plain concepts

Pair Programming: El despertar de la fuerza

Ander Conal

04.06.20

VIRTUAL COFFEE

plain concepts

Ander Conal

SOFTWARE DEVELOPER

Cuando no estoy trabajando, lo más posible es que me encuentres haciendo boxeo, paseando con mis 2 perros y el resto de mi familia, jugando un poco a la Switch o quitando el polvo a los vinilos de mi época como promotor de eventos y DJ. En la actualidad, trabajo en Plain Concepts Bilbao donde, además de programar, doy charlas y formaciones.

@anderconal

plain concepts

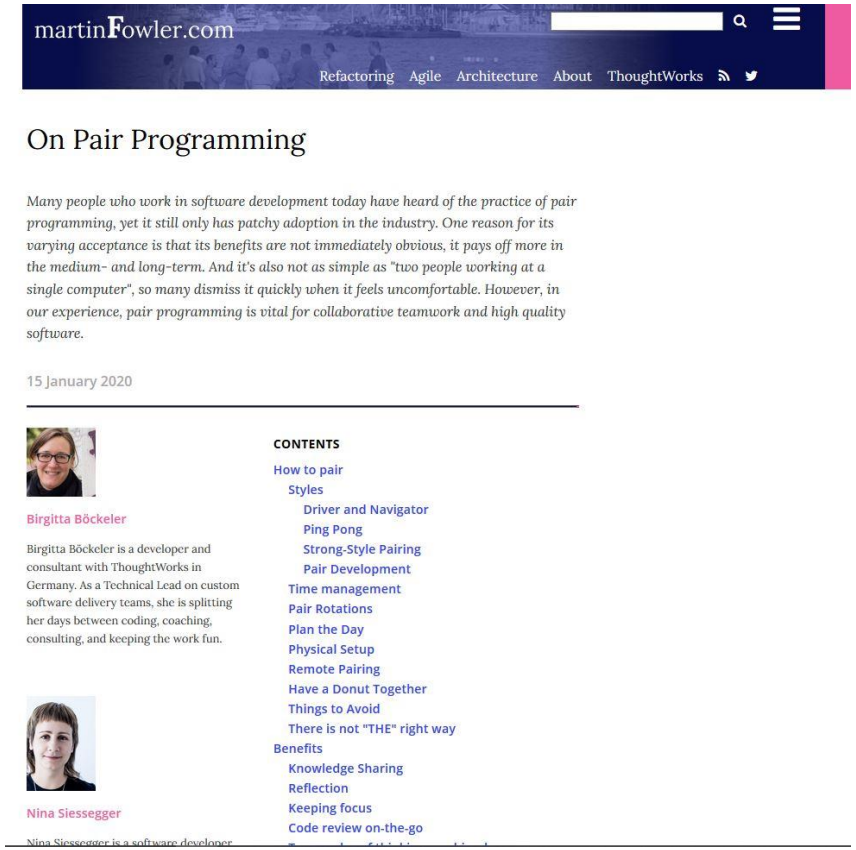
Agenda completa

1. ¿Qué es Pair Programming?
2. Estilos de Pair Programming
3. Los (numerosos) beneficios del Pair Programming
4. Los (peligrosos) peligros del Pair Programming
5. Pair Programming en el proceso de reclutamiento
6. Conclusiones (esperemos que con la fuerza despertada)

On Pair Programming

martinFowler.com

- Birgitta Böckeler
- Nina Siessegger



plain concepts

¿Qué es Pair Programming?

plain concepts

¿Qué es el Pair Programming?

- Es una **(buena)** práctica de programación
- **Modesta adopción** de la misma en la industria
- 2 personas trabajando en un único PC… ¿y ya está?
- Una de las buenas prácticas de desarrollo recomendadas en Extreme Programming (XP)

plain concepts

Estilos de Pair Programming

plain concepts

Driver and Navigator

plain concepts

Conductor y navegante (Driver & Navigator)

- **Conductor**
  - La persona que **lleva las riendas** (teclado, ratón...)
  - **Enfocada en lo que está hacienda en el momento**, sin preocuparse por problemas mayores
  - **Debe comentar siempre lo que hace** mientras lo hace
- **Navegante**
  - **La persona que observa** mientras el conductor escribe
  - **Revisa el código** en directo, comparte ideas y da indicaciones
  - **Se preocupa por los problemas mayores**, siguientes pasos a dar, bugs...

plain concepts

Conductor y navegante (Driver & Navigator)

- Se consigue tener **2 perspectivas diferentes** respecto al código
- El **conductor** tendrá una **perspectiva** más **táctica** (detalles)
- El **navegante** tendrá una **perspectiva** más **estratégica** (observando “desde fuera”)

plain concepts

Conductor y navegante (Driver & Navigator)

- Empezar con una **tarea bien definida**
- Acordar **un pequeño objetivo** cada vez
- **Cambiar** teclado y roles regularmente

- El **navegante evitará** el **pensamiento táctico** (detalles para el conductor)
- El **navegante pensará a medio-largo plazo**, próximos pasos, potenciales obstáculos, ideas a discutir una vez el pequeño objetivo se ha alcanzado...

plain concepts

Estilos de Pair Programming: Ping Pong

plain concepts

Ping Pong

- **Abraza** el Test-Driven Development (**TDD**) y, por tanto, una (buena) práctica más del Extreme Programming (**XP**)
- Perfecto para cuando temenos una **tarea bien definida** que puede ser **implementada con TDD**

plain concepts

Ping Pong

- **Ping**
  - Desarrollador “A” escribe un test que falla
  - Developer “B” escribe un nuevo test que falla
- **Pong**
  - Desarrollador “B” escribe la implementación y lo hace pasar
  - Desarrollador “A” escribe la implementación y lo hace pasar

plain concepts

Ping Pong

- Cada **Pong puede ir acompañado de** otra (buena) práctica de Extreme Programming (**XP**), **refactorizar entre los 2** antes de seguir con el siguiente test que falla
- **Red – Green – Refactor approach**
  - Escribe un test que falle (rojo), haz que pase con los mínimos (verde) y refactoriza

plain concepts

Strong-Style Pairing

plain concepts

Strong-Style Pairing

- Perfecta para la **transferencia de conocimiento**

- **Sigue una norma**:
  - “Para que una idea vaya de tu cabeza al ordenador, tiene que ser mediante las manos de otra persona”

plain concepts

Strong-Style Pairing

- El **navegante** normalmente es la **persona** más **familiarizada con** la **tarea** o la **tecnología**
- El **conductor** normalmente la **persona con menos experiencia** (con el lenguaje, las herramientas, la base de código...)

plain concepts

Strong-Style Pairing

- El **conductor se fía 100 % del navegante** y no tiene que estar molesto por no entenderlo todo al 100 %

- Útil para
  - Onboarding
  - Transferencia de conocimiento inicial
- No sobreutilizar
- La idea es poder cambiar los roles a corto plazo

plain concepts

Pair Development

plain concepts

Pair Development

- Más que una técnica es una forma de pensar

- Para sacar adelante una historia de usuario, no solo hace falta programar
- Como pareja, eres responsible de todo lo necesario para completer la tarea

plain concepts

Pair Development

- Ejemplos
  - Entender el problema
  - Encontrar una solución
  - Planificar tu enfoque
- Ejemplos
  - Investigar y explorar
  - Documentar

plain concepts

Los (numerosos) beneficios del Pair Programming

plain concepts

Transferencia de conocimiento

- **Previene** los **silos de conocimiento**
- 2 mentes entendiendo y duscutiendo un problema, **aumentan las probabilidades de encontrar una solución buena**
- Diferentes experiencias y perspectivas nos empujan a **considerar más alternativas**

plain concepts

Reflexión

- Con Pair Programming
  - Nos vemos forzados a **discutir soluciones**
  - Nos obligamos a **explicar más y mejor las cosas**, ayudándonos a descubrir si realmente hemos entendido bien la tarea, o si nuestra solución es la correcta
  - **Aplicable no solo a código**, si no también a historias de usuario

plain concepts

Mantener el foco

- Es **más sencillo** mantener el foco **cuando no estás solo**
- Cada parte de la pareja tiene que **comunicarse**, decir **con qué está y por qué**
- Si el por qué se desvía del objetivo, el compañero puede ayudar a **reconducir**

plain concepts

Revisión de código sobre la marcha

- **4 ojos** en lugar de 2 **revisando** cada detalle de la implementación
- La **refactorización** es parte del Pair Programming. Si sale algo, **se cambia sobre la marcha**

plain concepts

2 formas de pensar, combinadas

- Cerebro **táctico** + cerebro **estratégico**
- ¿Podría una sola persona combinar ambos modos de pensar?
- **Unir ambas formas de pensar mejorará la calidad de tu código**. Podrás atender a los detalles (táctico) y tener una vision global (estratégico) al mismo tiempo

plain concepts

Collective Code Ownership

- Hacer Pair Programming de forma continuada asegura que **cada línea de código ha sido tocada, al menos, por 2 personas**
- Facilita que **cualquiera** en el equipo se sienta **cómodo hacienda cambios** en casi cualquier sitio del **código**

plain concepts

Mantiene un bajo WIP (work in progress)

- Hacer Pair Programming en un equipo de 4 personas donde todas hacen Pair Programming, ayuda a limitar el WIP de 4 a 2
- **Más foco en las tareas realmente importantes**, con potencialmente mayor calidad de código

plain concepts

Onboarding rápido

- **Minimiza el impacto de las nuevas incorporaciones** porque fuerza a las personas a comunicarse mucho más que

plain concepts

Los (peligrosos) peligros del Pair Programming

plain concepts

Hacer Pair Programming puede ser agotador

- Al trabajar **solo**, **descansas cuando quieres**
- Hacer **Pair Programming** te fuerza a mantener el foco durante más tiempo y **encontrar descansos que coincidan con** la forma de pensar de la **pareja**, su ritmo, etcetera
- Puede ser muy **intenso y agotador**

plain concepts

Interrupciones por reuniones

- Si con una sola persona puede ser una locura, imaginad juntar los calendarios de 2 personas que tienen reunions en lugares y a horas distintas

plain concepts

Diferentes niveles

- Puede llevar a **dar por hecho hasta dónde puede llegar cada uno**
- Puede producir **frustación** debido a la **diferencia de ritmo**

plain concepts

Hacer Pair Programming con mucha incertidumbre

- Al trabajar con una **nueva tecnología** que ambos usáis por primera vez, o probar un **nuevo paradigma**, **puede haber problemas**
- Experimentar y buscar puede ser frustrante, porque **cada persona aprende a un ritmo y de una forma distinta**

plain concepts

No tienes tiempo para ti mismo

- **Cada uno tiene sus costumbres**... escuchar música, parar para aprender algo nuevo X minutos al día... con el Pair Programming esto se complica...

plain concepts

Cambios de contexto

- Si estás trabajando en una tarea X, te pones a hacer pair para la tarea Y y al día siguiente necesitas hacer pair para la tarea Z, puede que haya **demasiados cambios de contexto**

plain concepts

El Pair Programming requiere mostrar tus vulnerabilidades

- El **Pair Programming nos fuerza a compartir** todo lo que sabemos, pero también **lo que no sabemos**
- Puede ser duro **decir que no sabes algo o mostrarte inseguro** ante una decision
- Síndrome del impostor, el 10x engineer y demás no ayudan

plain concepts

Cuesta que la gente que nunca lo ha hecho lo acepte

- Cada vez va a menos, pero sigue habiendo mentalidades en diferentes puestos, desde los más bajos a los más altos, en las que el Pair Programming no cabe
- Se suele ver como una **pérdida de tiempo** y, por tanto, de **dinero**

plain concepts

Pair Programming en el proceso de reclutamiento

plain concepts

Pair Programming en el proceso de reclutamiento

- **Cada vez más común** como segundo o tercer paso de una entrevista de trabajo para la industria del software
- Como la práctica en sí, tiene sus ventajas y sus desventajas. **Si no se planifica bien puede hacer que este paso no sirva de nada**

plain concepts

Ejemplos y casuísticas en el proceso de reclutamiento

- **El entrevistado nunca ha hecho Pair Programming**
  - Strong-Style Pairing
  - Pair Development
- **El entrevistado tiene poca experiencia con el lenguaje / tecnología**
  - Strong-Style Pairing
  - Pair Development

plain concepts

Ejemplos y casuísticas en el proceso de reclutamiento

- **El entrevistado tiene experiencia con TDD y con el lenguaje / tecnología**
  - Ping Pong
  - Pair Development
- **El entrevistado tiene experiencia con el lenguaje / tecnología pero no con TDD**
  - Driver and Navigator
  - Pair Development

plain concepts

Conclusiones proceso de reclutamiento

- El **estilo más sencillo** de introducir en cualquier caso es el **Pair Development**
- Además, el Pair Development **te permitirá ver más allá del código**
- Hay que **explicar bien qué estilo vamos a utilizar y qué papel** va a jugar cada uno
- No bases tu decisión al 100 % en el resultado de este paso. **Los nervios y otros factores juegan malas pasadas**

plain concepts

Conclusiones (esperemos que con la fuerza despertada)

plain concepts

Pair Programming es bien

- Bien utilizado, es **muy potente y nos aporta muchos beneficios**
- Ahora que sabemos los (peligrosos) peligros, estamos más cerca de evitarlos y quedarnos solo con los beneficios
- Utilízalo siempre en **combinación con el resto de buenas prácticas** y una **buena planificación**

plain concepts

Pair Programming es bien

- **No te quedes al pie de la letra** con los estilos ni la forma de hacerlo
- **Combina** varios estilos
- **Prueba** todos por separado y elige el o los que más te gusten
- **Inventa** uno nuevo y cuéntanoslo

Thank you

@anderconal

@plainconcepts

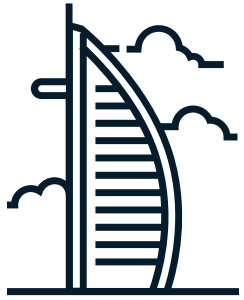
www.plainconcepts.com

plain concepts

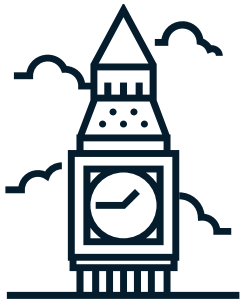
Rediscover
the meaning of technology

SPAIN

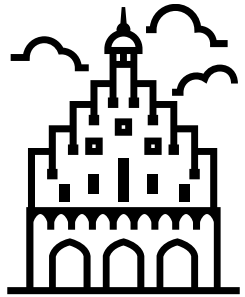
USA



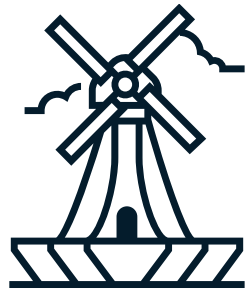
UAE



UNITED KINGDOM



GERMANY



NETHERLANDS

www.plainconcepts.com

For further information

info@plainconcepts.com