

# Pair Programming Showdown!

BarCampRDU 2008

Matthew Bass  
[pelargir@gmail.com](mailto:pelargir@gmail.com)  
<http://matthewbass.com>

# Ground Floor

- What is pair programming?

# Agile Manifesto

- Individuals and interactions...
- ...over process and tools



# The Black Sheep of Agility

- why does pairing get a bad rap?

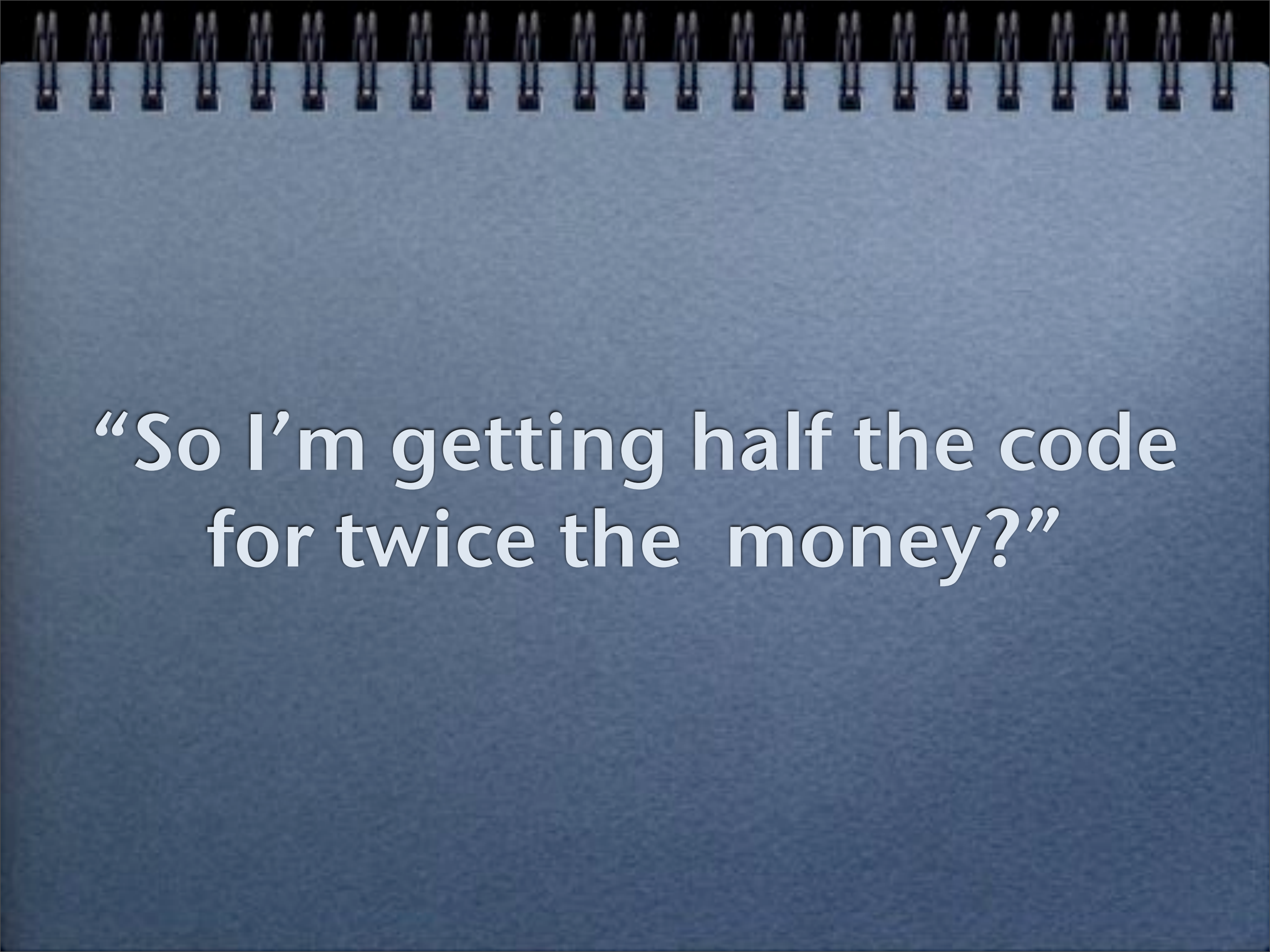




*X-Scream Programming*



# Dogmatism

A blue spiral-bound notebook with a silver metal spiral binding at the top. The text is written in white on the notebook page.

“So I’m getting half the code  
for twice the money?”



# Managers say...

- ☐ It's unproductive
- ☐ It's slow
- ☐ It's wasteful
- ☐ It drives developers away
- ☐ It's unproven



# Developers say...

- ☐ It's frustrating
- ☐ It's difficult
- ☐ It's uncomfortable
- ☐ It's tiring
- ☐ I don't need to do it



Why do initial attempts at  
pairing fail?

No preparation?



A blue spiral-bound notebook with a silver metal spiral binding at the top. The text is written on the front cover.

Is pairing for everyone?

Nope!

# Pairing pragmatically

- Throw out what doesn't work
- Keep what works
- Be willing to adapt
- Communicate frequently
- Don't be dogmatic\*

\* Don't do something for its own sake



# Pairing pragmatically

- Navigator vs. driver
- Tie breaking
- Ping ponging
  - Own the code
  - Be patient (but not forever)

# Navigator vs. driver

- Driver

- Controls the mouse / keyboard

- Down in the details

- Navigator

- Thinks higher level

- Watches for typos, logic errors, etc.

- Switch off?



# Tie breaking

- ❑ When a disagreement occurs...
- ❑ Rank importance (1 to 3)
- ❑ Count down, then reveal
- ❑ Involve a third party when necessary
- ❑ Tweak as needed

# Ping ponging

- ☐ Makes the most sense with TDD
- ☐ One person writes tests...
- ☐ The other fixes them
- ☐ Swap?
- ☐ Turns coding into a game!

# Own the code

- ☐ Anyone can change anything
- ☐ Anyone can wave a red flag
- ☐ Code you just wrote is fair game!
- ☐ No bouncing back and forth



# Be patient

- ...but not forever
- Antonyms:
  - hasty
  - impetuous
- very important when mentoring

A blue spiral-bound notebook with the text "So what?" written in the center.

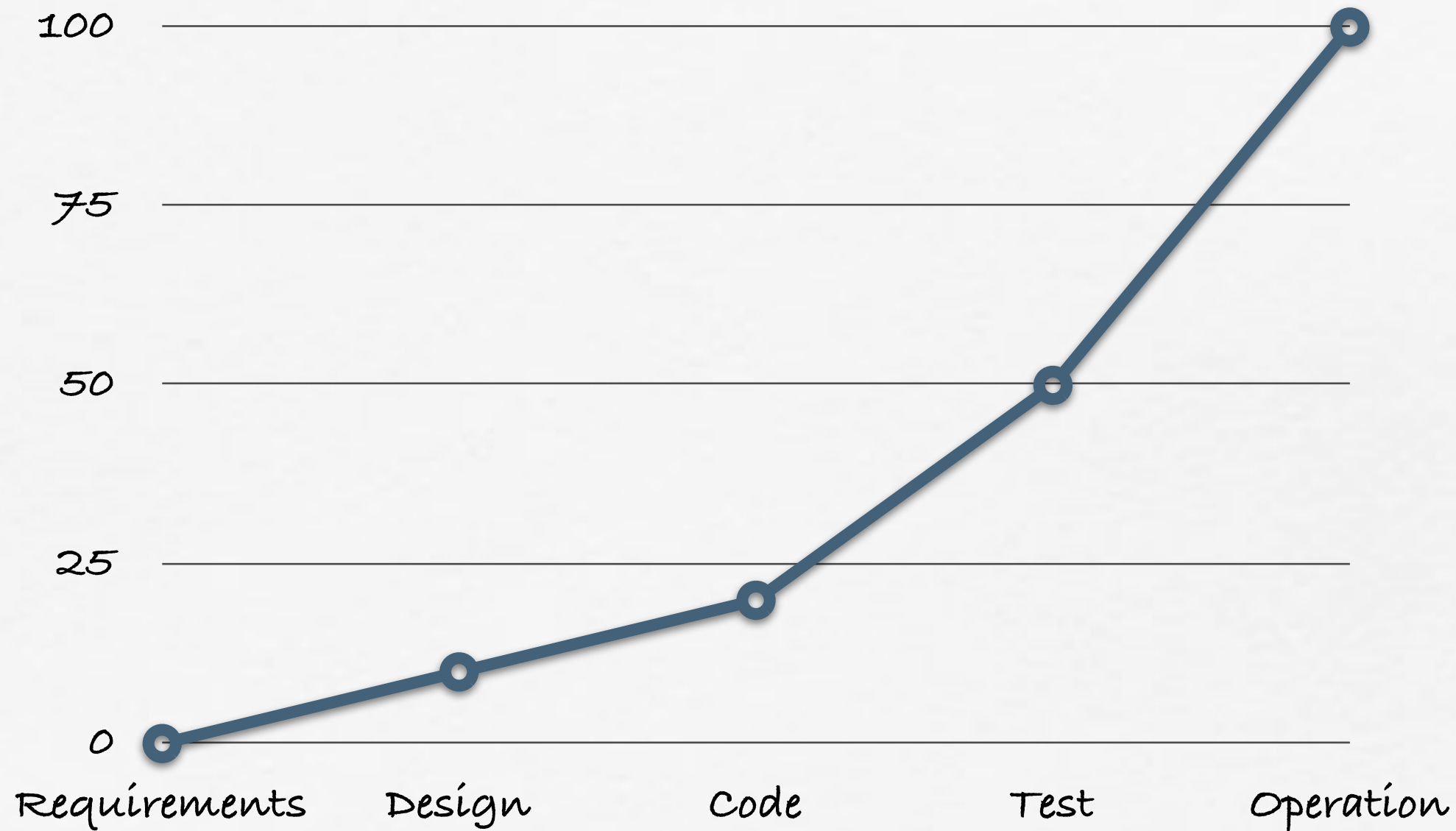
So what?

# Benefits

- ❑ Reduced development time
- ❑ Better code quality



# Cost to correct a defect



[BOEH]

"Software Defect Reduction Top 10 List"

# Code quality

- On a typical project:
  - 40-50% of effort is avoidable rework
  - Peer review catches 60% of defects
  - Disciplined personal practices reduce defect introduction by 75%

# Benefits

- ☐ Reduced development time
- ☐ Better code quality
- ☐ Better design
- ☐ Built-in code reviews
- ☐ Built-in cross-training
- ☐ Bad programming habits get squashed



# Benefits

- ☐ Increases truck number
- ☐ Mandates undivided attention
- ☐ Improves communication
- ☐ Produces "flow"

“Flow is a condition of deep, nearly meditative involvement. In this state, there is a gentle sense of euphoria, and one is largely unaware of the passage of time.” -- Tom DeMarco

# Benefits

- ☐ Increases truck number
- ☐ Mandates undivided attention
- ☐ Improves communication
- ☐ Produces "flow"
- ☐ Tips and tricks
- ☐ More fun!



## Another benefit

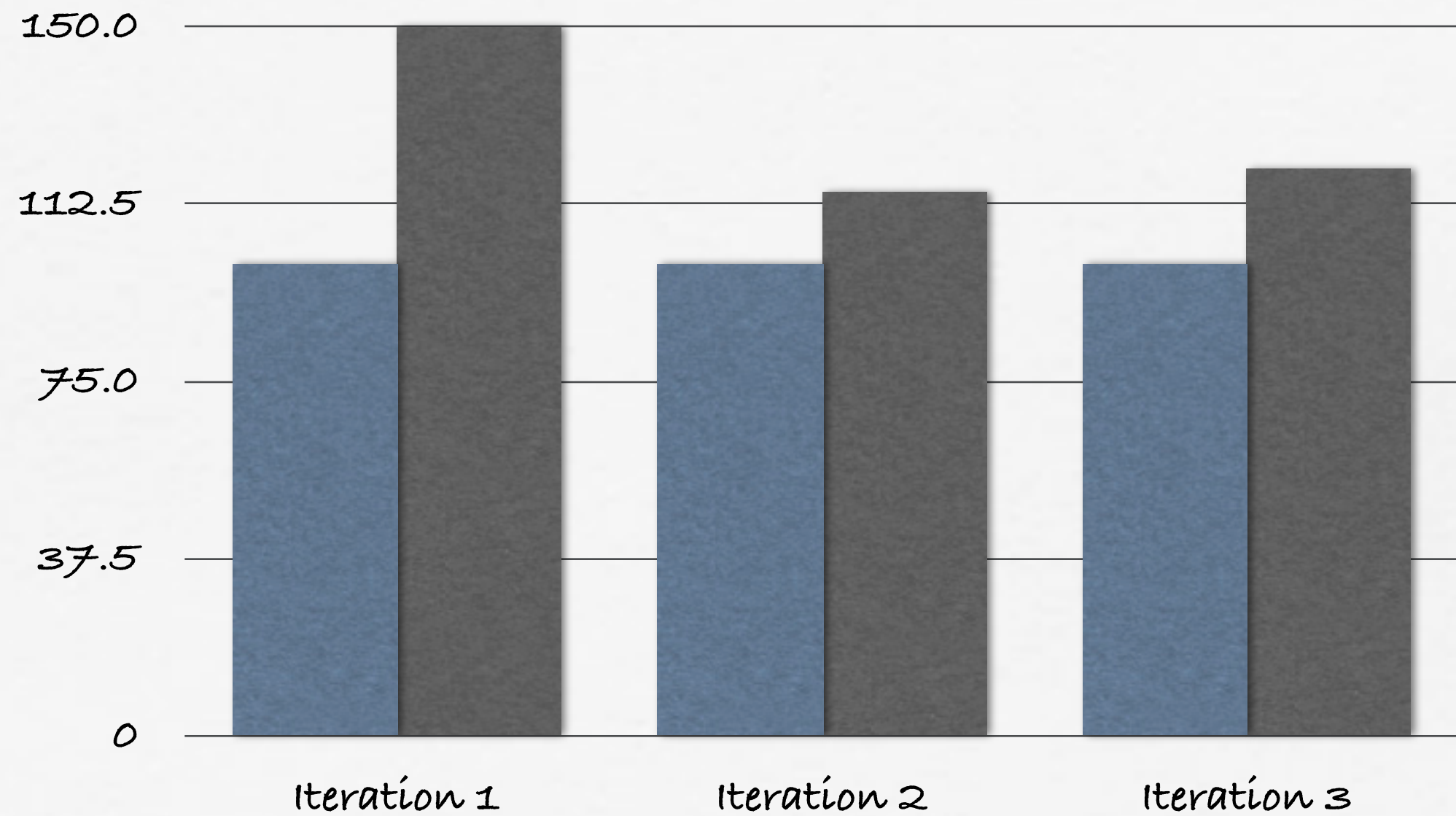
- ☐ Mentoring!
- ☐ Master / apprentice model
- ☐ Productivity moves up in increments
- ☐ Earlier tangible contributions
- ☐ Hiring / auditioning

# Metrics

- How does pairing stack up?
  - Economics
  - Satisfaction
  - Design quality
- University of Utah study

# Relative time

One Individual  
Two Collaborators

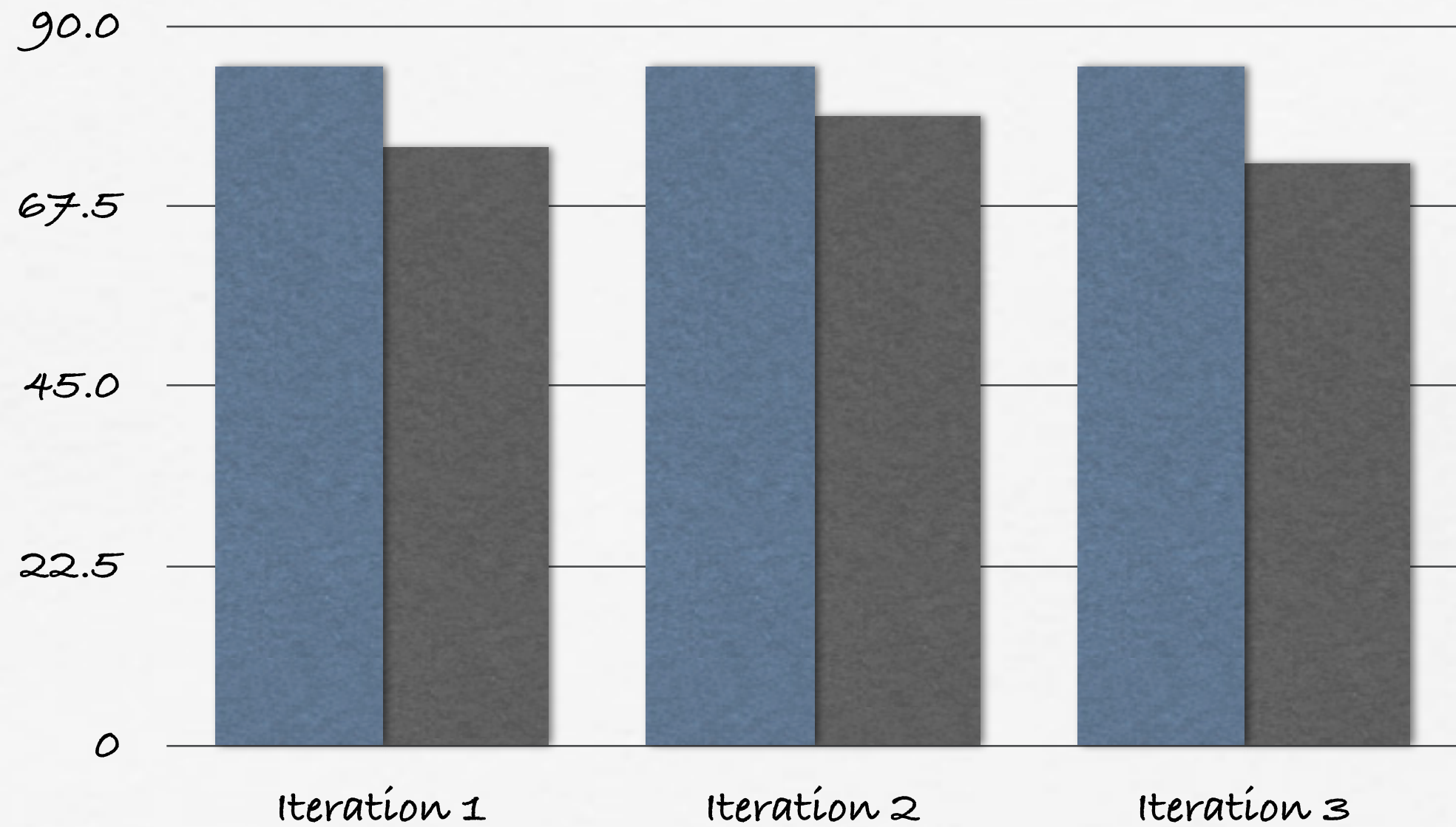


[ACWL]



# Defect count

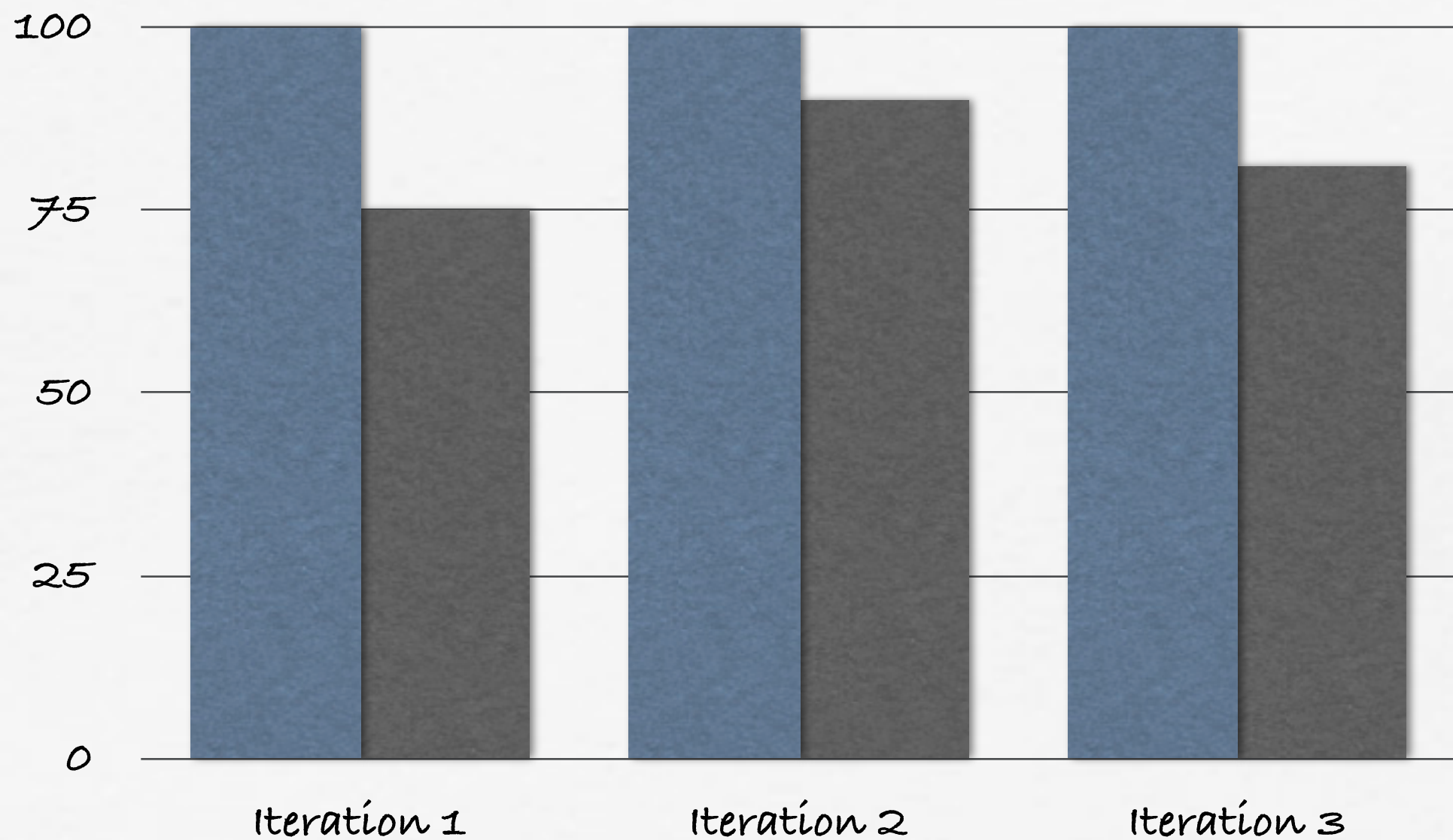
One Individual  
Two Collaborators



[ACWL]

# Lines of code

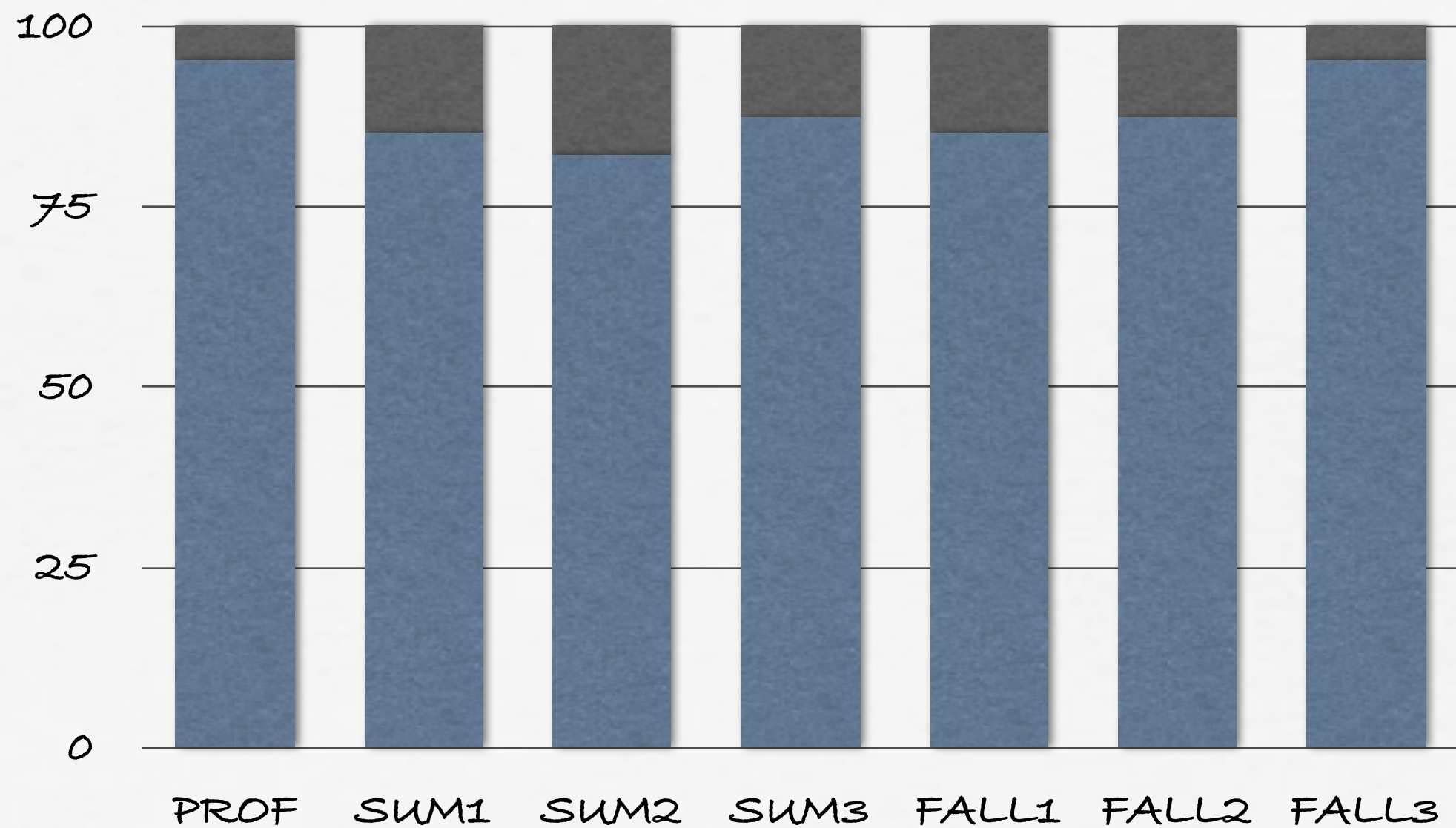
■ One Individual  
■ Two Collaborators



[ACWL]

# Enjoyed pairing?

Disagree  
Agree



[ACLW]



# Where to go from here...

- ☐ Try pairing for an hour
  - ☐ Okay, a half hour
  - ☐ 15 minutes?
- ☐ Or just do a peer code review
- ☐ Take gradual steps
- ☐ Do your research

# Resources

"The Cost and Benefits of Pair Programming"  
(Alistair Cockburn, Laurie Williams) <http://tinyurl.com/z19i>

"Pair Programming Illuminated" (Robert Kessler)  
<http://tinyurl.com/4ezmms>

"Fearless Change: Patterns for Introducing New Ideas"  
(Mary Lynn Manns) <http://tinyurl.com/5bngkh>

# References

- [LITT] Todd Little. (<http://alistair.cockburn.us/index.php/Image:XScreamProgramming.gif>).
- [BOEH] "Software Defect Reduction Top 10 List," by Barry Boehm and Victor R. Basili, IEEE Computer, January 2001. (<http://www.cebase.org/www/resources/reports/usc/usccse2001-515.pdf>).
- [DEMA] "Peopleware: Productive Projects and Teams," by Tom DeMarco and Timothy Lister, Dorset House Publishing Company, February, 1999. (<http://www.amazon.com/Peopleware-Productive-Projects-Teams-Second/dp/0932633439>).
- [ACWL] "Costs and Benefits of Pair Programming," by Alistair Cockburn and Laurie Williams, January, 2000. ([http://alistair.cockburn.us/index.php/Costs\\_and\\_benefits\\_of\\_pair\\_programming](http://alistair.cockburn.us/index.php/Costs_and_benefits_of_pair_programming)).
- [NB] NetBeans Collaboration Module. (<http://www.netbeans.org/kb/articles/quickstart-collaboration.html>).
- [FOW] "The Improvement Ravine," by Martin Fowler, October, 2006. (<http://www.martinfowler.com/bliki/ImprovementRavine.html>).





# Thanks!

Matthew Bass  
[pelargir@gmail.com](mailto:pelargir@gmail.com)  
<http://matthewbass.com>