plain concepts

Pair Programming: The Force Awakens

Ander Conal

24.07.20

VIRTUAL COFFEE

plain concepts

Ander Conal

SOFTWARE DEVELOPER

When I’m not working, you’d probably catch me boxing, walking with my dogs and family, playing with the Nintendo Switch or playing my old vynils from my era as DJ and event promotioner.

Actually I work at Plain Concepts Bilbao where I work programming, talking and educading.

@anderconal

plain concepts

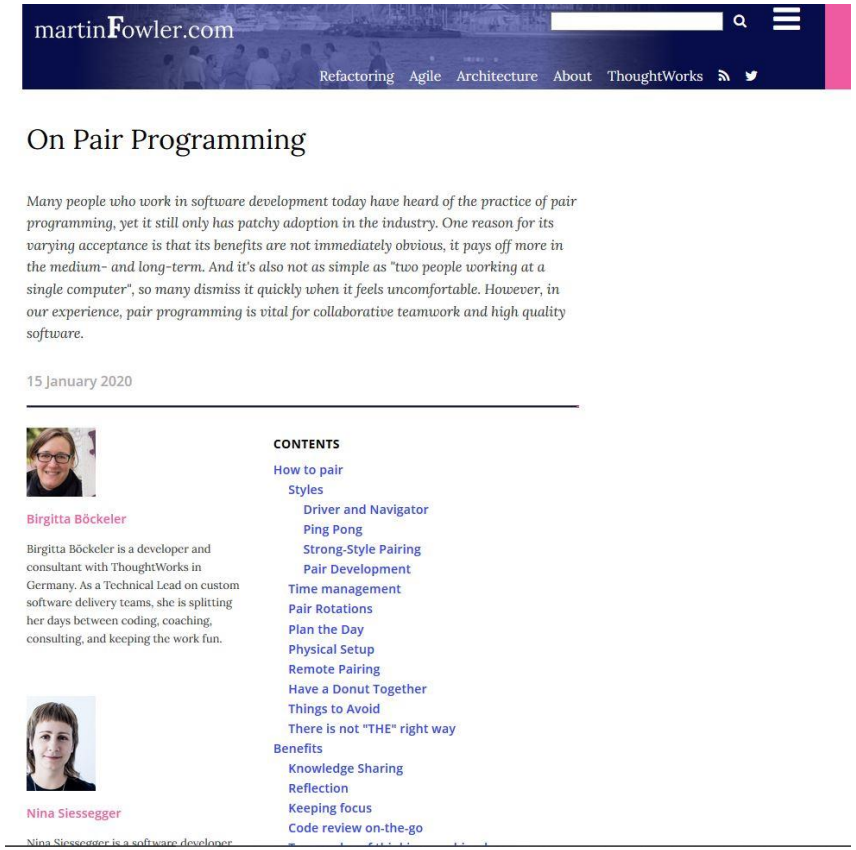
Schedule

1. What is Pair Programming?
2. Pair Programming Styles
3. The (numerous) benefits of Pair Programming
4. The (dangerous) dangers of Pair Programming
5. Pair Programming in the recruitment process
6. Last conclusions (hopefully with the force awakened)

On Pair Programming

martinFowler.com

- Birgitta Böckeler
- Nina Siessegger



plain concepts

What is Pair Programming?

plain concepts

What is Pair Programming?

- Is a **(good)** programming practice
- **Modest adoption** in the industry
- **2 people working in the same PC**... ¿and nothing more?
- One of the good practices recommended by **Extreme Programming (XP)**

plain concepts

Pair Programming Styles

plain concepts

Driver and Navigator

plain concepts

Driver & Navigator

- **Driver**
  - **The one who rides** (takes care of the keyboard, the mouse...)
  - **Focuses on what is being doing at the moment**, without worry about the big picture
  - **Comments what is doing** while is doing it
- **Navigator**
  - **The one who observes** while the driver is writing
  - **Reviews the code in live**, shares ideas and gives directions
  - **Takes care about the big picture**, next steps, bugs...

plain concepts

Driver & Navigator

- **2 different perspectives** about the code
- The **driver** will have a more **tactical** perspective (**details**)
- The **navigator** will have a more **strategic** perspective (**observing from outside**)

plain concepts

Driver & Navigator

- Start with a **well defined task**
- Set and focus on a **small goal each time**
- **Change keyboard and roles** regularly

- The **navigator** will **avoid tactical thinking** (**details** for the **driver**)
- The **navigator** will think in the **medium-long term**. Next steps, obstacles, ideas to argue about once the small goal have been reached....

plain concepts

Pair Programming Styles: Ping Pong

plain concepts

Ping Pong

- **Embraces** Test-Driven Development (**TDD**) a (good) practice of Extreme Programming (**XP**)
- **Ideal for really well defined tasks** which can be developed via **TDD**

plain concepts

Ping Pong

- **Ping**
  - Developer “A” writes a failing test
  - Developer “B” writes a new failing test
- **Pong**
  - Developer “B” writes the implementation and makes the test pass
  - Developer “A” writes the implementation and makes the test pass

plain concepts

Ping Pong

- Each **Pong can be accompanied by another good practive of** Extreme Programming (**XP**), **pair refactor** before continue with the next failing test
- **Red – Green – Refactor approach**
  - Write a **failing** test (red), make it pass with the **minimum** (green) and **refactor**

plain concepts

Strong-Style Pairing

plain concepts

Strong-Style Pairing

- Perfect for the **knowledge transfer**
- **Follows a rule**:
  - “For an idea to go from your brain to the computer, it must do it using the hands of other person”

plain concepts

Strong-Style Pairing

- The **navigator** is normally the **most familiar person with** the **technology**, the **task** or the **business**
- The **driver** is normally **the one with less experience** (with the code, with the tools, with the business...)

plain concepts

Strong-Style Pairing

- The **driver trusts** the **navigator**. Should not be upset because of the lack of understanding of some parts of the session
- Useful for
  - Onboarding
  - Knowledge transfer
- Don't overuse
- The idea is to change roles in the short term

plain concepts

Pair Development

plain concepts

Pair Development

- Not a technique but **a way of thinking**

- To carry out a user story, it is not only necessary to program
- As peer, **you are responsible of all the needed stuff** to finish the task

plain concepts

Pair Development

- Examples
  - Understand the problem
  - Find a solution
  - Plan your approach
- Examples
  - Investigate and explore
  - Document

plain concepts

The (numerous) benefits of Pair Programming

plain concepts

Knowledge transfer

- **Prevents knowledge silos**
- **2 minds understanding** and **discussing** a problem, increase the probability of find a good solution
- **Different experiences** and **perspectives** allow us to consider more **alternatives**

plain concepts

Reflection

- With Pair Programming
  - We are forced to **discus solutions**
  - We force ourselves to **explain things** better and better, helping us discover if **we have really understood the task** well, or if our solution is the right one
  - **Not only applicable to the code**, but to the user stories

plain concepts

Focus

- Is **easier to focus** when you are not alone
- Each part of the peer have to communicate, **say what is doing and why**!
- If the “why” is not on the way of the main goal, the peer can help **focusing again**

plain concepts

Pair review in live

- **4 eyes** instead of 2 **reviewing each implementation detail**
- **Refactoring** is a part of Pair Programming. If something is wrong, we **change it on the fly**

plain concepts

2 ways of thinking, combined

- **Tactic** + **strategic** brain
- Could an individual combine both?
- **Unify both ways of thinking will increase code quality**. You'll be able to set focus on the details (tactic) and have a global vision (strategic) at the same time!

plain concepts

Collective Code Ownership

- Continuously practice of Pair Programming ensures **each line of code have been touched by, at less, 2 people**
- Facilitates each member of the team to **be comfortable changing anything in the code**

plain concepts

Maintains a low WIP (Work In Progress)

- Doing Pair Programming in a 4 people team who all do Pair Programming, helps limiting the WIP from 4 to 2
- **More focus on the important tasks**, with more code quality

plain concepts

Fast onboarding

- **Minimizes the impact of the onboarding of new employees** by forcing people to communicate much more

plain concepts

The (dangerous) dangers of Pair Programming

plain concepts

Pair Programming can be exhausting

- When working alone, you take breaks when you want
- Doing Pair Programming forces you to **stay focused for longer** and **find breaks that match the couple's way of thinking**, their rhythm, etc.
- It can be very **intense** and **exhausting**

plain concepts

Interruptions due to meetings

- If with one person it can be crazy, imagine putting together the calendars of 2 people who have meetings in different places and at different times

plain concepts

Different levels

- Can take for granted how far each can go
- May cause **frustration** due to **difference in rhythm**

plain concepts

Doing Pair Programming with a lot of uncertainty

- When working with a new technology that you both use for the first time, or trying a new paradigm, there may be problems
- **Experimenting and searching can be frustrating**, because **each person learns at a different pace and in a different way**

plain concepts

You don't have time for yourself

- **Each one has their habits** ... listening to music, stopping to learn something new X minutes a day ... with Pair Programming this is complicated ...

plain concepts

Context changes

- If you are working on task X, you pair with task Y and the next day you need to pair for task Z, there may be too many context changes

plain concepts

Pair Programming requires showing your vulnerabilities

- **Pair Programming forces us to share** everything we know, but also **what we don't know**
- It can be **hard to say you don't know something** or to be insecure about a decision
- Imposter syndrome, 10x engineer and others don't help

plain concepts

It costs that people who have never done it accept it

- Every time it goes down, but there are still mentalities in different positions, from the lowest to the highest, in which Pair Programming does not fit
- It is usually seen as a waste of time and therefore money

plain concepts

Pair Programming in the recruitment process

plain concepts

Pair Programming in the recruitment process

- Increasingly common as a **second or third step of a job interview** for the software industry
- Like the practice itself, it has its advantages and its disadvantages. **If not planned well you can make this step useless**

plain concepts

Examples and casuistry in the recruitment process

- **The interviewee has never done Pair Programming**
  - Strong-Style Pairing
  - Pair Development
- **The interviewee has little experience with language / technology**
  - Strong-Style Pairing
  - Pair Development

plain concepts

Examples and casuistry in the recruitment process

- **The interviewee has experience with TDD and with language / technology**
  - Ping Pong
  - Pair Development
- **The interviewee has experience with language / technology but not with TDD**
  - Driver and Navigator
  - Pair Development

plain concepts

Recruitment process conclusions

- The **easiest** style to introduce in any case is **Pair Development**
- In addition, **Pair Development will allow you to see beyond the code**
- It is necessary to **explain well what style** we are going to use and **what role** each one is going to play
- Do not base your decision 100% on the result of this step. **Nerves and other factors play tricks**

plain concepts

Recruitment process conclusions

- Tech Sector Job Interviews **Assess Anxiety, Not Software Skills** (https://news.ncsu.edu/2020/07/tech-job-interviews-anxiety)
- Does Stress Impact Technical Interview Performance? (http://chrisparnin.me/pdf/stress_FSE_20.pdf)
- Depending on the person you are interviewing, **Pair Programming could cause anxiety and stress** because they're not used to working on a whiteboard in front of an audience

plain concepts

Last conclusions (hopefully with the force awakened)

plain concepts

Pair Programming is fine

- Well used, it is very powerful and **gives us many benefits**
- Now that we know the (dangerous) dangers, we are closer to avoiding them and **staying with only the benefits**
- Always **use it in combination with other** good **practices** and good **planning**

plain concepts

Pair Programming is fine

- **Do not stick to the letter** with the styles or the way to do it
- **Combine** various styles
- **Try them all separately** and choose the one or the ones you like the most
- **Invent a new one** and tell us about it

Thank you

@anderconal

@plainconcepts

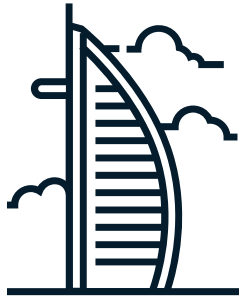
www.plainconcepts.com

plain concepts

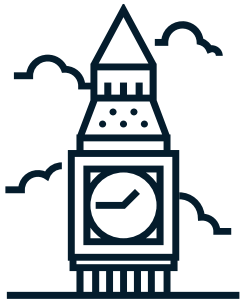
Rediscover
the meaning of technology

SPAIN

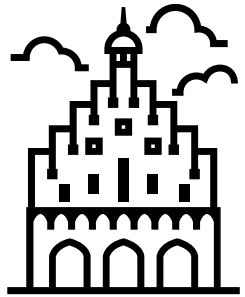
USA



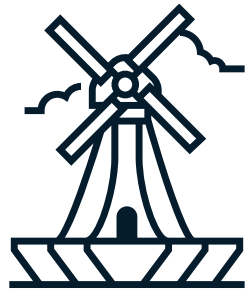
UAE



UNITED KINGDOM



GERMANY



NETHERLANDS

www.plainconcepts.com

For further information

info@plainconcepts.com