Put it to the Test: Using Lightweight Experiments to Drive Process Improvement

Michael Keeling
Institute for Software Research
Carnegie Mellon University

Process improvement is important but…

The Scientific Method

Ask a Question

↓

Do Background Research

↓

Construct a Hypothesis

↓

Test Hypothesis, Collect Data

↓

Analyze Data, Conclusions

↓

Report Results

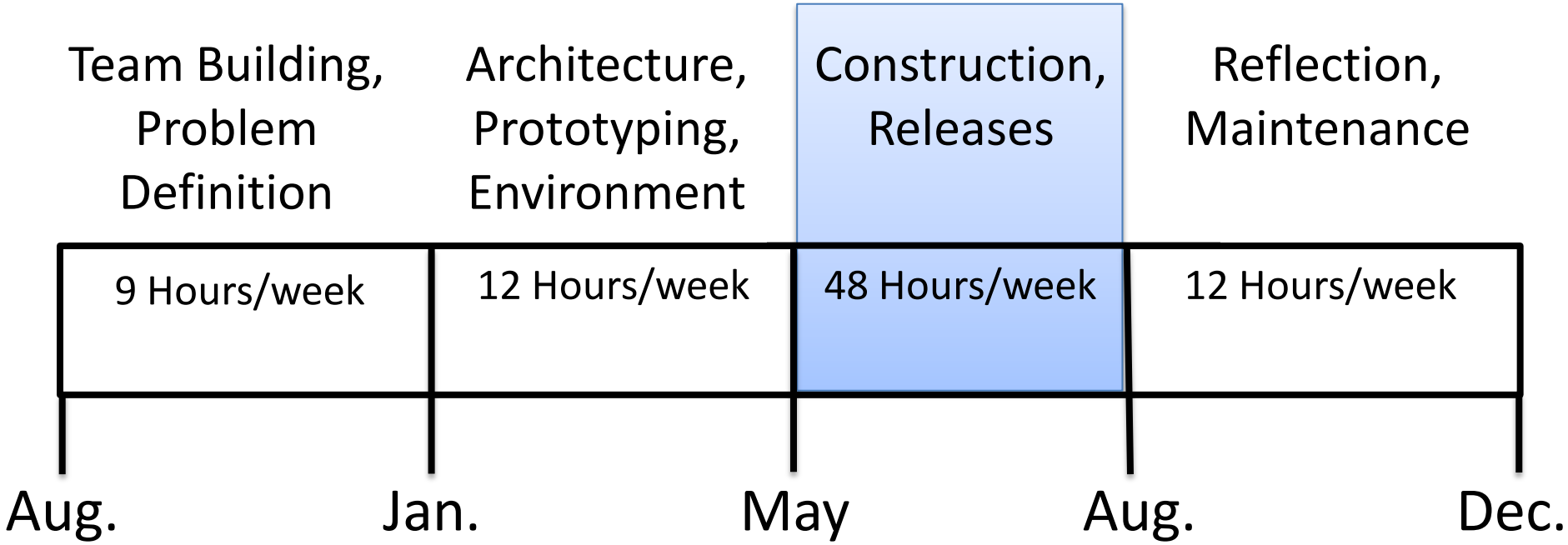
On Experimentation

Lightweight Experiments

- Focus on a single question
- Divide work, not teams
- Release!
- Use the data you have
- Statistical significance is overrated
- Retrospectives get immediate value

One Year Ago…

Master of Software Engineering Studio Timeline



Academic and client deliverables are made throughout the 16 month program.

Project Workbook - Squa... | SQUARE :: Security Qualit...

http://msesrv4d-vm.mse.cs.cmu.edu:8686/Square_QA/#chooseStep/choose-step

Google Reader | Twitter | Other bookmarks

SQUARE
Security Quality Requirements Engineerings

admin (Lead Requirements Engineer) | site settings | preferences | help | sign out

Choose Projects > Demo Project > Choose a Step

Edit Project Settings View Project Members

Determine Context	
Step 1: Agree On Definitions	In Progress
Step 2: Identify Assets and Goals	In Progress
Step 3: Collect Artifacts	In Progress
Gather Security Requirements	
Step 4: Perform Risk Assessment	In Progress
Step 5: Select Elicitation Technique	In Progress
Step 6: Elicit Security Requirements	In Progress
Analyze Requirements	
Step 7: Categorize Requirements	In Progress
Step 8: Prioritize Requirements	In Progress
Step 9: Inspect Requirements	In Progress

SQUARE TOOL - POWERED BY CERT
CARNEGIE MELLON UNIVERSITY
All rights reserved

Available from: http://www.cert.org/sse/square-tool.html

We decided to use XP but…

Pair Programming

Program Alone, Inspection

Findings from Academic Research

- Pair Programming **requires between 15% and 60% more effort** than programming alone.
- Pair Programming is between **20% and 40% faster** than programming alone.
- Pair Programming produces code with **about the same quality** as programming alone with inspection.

Concerns with Academic Research

Temporary groups – **not "true" teams**

Undergrads – **not experienced engineers**

"Toy" Projects – **one and done, no client**

Our Hypothesis

We should be able to approximately reproduce results from previous academic experiments.

Lightweight Experiments

- Focus on a single question
- **Divide work, not teams**
- **Release!**
- Use the data you have
- Statistical significance is overrated
- Retrospectives get immediate value

Experiment Plan

Analysis of Pair Program...

http://msesrv4d-vm.mse.cs.cmu.edu/wiki/Analysis_of_Pair_Programming

Google Reader | Twitter | Shorten with bit.ly | NetHealth | API Search | 960 Gridder | InfoQ: Philippe Kruc... | Welcome to the Co... | Other bookmarks

Have you hugged your mentor today?

Log in

page | discussion | view source | history

square root

- Main Page
- Project Workbook
- Team Processes
- Meeting Notes
- Sharepoint
- Information Technology
- Template Library
- Help

navigation

- Community portal
- Current events
- Recent changes
- Random page

search

Go | Search

toolbox

- What links here
- Related changes

Analysis of Pair Programming

Given that the team has been expending 48 hours on Studio pairing up on tasks, it is feasible to quantitiatively analyze the effectiveness of pair-programming

Contents [hide]

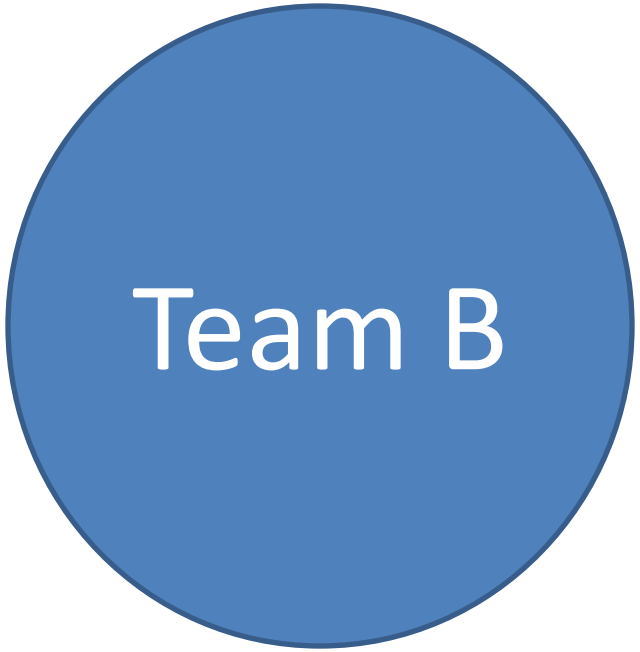
1 Hypotheses
2 Questions
3 Metrics and Data
4 Procedure
4.1 Milestone Category Types
4.2 Defect categories, types, and reasons
4.3 Milestone Allocation
5 Analysis and Reflection
6 References

Hypotheses

1. Pair programming produces code of same or better quality than as Fagan inspection programming alone.
2. Pair Programming requires 15 - 80% more effort to complete a feature than programming alone.
3. Pair Programming requires 60 – 80% less calendar time to complete a feature than programming alone.
4. The most effective pairs have similar problem solving styles, creative problem solving styles, and personalities.
   - Define "effective"

Questions

Experiment Groups



Control Group

Experiment Group

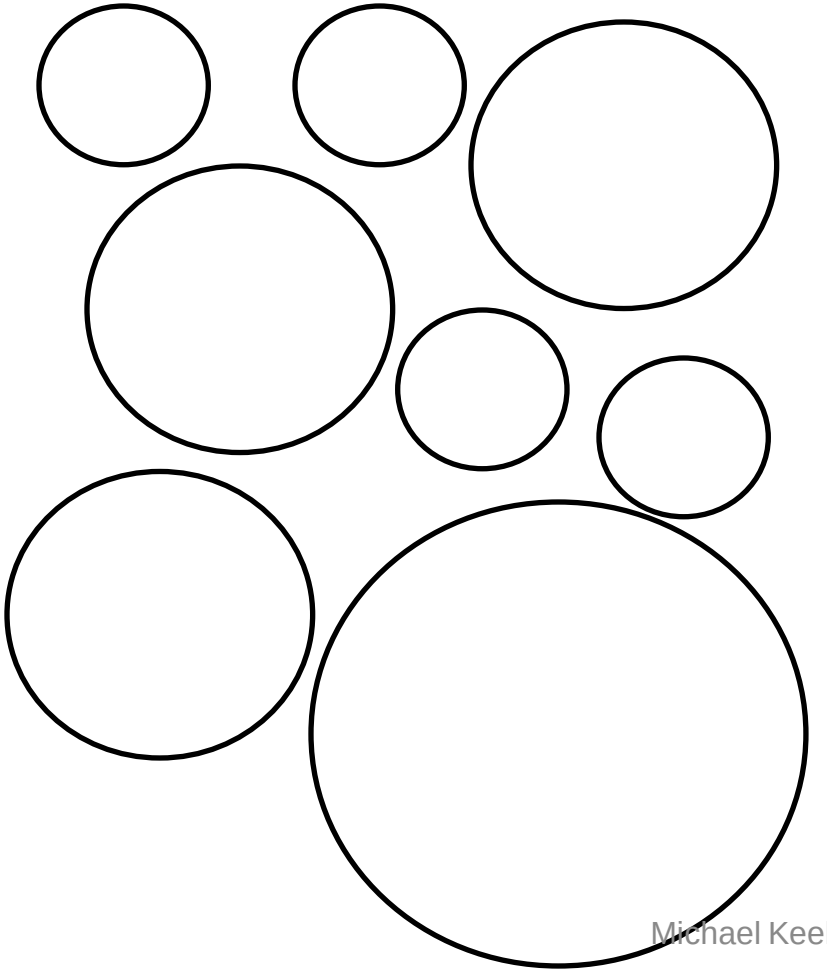
Experiment Groups

Control Group

Experiment Group

Experiment Planning Process

Product Backlog

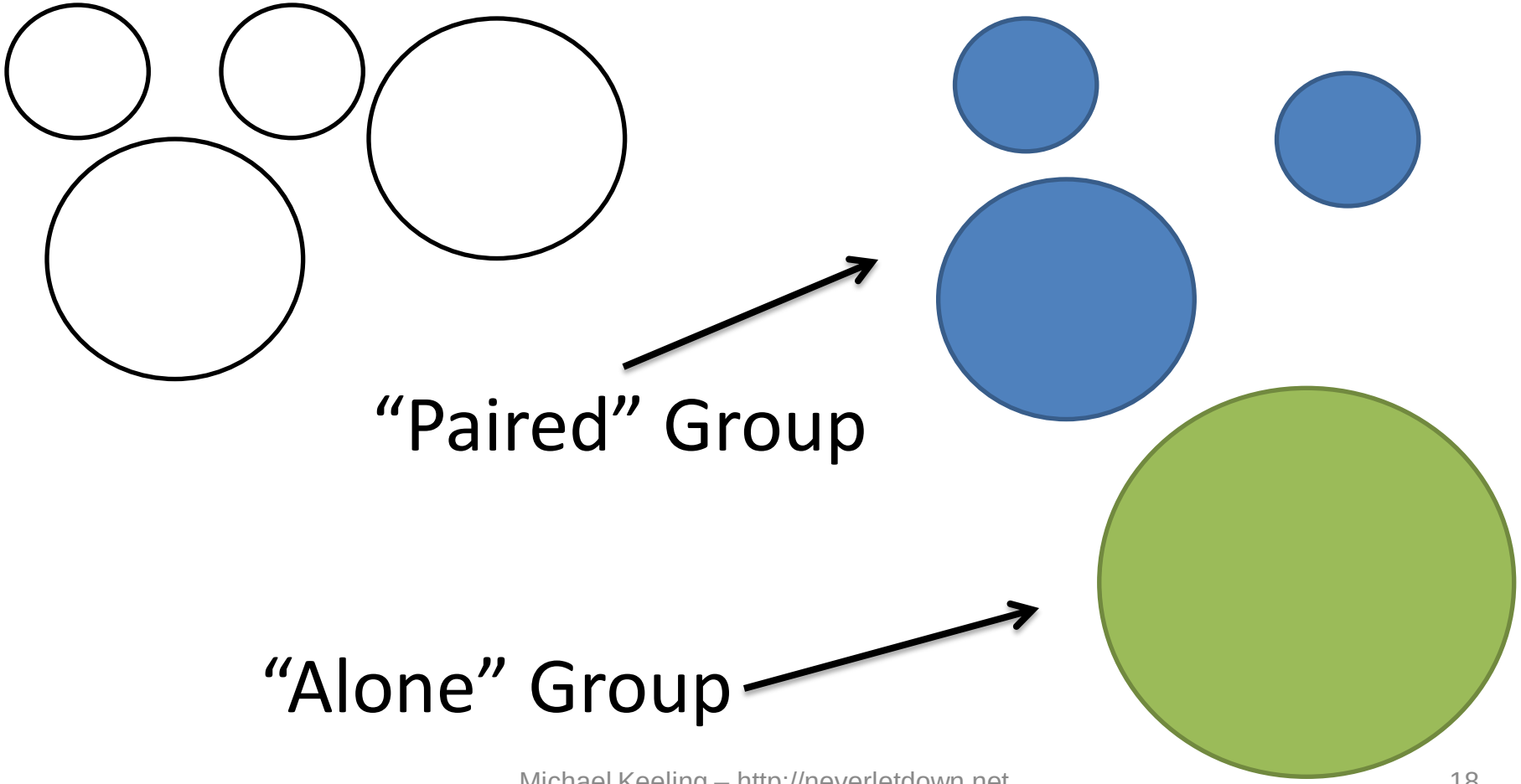


Iteration Backlog

Experiment Planning Process

Product Backlog

Iteration Backlog



WILLIAM & MARY
1

Lightweight Experiments

- Focus on a single question
- Divide work, not teams
- Releases!
- **Use the data you have**
- Statistical significance is overrated
- Retrospectives get immediate value

Metrics and Data

- Metrics
  - Effort paired vs. alone
  - Duration paired vs. alone
  - Escaped defects paired vs. alone (inspection)
  - Discovered issues paired vs. inspection
- Data collection
  - Leverage existing practices
  - Guidance from Team Software Process
  - Had to get creative…

Real-time Defects Tally Sheet

Programmer 1: Marco Estimate (hours): —

Programmer 2: Michael Estimate (hours): —

date: 7-16-2009 Start Time : 5:36 End Time: 7:39

Milestone/Feature: Create Project Task Points: ✓ (Low/Medium/High)

Task Description: Dialog Boxes

Data / Documentation	Syntax	Build / Package	Assignment	Checking	Interface	Function	System	Environment
Comments, coding standard/style	Things that don't compile	CM, file maintenance, package stuff	Declarations, duplicate names, scope	Business rule violations, verify inputs	I/O, Method calls and references	Logic, algorithms	Performance, Timing, Java, design	Compile, test, hot keys, support
	~~IIII~~		~~IIII~~	~~IIII~~	II	III		I

Instructions: Record defects in real-time according to the type of defect detected by marking the appropriate column. The current co-pilot should record the defects. This paper should be traded for the keyboard and mouse between the pair.

Tasks - CP: CODE UI

http://msesrv4d-vm.scs.ad.cs.cmu.edu:8282/Lists/Tasks/DispForm.aspx?

Square Root Team Site

Sign In

Square Root Team Site

This List

Home | Wiki

Square Root Team Site > Tasks > CP: CODE UI

Tasks: CP: CODE UI

The content of this item will be sent as an e-mail message to the person or group assigned to the item.

Close

Version History

Title	CP: CODE UI
Priority	(2) Normal
Status	Completed
Assigned To	Michael Keeling
Description	Created the create project summary and dialog box
Due Date	7/28/2009
Completion Date	7/17/2009
Original Estimate	2
Current Estimate	2
Elapsed	2
Summer Milestone	Develop_UC_Create Project
Iteration	9
Milestone	
Planned	No
Estimation Variance	0
Difficulty	Medium
Discipline	Coding
Co-pilot	Yes
Remaining	0

Content Type: Task
Version: 1.0
Created at 7/19/2009 1:59 PM by Marco Len
Last modified at 7/19/2009 1:59 PM by Marco Len

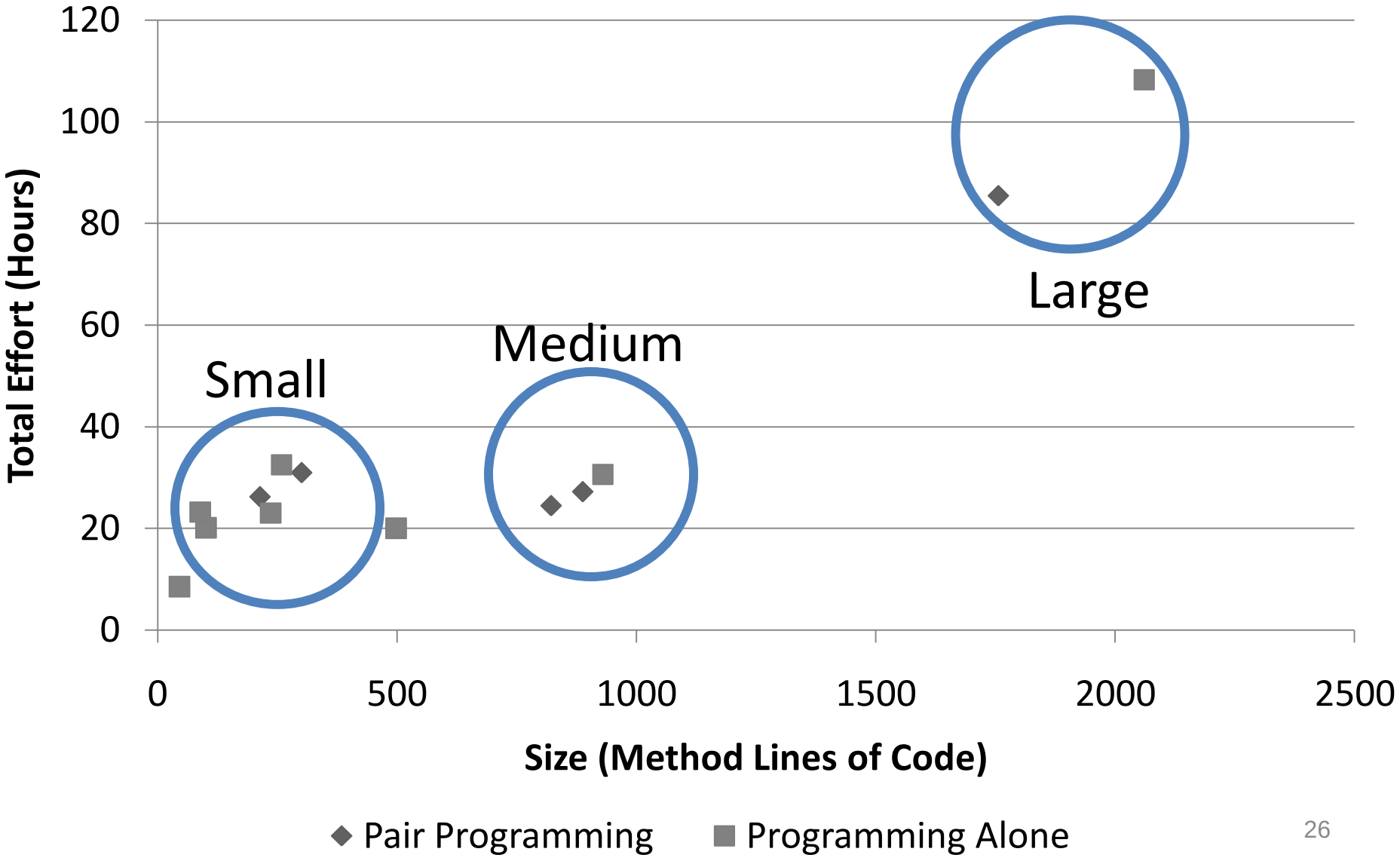
Close

Added "Co-Pilot" Field

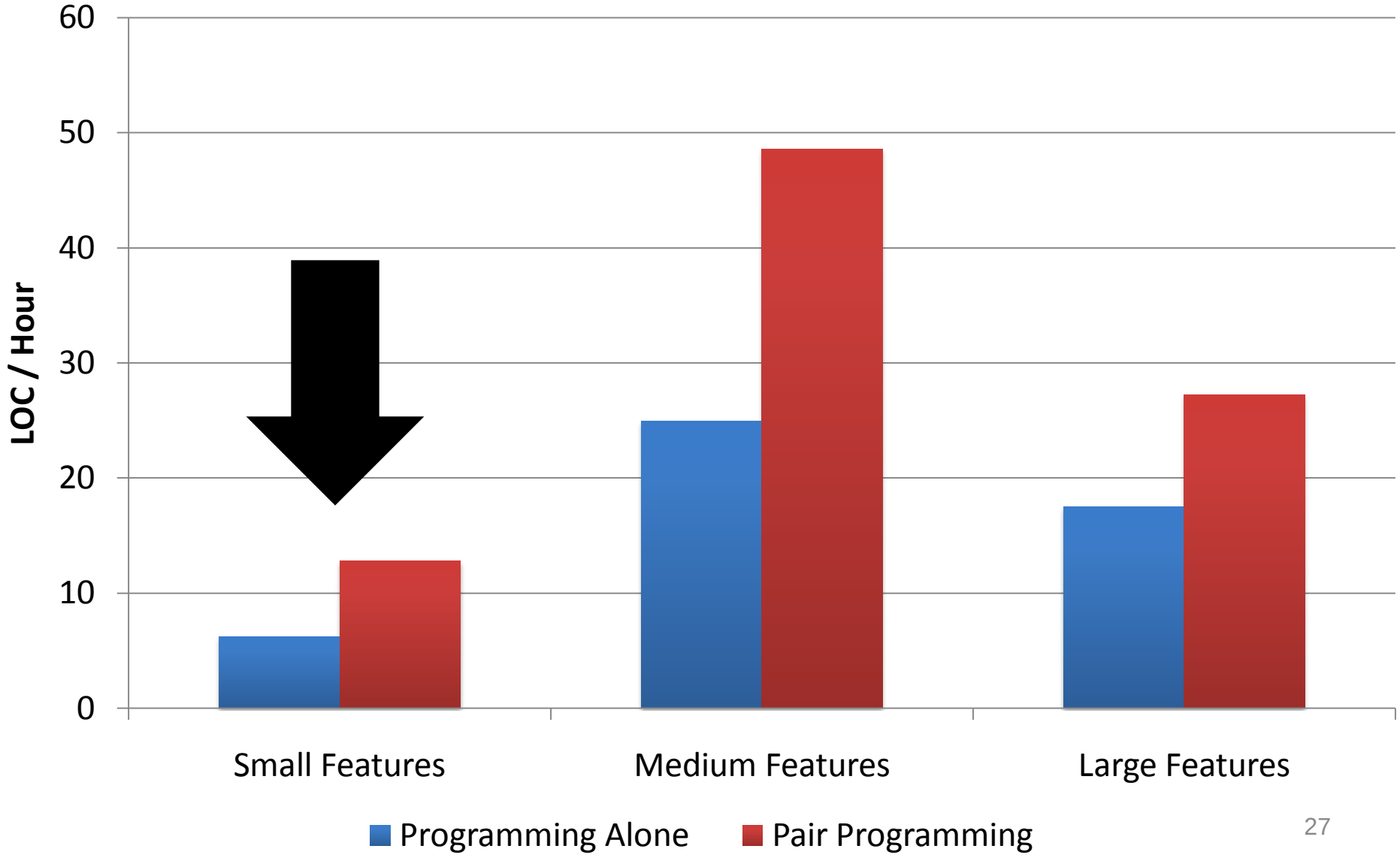
Lightweight Experiments

- Focus on a single question
- Divide work, not teams
- Releases!
- Use the data you have
- **Statistical significance is overrated**
- Retrospectives get immediate value

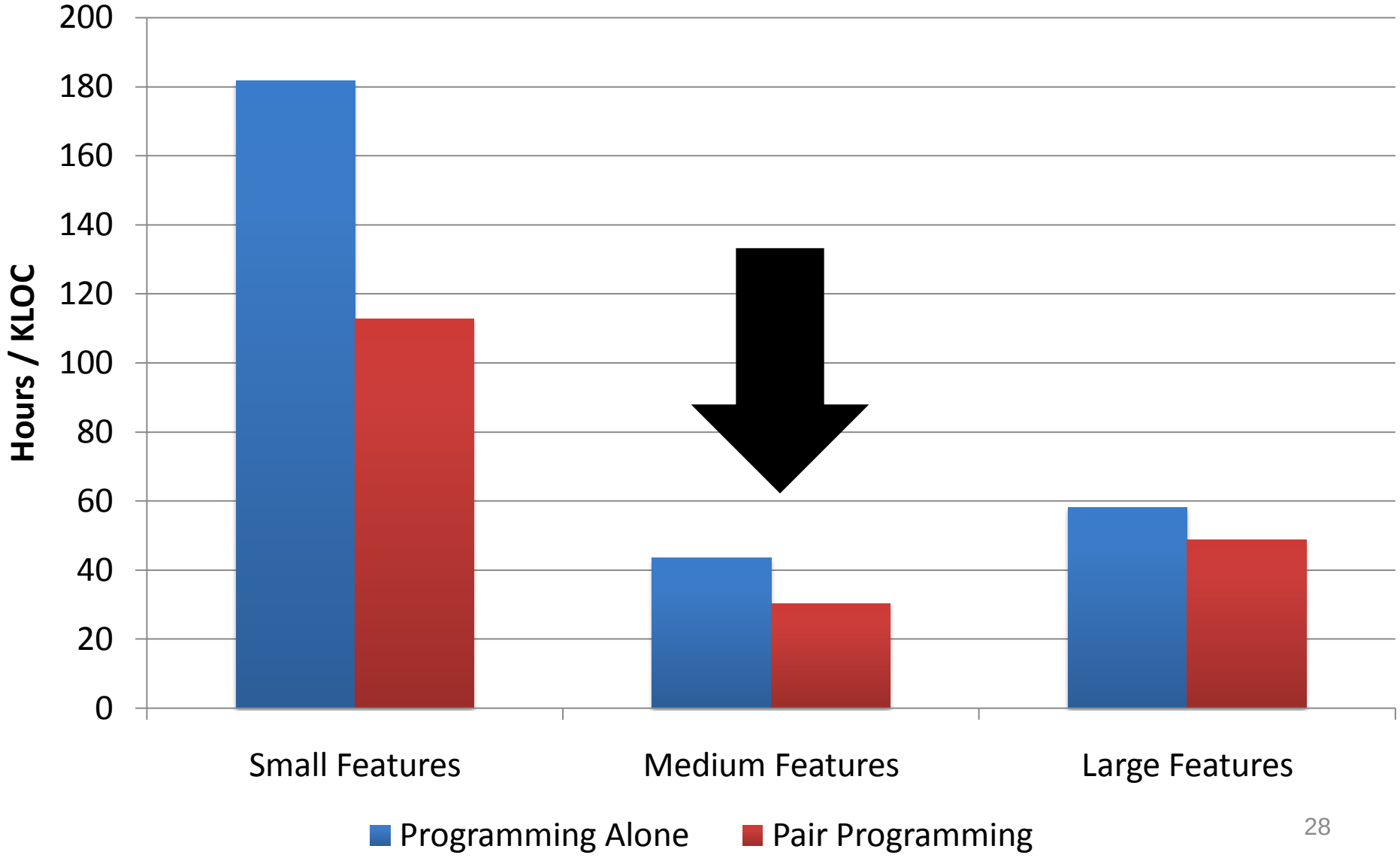
Effort vs. Actual Size of Features



Average Development Speed (Lower is Slower)



Average Development Cost (Lower is Cheaper)



Comparing Escaped Defects

Pair Programming	Programming Alone
More escaped defects / method LOC	Fewer escaped defects / method LOC
Higher Correlation (More predictable)	Lower Correlation (Less predictable)

Qualitative Observations

	Pair Programming	Programming Alone, Inspection
Primary Focus	Create working code	Find issues
Fixing Issues	Easy, issues in context	Frustrating, long turnaround time
Knowledge Sharing	Code-level details	System-level concerns

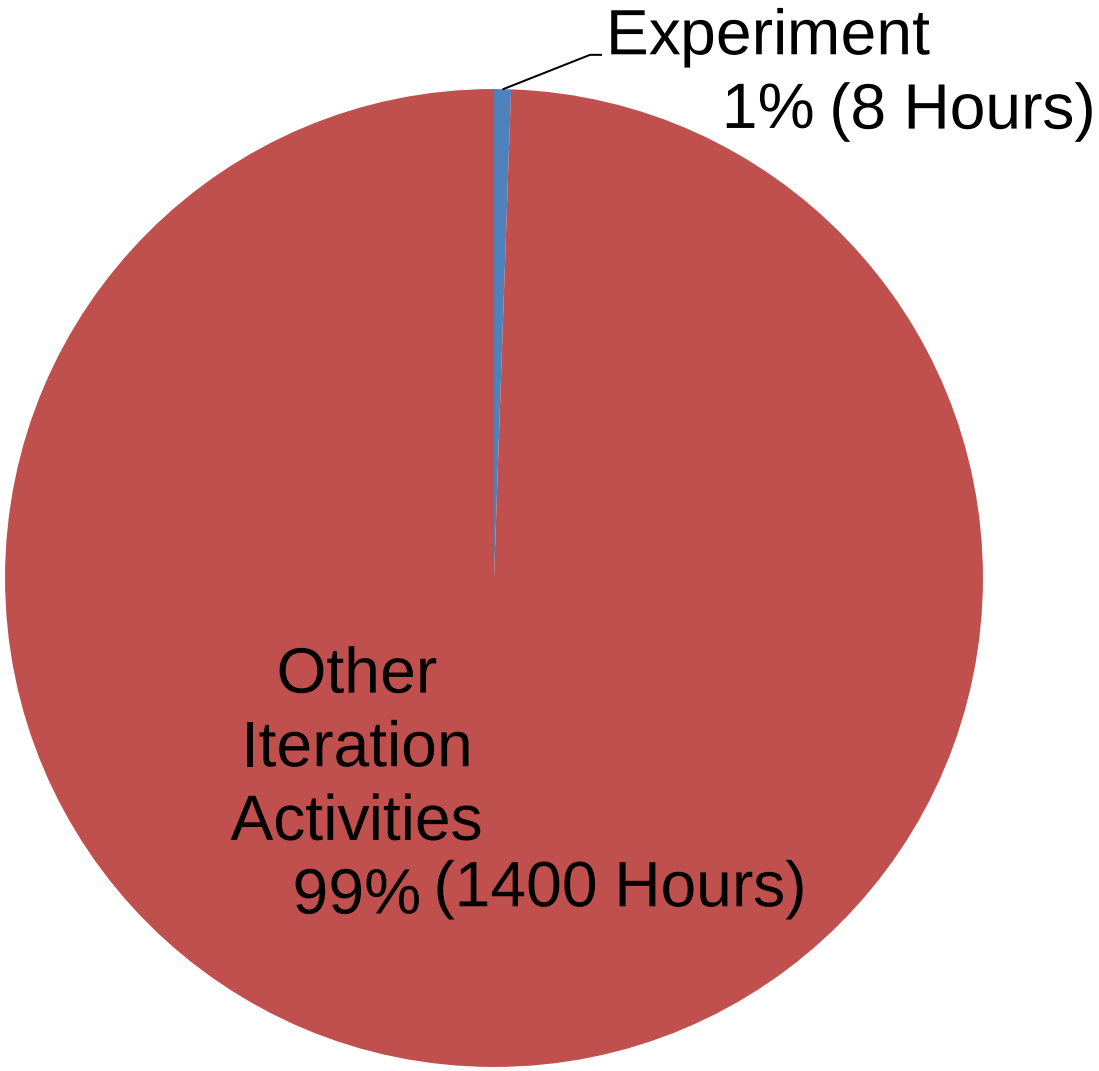
Lightweight Experiments

- Focus on a single question
- Divide work, not teams
- Releases!
- Use the data you have
- Statistical significance is overrated
- **Retrospectives get immediate value**

Stick with Pair Programming

Faster,
Cheaper,
More Predictable

Cost of the Experiment



3, Two Week Iterations

On Experimentation

What else can we learn?

- User stories vs. use cases
- Estimation techniques
- Unit testing vs. inspection
- Design practices
- More…?

Share what you learn and help grow our understanding of software engineering!

Thank you!

Full project archive available online:
http://dogbert.mse.cs.cmu.edu/MSE2009/Projects/SquareRoot/

Michael Keeling
mkeeling@neverletdown.net
http://neverletdown.net

References

Fagan, M,: Design and Code Inspections to Reduce Errors in Program Development. IBM Systems Journal 15, 182-211 (1976)

Williams, L., Kessler, R.R., Cunningham, W., Jefferies, R.: Strengthening the Case for Pair Programming. IEEE Software 26, 19-25 (2000)

Phongpaibul, M., Boehm, B.: A Replicate Empirical Comparison between Pair Development and Software Development with Inspection. In: First International Symposium on Empirical Software Engineering and Measurement, pp. 265-274. IEEE Computer Society, Washington, D.C. (2007)

Garlan, D., Gluch, D.P., Tomayko, J.E.: Agents of Change: Educating Software Engineering Leaders. IEEE Computer 30, 59{65 (1997)

Mead, N.R., Hough, E.D., Stehney, T.R.: Security Quality Requirements Engineering (SQUARE) Methodology. Technical report, Software Engineering Institute (2005)

Square Root Project Archive and Experiment Data Set (available indefinitely), http://dogbert.mse.cs.cmu.edu/MSE2009/Projects/SquareRoot/