Pair Programming Concepts and Considerations

Paul Daigle

DealerMatch

2013-Aug-20

Outline

Outline

What is Pair Programming?

Pair Programming

Pair Programming is the productivity practice of having two developers working at one computer.

What's the Point?

Pairing improves overall team productivity by improving focus and fostering an energized working environment.[6]

Why pair program?

1. To improve communication
2. To improve the team's capability
3. To improve productivity
4. To improve product quality

Outline

Improving Focus

Interuptions

In general, a pair is better able to return to work after an interuption than an individual[6].

Distractions

In one study, pairs were found to spend twice as much time using the IDE and half as much time surfing the web or checking email[7].

Improving Results

Reduction of Defects

When pairs work together well, they tend to produce code with fewer defects than single programmers[2].

Knowledge Sharing

Pairing is an effective way to get new programmers up to speed, especially when those programmers are novices to a skill or environment[3].

Total Velocity

Studies suggest that pairing either has little effect on velocity or increases overall velocity[4].

Outline

Objections from Management

Pairing reduces capacity

Although it's counterintuitive, the evidence doesn't support this. A single pair is as productive or more productive than two single programmers.

Objections from Developers I

Pairing is Difficult

Pairing *is* a skill, but not one that is difficult to aquire[5]. Participants in pairing tend to view the practice positively, even after initial resistance[1].

Process Problems

- Uncertainty about how to design in a pair
- Need to research privately
- Lack of infrastructure

Objections from Developers II

Personality Issues

- Toxic pairs
- Pair Marriage
- Mismatched skill level

Not if, but how

Mostly, these sorts of problems are not problems with pair programming, but with how a particular group manages and supports pair programming.

Outline

Bibliography I

A. Begel and N. Nagappan, "Pair programming: what's in it for me?" in *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*, ser. ESEM '08. New York, NY, USA: ACM, 2008, pp. 120–128. [Online]. Available: http://doi.acm.org/10.1145/1414004.1414026

G. Canfora, A. Cimitile, F. Garcia, M. Piattini, and C. A. Visaggio, "Evaluating performances of pair designing in industry," *Journal of Systems and Software*, vol. 80, no. 8, pp. 1317 – 1327, 2007, <ce:title>The Impact of Barry Boehm's Work on Software Engineering Education and Training</ce:title>. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121206003414

Bibliography II

I. Fronza, A. Sillitti, and G. Succi, “An interpretation of the results of the analysis of pair programming during novices integration in a team,” in *Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on*, 2009, pp. 225–235.

H. Hulkko and P. Abrahamsson, “A multiple case study on the impact of pair programming on product quality,” in *Software Engineering, 2005. ICSE 2005. Proceedings. 27th International Conference on*, 2005, pp. 495–504.

Bibliography III


E. Mendes, L. B. Al-Fakhri, and A. Luxton-Reilly, "Investigating pair-programming in a 2nd-year software development and design computer science course," *SIGCSE Bull.*, vol. 37, no. 3, pp. 296–300, Jun. 2005. [Online]. Available: http://doi.acm.org/10.1145/1151954.1067526


A. Sillitti, G. Succi, and J. Vlasenko, "Understanding the impact of pair programming on developers attention: A case study on a large industrial experimentation," in *Software Engineering (ICSE), 2012 34th International Conference on*, 2012, pp. 1094–1101.


L. Williams, R. Kessler, W. Cunningham, and R. Jeffries, "Strengthening the case for pair programming," *Software, IEEE*, vol. 17, no. 4, pp. 19–25, 2000.