



**17-356/17-766
SOFTWARE ENGINEERING
FOR STARTUPS**



Carnegie Mellon University
School of Computer Science

Michael Hilton & Heather Miller

Administrativa

HW2: due tonight

HW3: Released soon

Quality Assurance

**The best way to assure quality is
a good process**

“Continuous Integration doesn’t get rid of bugs, but it does make them dramatically easier to find and remove.”

— Martin Fowler, Chief Scientist, ThoughtWorks

History of Continuous Integration



(1999) Extreme Programming (XP) rule: “Integrate Often”



(2000) Martin Fowler posts “Continuous Integration” blog



(2001) First CI tool








(2005) Hudson/Jenkins



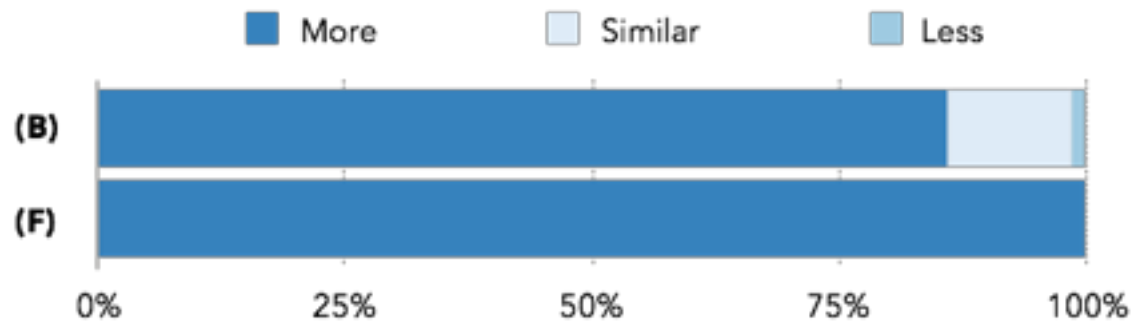
(2011) Travis CI

Sample CI Workflow (Github + Travis)

-  Create Pull Request
-  GitHub tells Travis CI build is mergeable
-  It builds and passes tests
-  Travis updates PR
-  PR is merged

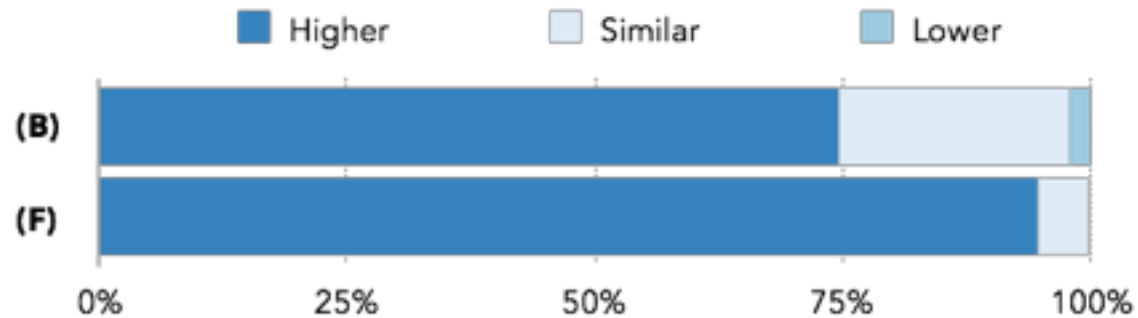
Developers Say:

Do developers on projects with CI give (more/similar/less) value to automated tests?



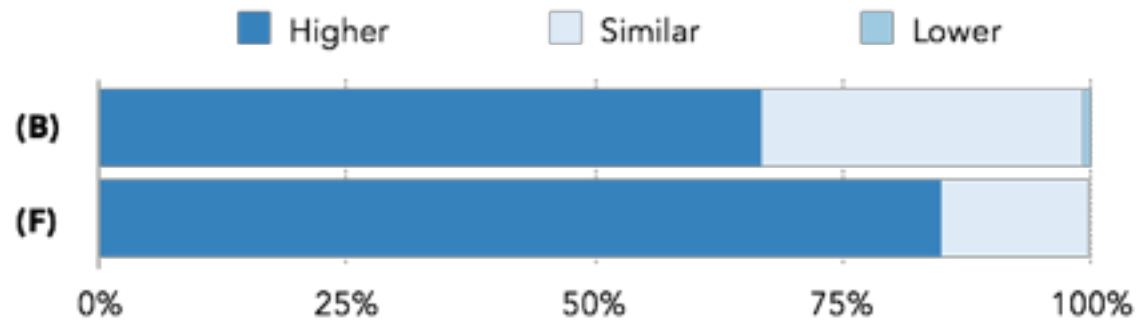
Developers Say:

Do projects with CI have (higher/similar/lower) test quality?



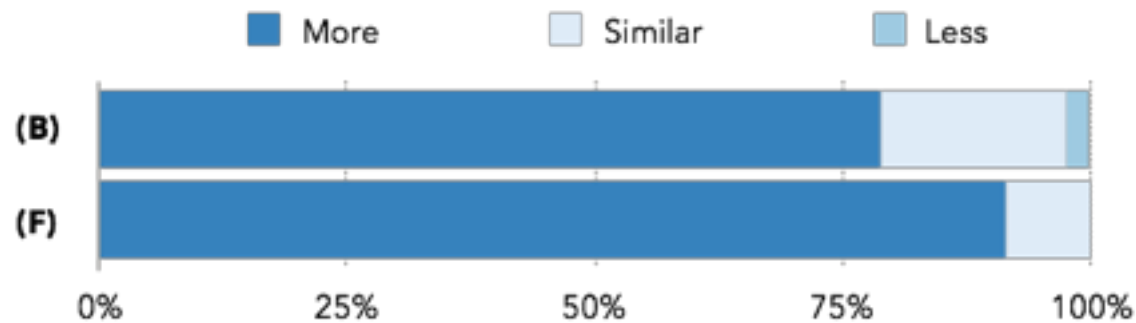
Developers Say:

Do projects with CI have (higher/similar/lower) code quality?

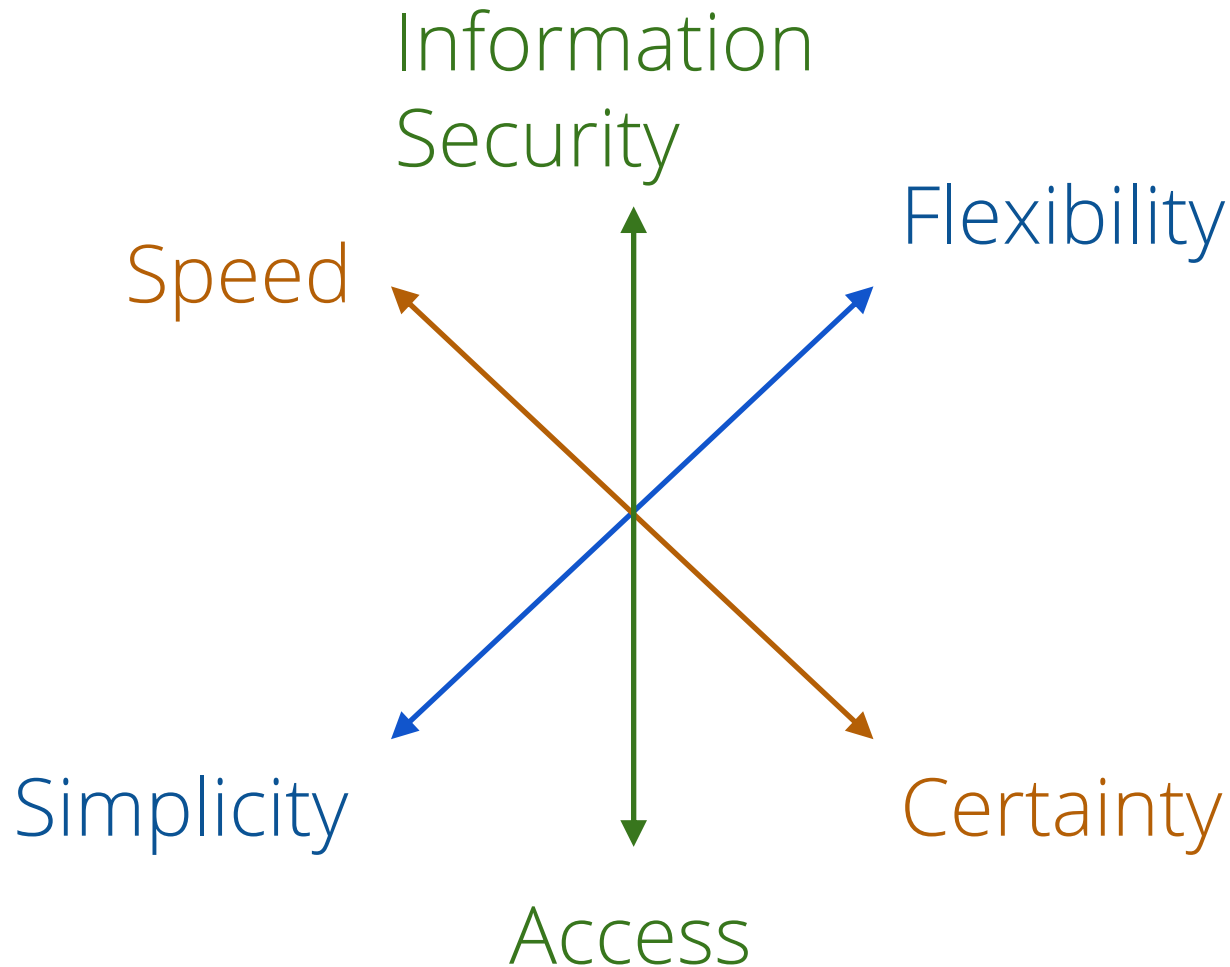


Developers Say:

Are developers on projects with CI (more/similar/less) productive?



Conflicts:



Implications

Write tests that provide value

Spend time to maintain your tests, improve quality, remove unneeded tests

Balance rigorousness of the test suite with speed of running tests

Follow security best practices

Consider long term costs of complex, custom workflows

Observation

Most of the benefits of CI come from running tests

Test Driven Development (TDD)

Three Simple Rules

- 1) You are not allowed to write any production code unless it is to make a failing unit test pass.
- 2) You are not allowed to write any more of a unit test than is sufficient to fail; and compilation failures are failures.
- 3) You are not allowed to write any more production code than is sufficient to pass the one failing unit test.

Test Driven Development (TDD)



From Growing Object-Oriented Software by Nat Pryce and Steve Freeman

<http://www.growing-object-oriented-software.com/figures.html>

@sebrose

<http://cucumber.io>

Why TDD

“The act of writing a unit test is more an act of **design** than of verification.

It is also more an act of **documentation** than of verification.

The act of writing a unit test closes a remarkable number of feedback loops, the least of which is the one pertaining to **verification** of function”.

Advantages of TDD

Clear place to start

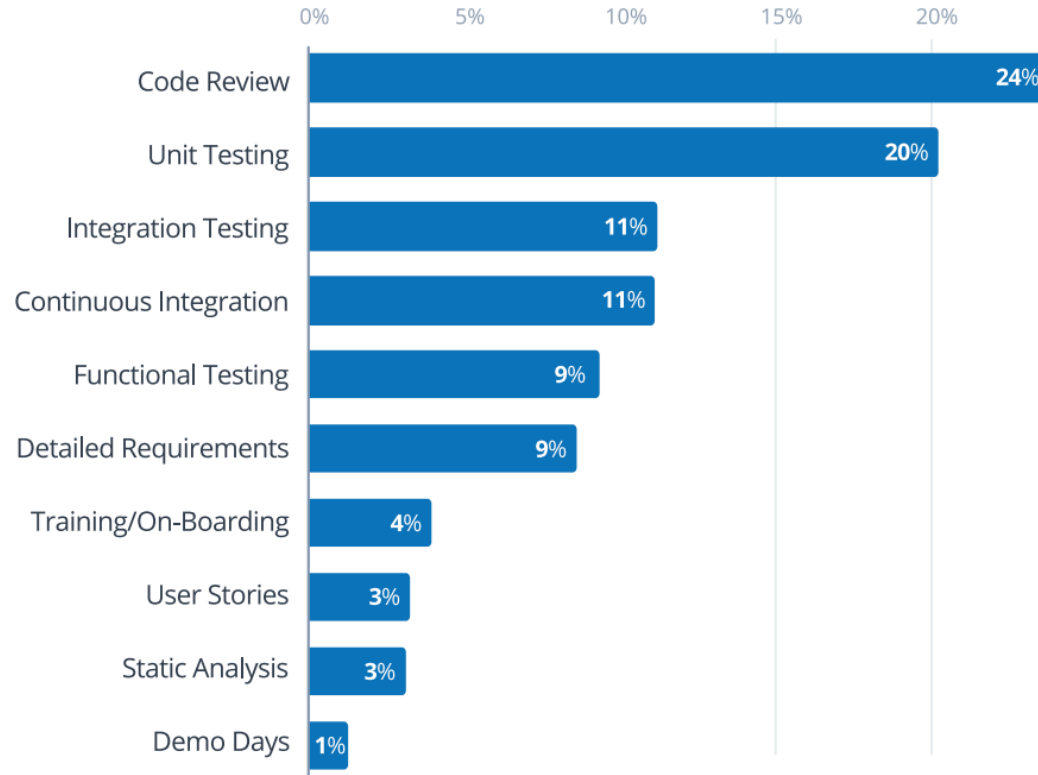
Much less code thrown away, less wasted effort

Less Fear

Side Effect: Robust test suite

Code Review

What is the number #1 thing a company can do to improve code quality?



n = 856

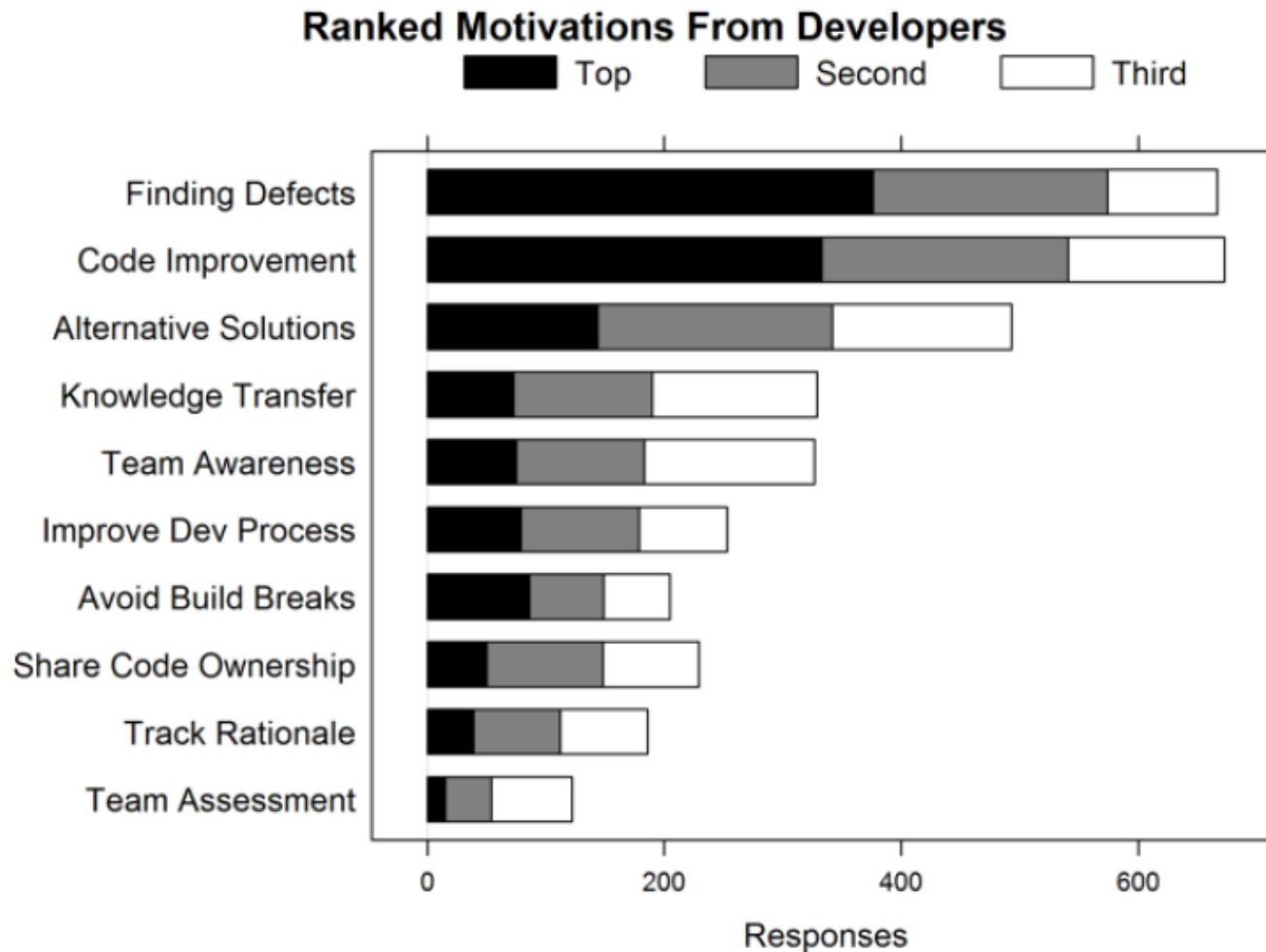
Motivation:

“Many eyes make all bugs shallow”

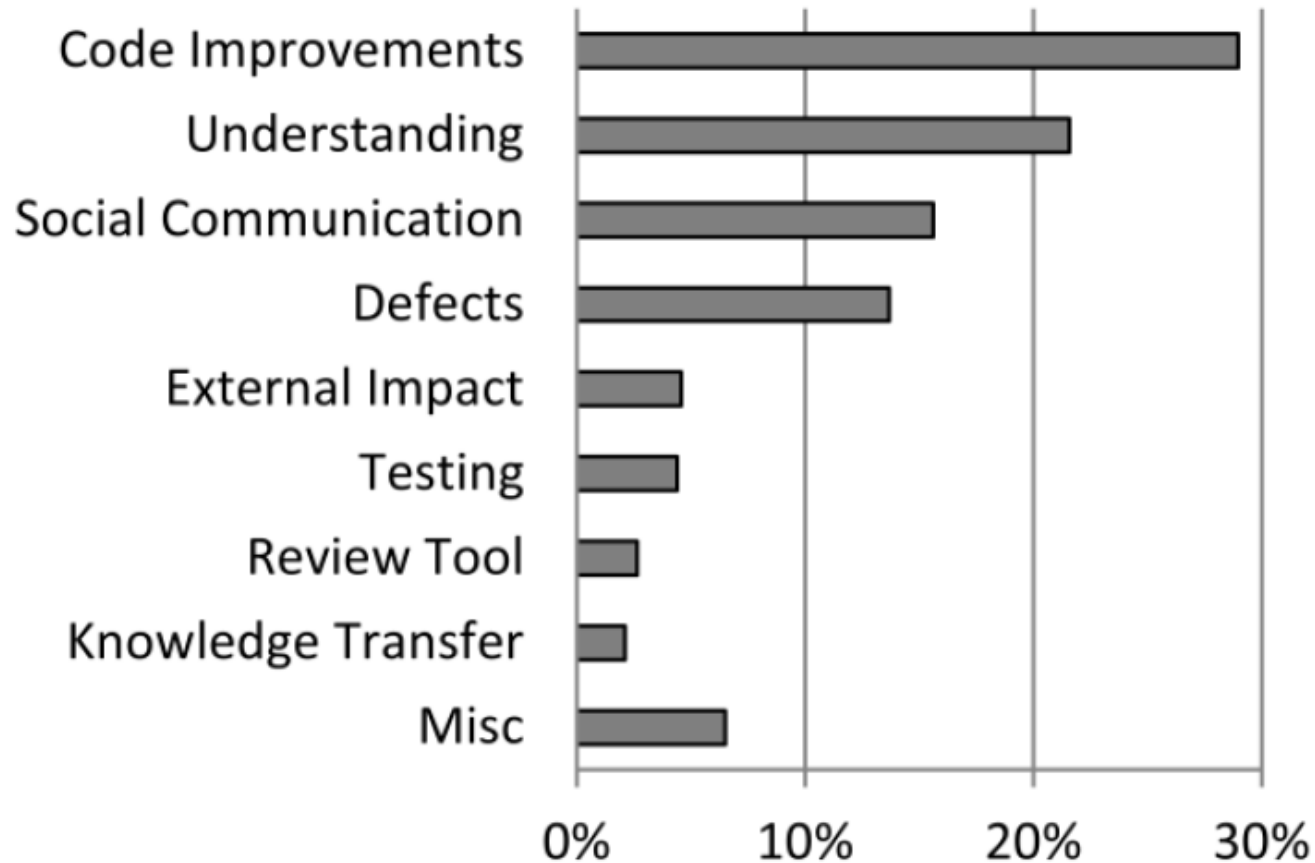
“Have peers, rather than customers,
find defects” -Karl Wieggers

Expectations and Outcomes of Code Reviews

Code Review at Microsoft



Outcomes (Analyzing Reviews)



Mismatch of Expectations and Outcomes

Low quality of code reviews

- Reviewers look for easy errors, as formatting issues
- Miss serious errors

Understanding is the main challenge

- Understanding the reason for a change
- Understanding the code and its context
- Feedback channels to ask questions often needed

No quality assurance on the outcome

Code Review at Google

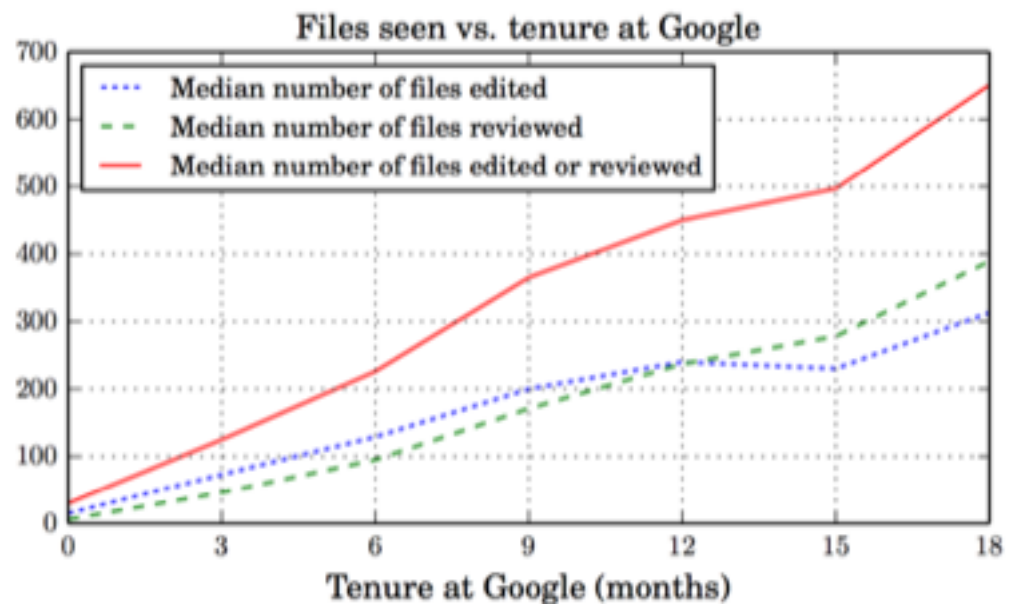
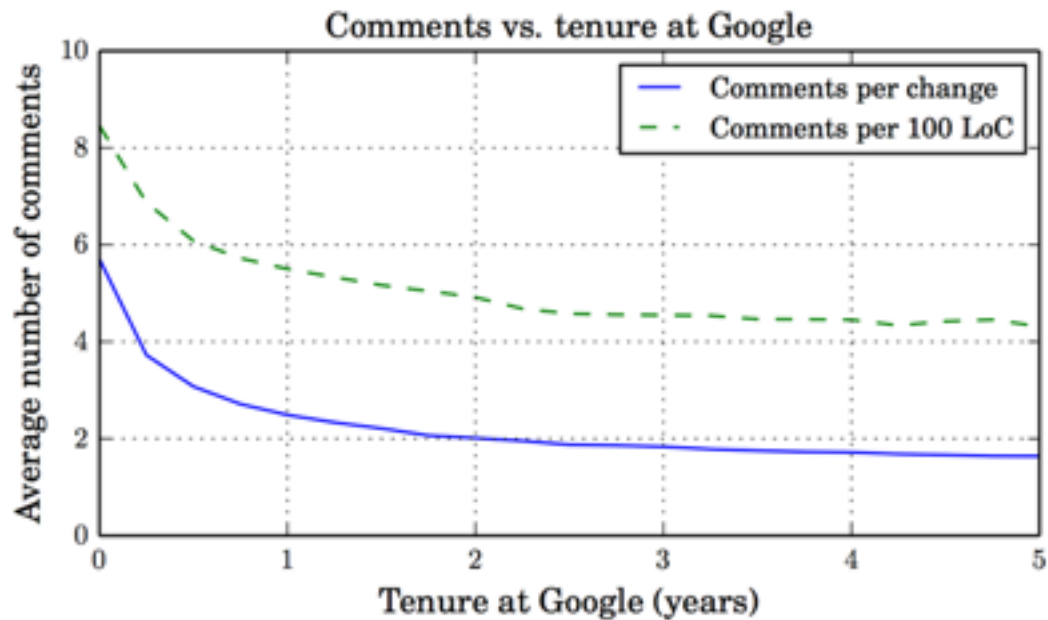
Introduced to “force developers to write code that other developers could understand”

3 Found benefits:

- checking the consistency of style and design
- ensuring adequate tests
- improving security by making sure no single developer can commit arbitrary code without oversight

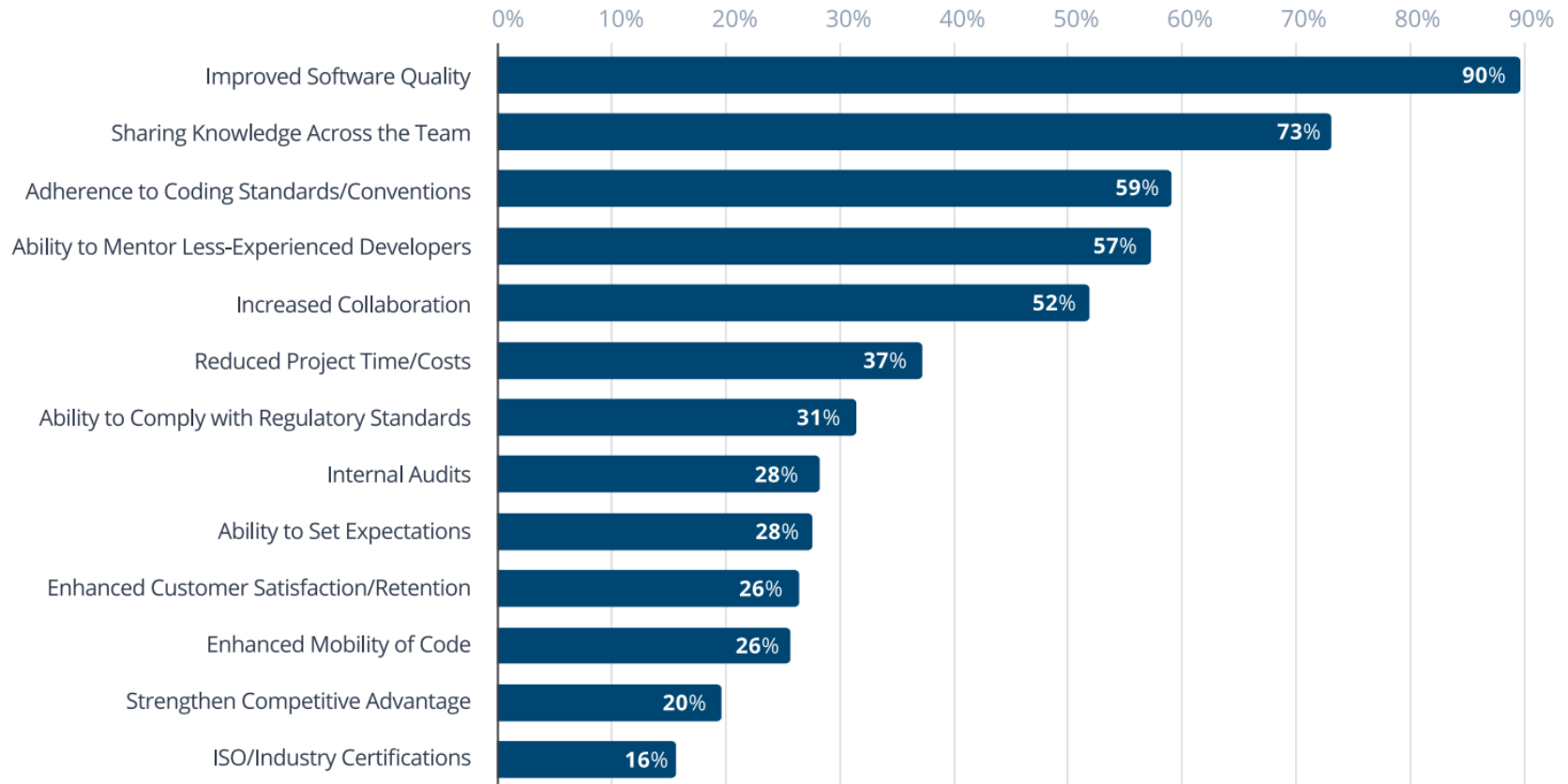
Reviewing Relationships





The State of Code Review survey

What do you believe are the most important benefits of code review?



n = 1129

Code Review

- Start with the “big ideas”
- Automate the little things
- Focus on understanding
- Remember a person wrote the code
- Don't overwhelm the person with feedback

Checklists



| OFFICIAL A.A.F. PILOT'S CHECK LIST | |
|--|--|
| B-17F AND B-17G For detailed instructions see Pilot's Handbook AN 01-208F-1 or AN 01-208G-1 in date case | |
| PILOT | CO-PILOT |
| BEFORE STARTING | BEFORE TAKE OFF |
| 1. Plan's Fire Right — <u>Checked</u> . | 1. Tail Wheel — <u>Locked</u> . |
| 2. Form 1A, Form F, Wright and Balance — <u>Checked</u> . | 2. Gyro — <u>Set</u> . |
| 3. Controls and Seats — <u>Checked</u> — <u>Checked</u> . | 3. Generators — <u>On</u> . |
| 4. Fuel Transfer Valves and Switch — <u>OK</u> . | AFTER TAKE OFF |
| 5. Instruments — <u>Cold</u> . | 1. Wheels — <u>Pilot's Signal</u> . |
| 6. Gyros — <u>Uncaged</u> . | 2. Power Reduction. |
| 7. Fuel Shut-off Switches — <u>Open</u> . | 3. Cool Flaps. |
| 8. Over Switch — <u>Neutral</u> . | 4. Wheel Check — <u>OK Right</u> . |
| 9. Cool Flaps — <u>Open Right</u> — <u>Open</u> <u>Left</u> — <u>Locked</u> . | <u>OK Left</u> . |
| 10. Carbs — <u>OK</u> . | BEFORE LANDING |
| 11. Life raft — <u>Checked</u> . | 1. Radio Call Altimeter — <u>Set</u> . |
| 12. Hydraulic — <u>Closed</u> . | 2. Crew Positions — <u>OK</u> . |
| 13. High RPM — <u>Checked</u> . | 3. Auto Pilot — <u>OK</u> . |
| 14. Auto Pilot — <u>OK</u> . | 4. Booster Pumps — <u>On</u> . |
| 15. De-icers and Anti-icers Wing and Prop. — <u>OK</u> . | 5. Machine Controls — <u>Auto Rich</u> . |
| 16. Cabin Warm — <u>OK</u> . | 6. Intercooler — <u>Set</u> . |
| 17. Generators — <u>OK</u> . | 7. Carburetor Filters — <u>Open</u> . |
| STARTING ENGINES | 8. Wing De-icers — <u>OK</u> . |
| 1. Fire Guard and Call Clear — <u>Left</u> — <u>Right</u> . | 9. Landing Gear. |
| 2. Master Switches — <u>On</u> . | a. Visual — <u>Down right</u> <u>Down left</u> <u>Tail wheel</u> <u>Down</u> , <u>Antenna In</u> . |
| 3. Battery Switches and Inverters — <u>On and Checked</u> . | b. Light — <u>OK</u> . |
| 4. Parking Brakes — Hydraulic Check <u>On — Checked</u> . | c. Switch <u>OK</u> — <u>Neutral</u> . |
| 5. Booster Pumps — Pressure — <u>On</u> <u>and Checked</u> . | 10. Hydraulic Pressure — <u>OK</u> . Valve <u>closed</u> . |
| 6. Carburetor Filters — <u>Open</u> . | 11. RPM 2100 — <u>Set</u> . |
| 7. Fuel Quantity — Gallons per tank. | 12. Turbos — <u>Set</u> . |
| 8. Start Engines. | 13. Flaps 1/2 — 1/2 <u>Down</u> . |
| a. Fire Extinguisher Engine Safety — <u>Set</u> — <u>Checked</u> . | FINAL APPROACH |
| b. Prime — <u>As Necessary</u> . | 14. Flaps — <u>Pilot's Signal</u> . |
| | 15. High RPM — <u>Pilot's Signal</u> . |



The Checklist: <https://www.newyorker.com/magazine/2007/12/10/the-checklist>

Activity

Code Review Checklist

Checklist Suggestions

- Review the unit tests
- Consider readability
- Best practices
- Quality Attributes
- Design Principles
- Communication around code (names, comments, etc)
- Maintain the Checklist