Defect Reporting

EDGE CASE RESEARCH



Review More Than Just The Code

EDGE CASE RESEARCH



Economics Of Peer Review

EDGE CASE RESEARCH

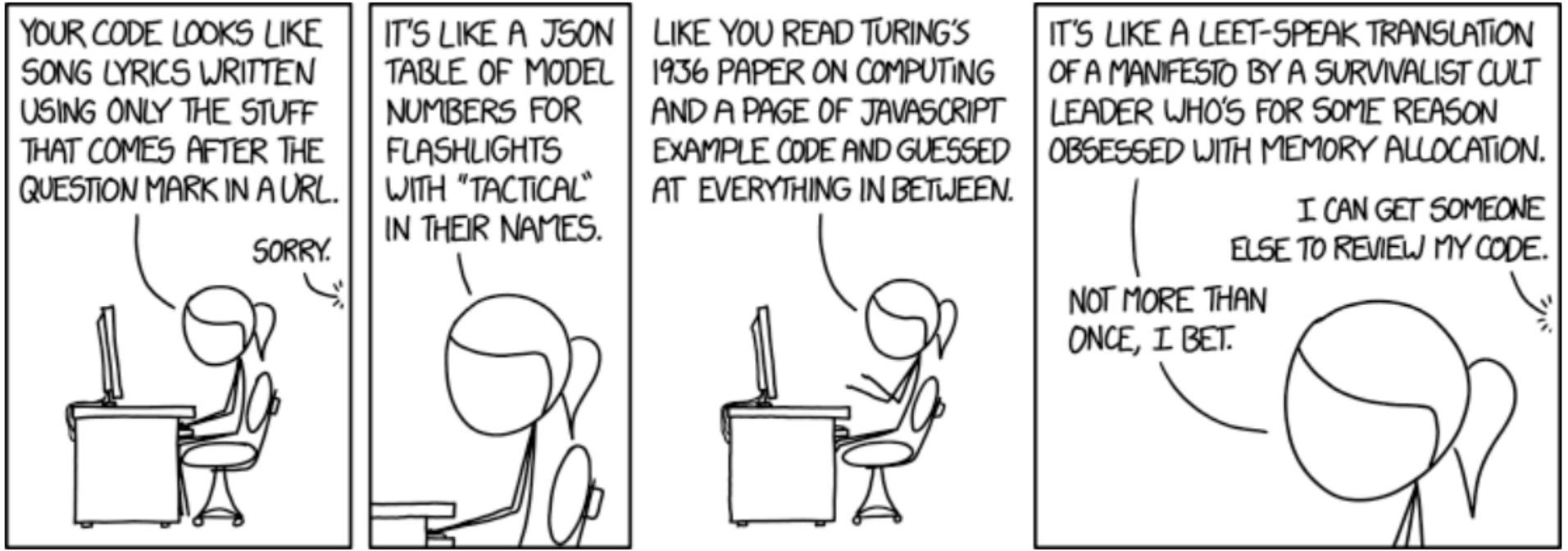
- Peer reviews provide more eyeballs to find bugs in an affordable way
  - Good embedded coding rate is 1-2 lines of code/person-hr
    - (Across entire project, including reqts, test, etc.)
  - A person can review 50-100 times faster than they can write code
    - If you have 4 people reviewing, that is still >10x faster than writing!
  - How much does peer review cost?
    - 4 people * 100-200 lines of code reviewed per hour
    - E.g., 300 lines; 4 people; 2 hrs review+1 hr prep = 25 LOC/person-hr
  - Reviews are only about 5%-10% of your project cost
- Good peer reviews find at least half the bugs!
  - And they find them early, so total project cost can be reduced
- Why is it folks say they don't have time to do peer reviews?

Peer Review Best Practices

EDGE CASE RESEARCH

- Formal reviews (inspections) optimize bugs/$
  - Target 10% of project effort to find 50% of bugs
    - You can review 100x faster than write code; it's cheap
  - Review everything written down, not just code
  - Use a perspective-based checklist to find more bugs

- Review pitfalls
  - If your reviews find <50% of defects, they are BROKEN
    - The 80/20 rule does NOT apply to review formality! Formal reviews are best.
    - You can't review at end; need to review throughout project
- Review tools
  - On-line review tools are OK, but not a substitute for in-person meeting
  - Static analysis tools are great – but not a review!



https://www.xkcd.com/1833/

