CS 4530: Fundamentals of Software Engineering
Lesson 4.4: Pair Programming

Apurva Saini and Rob Simmons

Khoury College of Computer Sciences

Pair Programming is a Knowledge Sharing Activity

- Two programmers work together at one computer, one “driving,” one “navigating”
- Survey of professional programmers (2001):
  - 90% “enjoyed collaborative programming more than solo programming”
  - 95% were “more confident in their solutions” when pair programmed
  - Provides long-term benefits: reduces defects by 15%, code size by 15%
  - Increases costs by 15% to 100% compared to single developer on the task

Cockburn and Williams. The Costs and Benefits of Pair Programming, (In: Extreme Programming Explained 2001)

Roles in Pair Programming

- Driver
    - Types the code
    - Focused on immediate task
- Navigator
    - Reviews each line of code
    - Spots errors and suggests improvements
- How does it help:
    - Improves code quality
    - Encourages knowledge sharing
    - Reduces bugs early
    - Improves team communication

When to use Pair Programming

- **Complex problems**: Two minds can break down and solve difficult logic more efficiently, catching edge cases early.
- **Learning new technologies**: One person may have experience, and the other can learn by doing and observing.
- **Code reviews in real time**: Pairing acts like a continuous code review, allowing for cleaner, more robust code from the start.
- **Mentorship**: Great for onboarding new team members—pairing allows them to learn the system while actively contributing.
- **Critical code paths**: Important features (e.g., payment logic, auth systems) benefit from the extra scrutiny and collaboration.

Common Pair Programming Styles

- **Ping Pong pairing**: Switch roles with each tests
- **Strong style pairing**: Driver only writes code as directed by the navigator
- **Tour Guide**: One that is familiar with the code guides another
- When not to pair:
  - Simple or repetitive tasks
  - Tasks requiring long research or reading
  - When you need deep focus
- How to pair effectively:
  - Communicate clearly and frequently
  - Take breaks
  - Switch roles effectively (every 20-30 min)
  - Use proper tools (Screen Share, live share, etc)

Pair Programming Improves Tool Diffusion

- Peer observation and recommendation shown to be more effective at discovering new tools than other knowledge sharing approaches
- Examples: Hot keys, especially for CLI; IDE tricks
- Most common in 2011 survey: "Open Type" feature in Eclipse, developer tools in web browser

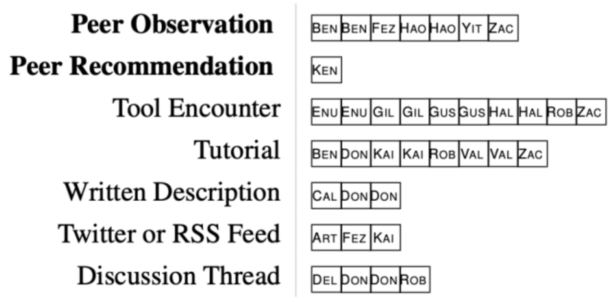

Figure 2: Histogram of the most frequent discovery modes.

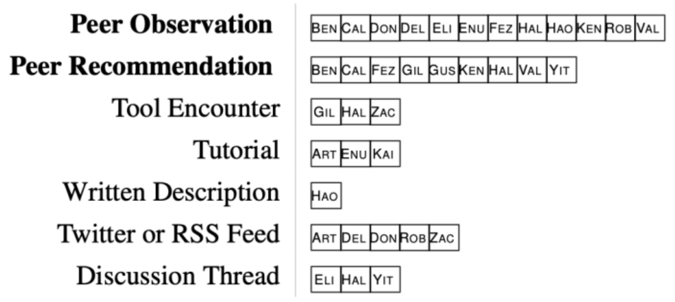

Figure 3: Histogram of the most effective discovery modes.

"Peer interaction effectively, yet infrequently, enables programmers to discover new tools", Emerson Murphy-Hill & Gail C. Murphy, CSCW 2011