Berlin 12-13 October 2017

Pair Programming:

Does it Really Work?

About me

Mirko Urru (🐦@MirkoUrru)

Software Engineer @DellEMC
Cork

Agenda

What is Pair Programming?

Why Pair Programming?

When is it working?

Cost of pair programming

1

What is Pair Programming?

“Pair programming is an Agile software development technique in which two programmers work together at one workstation. One, the ***driver****, writes code while the other, the observer or* ***navigator****, reviews each line of code as it is typed in. The two programmers switch roles frequently.”*

2

Why Pair programming?

Why Pair Programming?

Code Quality

Instant code review

Learning

Sharing knowledge

3

When is it working?

Rules for a good pair programming section

1. Agree on the **physical environment** beforehand
2. When talking about code, always **refer to line number and file name**
3. When **disagreeing**, talk in **terms of benefit**
4. **Switch role frequently** (a Tomato each)

When do pair programming

New hires

Complex tasks

Early stage project

Approaching new technologies

Best case scenario

Let me write this test first..

Sure, I'll write the method then… Why don't we use this library?

Oh cool, we can re-use it in this method later..

This class has too many methods, let me refactor it…

Perfect, all tests are still green …

Cool, do you know this tool to test code coverage?

Performance Indicators

- Satisfaction
- Problem solving
- Learning

No-Performance Indicator

- Lack of engagement
- Silence
- “Watch the Master”

!

Pair-programming is

NOT the right tool for every task

and it is NOT suitable for everyone

4

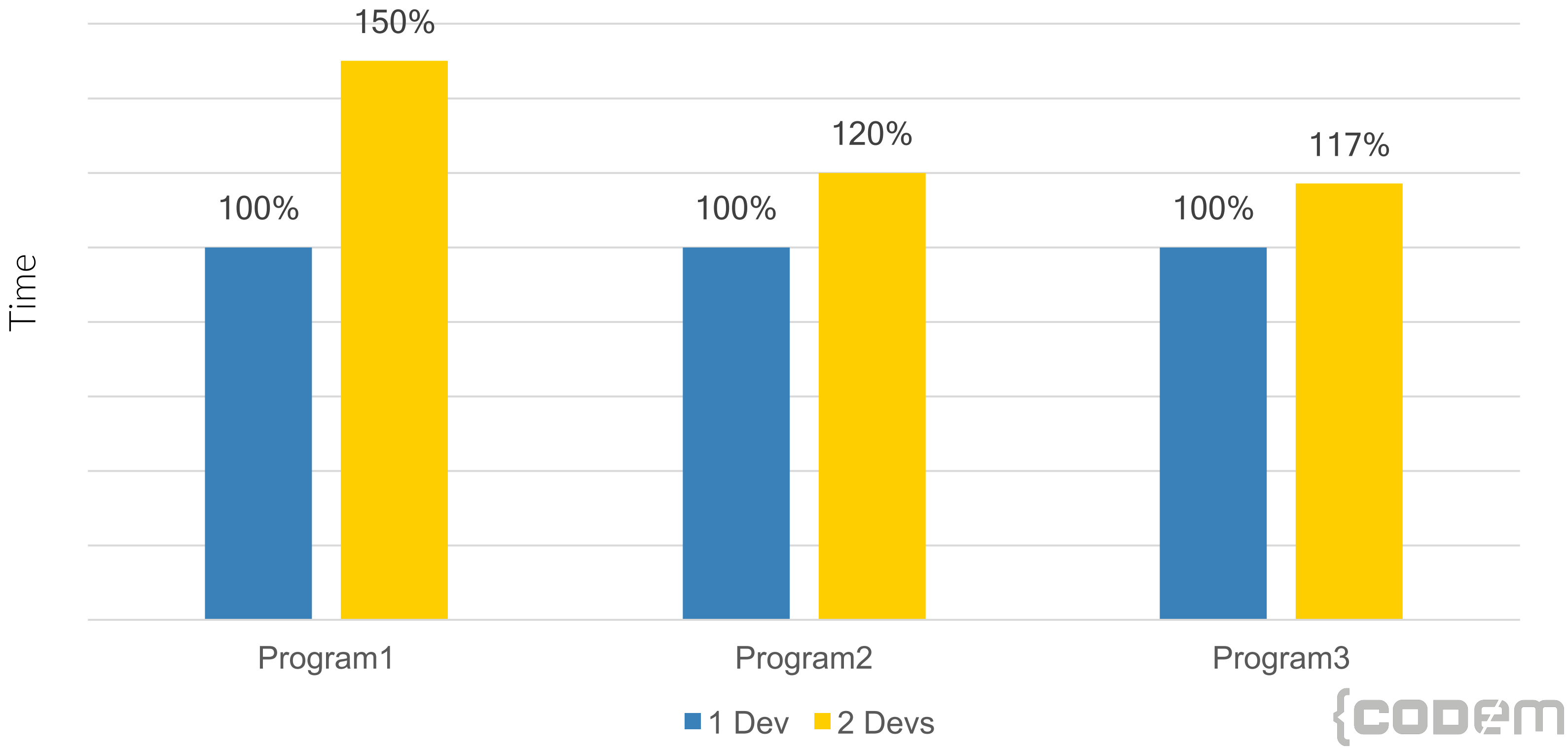
Costs of Pair Programming

15%

The development cost for these benefits is not the 100%, but is approximately 15%.

The Costs and Benefits of Pair Programming
https://collaboration.csc.ncsu.edu/laurie/Papers/XPSardinia.PDF

Relative Time: One Individual vs Two Collaborators



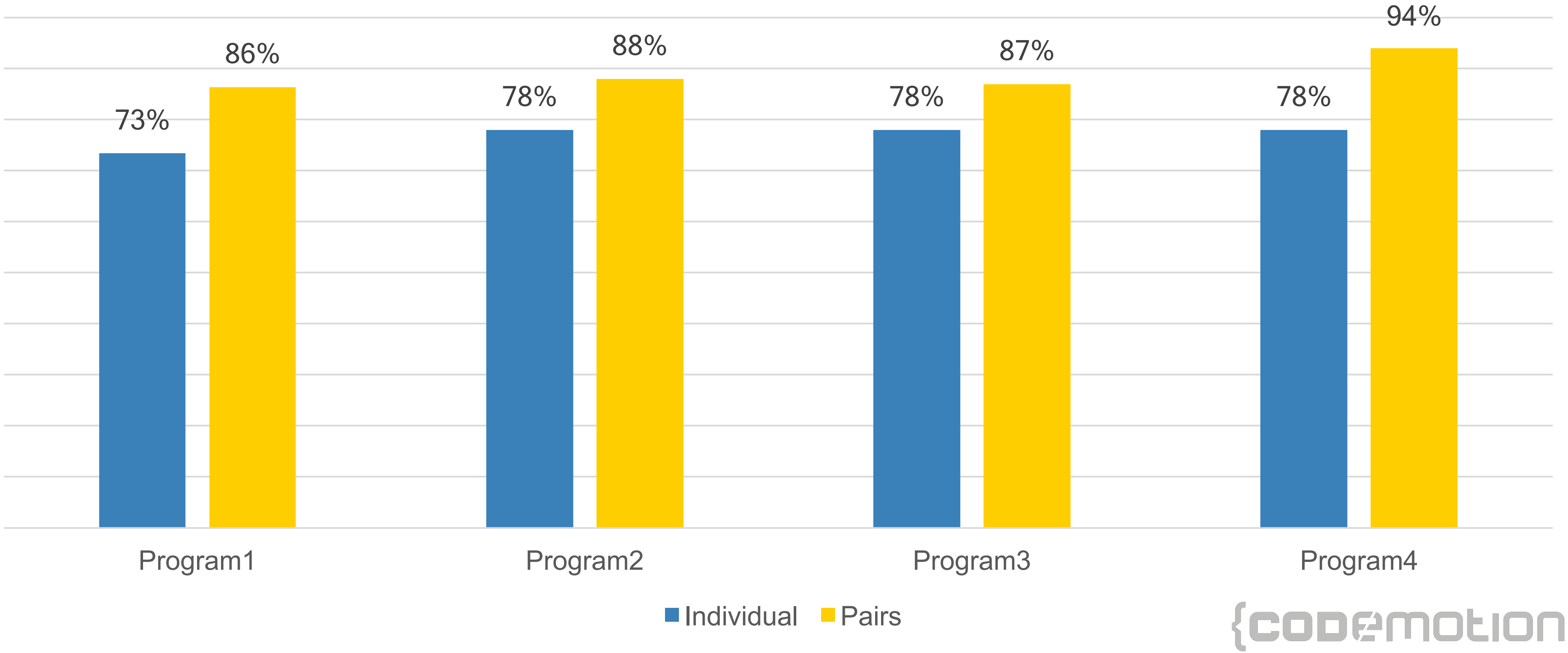
{codemotion}

15%

Less defects

The Costs and Benefits of Pair Programming
https://collaboration.csc.ncsu.edu/laurie/Papers/XPSardinia.PDF

Post Development Test Cases Passed



Experiment

- ❑ 50,000 lines of code (LOC)
- ❑ Speed of 50 LOC/hour
- ❑ Individual will take 1000 hours
- ❑ Pairs 15% longer so 1150 hours (+150 hours)

Experiment

Industry Average: errors per 1000 lines of delivered code

Industry Average: about 15 – 50 *(Clean Code)*

Microsoft: 10-20

Nasa: 0

Example

N°Defects per 1000 LOC * Thousand code lines

Individual:	30 * 50 = 1500
Pair:	25.5 * 50 = 1275

225 less defects

Single bugfix average time fixing: ***10 hours***

2250 hours > 150 extra hours for a pair section

Summary

- ✓ many mistakes get caught
- ✓ the final defect content is statistically lower (continuous code reviews)
- ✓ the designs are better and code is shorter (ongoing brainstorming and pair relaying);
- ✓ the team solves problems faster (pair relaying);
- ✓ people learn significantly more, about the system and about software development (line- of-sight learning);
- ✓ the project ends up with multiple people understanding each piece of the system;
- ✓ the people learn to work together and talk more often together, giving better information flow and team dynamics;

86%

Developers prefer to work alone

Forbes.com

Thanks!

Any questions ?

You can find me at

- @MirkoUrru
- Mirko.urru@me.com