Michael Johnson
Bro Parsons
CIT 380

PAIRED PROGRAMMING

When it comes to agile methodology there my preferred method to accomplish a software project would be paired programming. “Agile teams, committed to frequent, regular, high-quality production, find themselves striving to find ways to keep short-term and long-term productivity as high as possible. Proponents of pair programming ("pairing") claim that it boosts long-term productivity by substantially improving the quality of the code. But it is fair to say that for a number of reasons, pairing is by far the most controversial and least universally-embraced of the agile programmer practices.” (Version One)

Benefits to using Paired Programming

1. Increased code quality: “programming out loud” leads to clearer articulation of the complexities and hidden details in coding tasks, reducing the risk of error or going down blind alleys
2. Better diffusion of knowledge among the team, in particular when a developer unfamiliar with a component is pairing with one who knows it much better
3. Better transfer of skills, as junior developers pick up micro-techniques or broader skills from more experienced team members
4. Large reduction in coordination efforts, since there are N/2 pairs to coordinate instead of N individual developers
5. Improved resiliency of a pair to interruptions, compared to an individual developer: when one member of the pair must attend to an external prompt, the other can remains focused on the task and can assist in regaining focus afterwards

Pairing Mechanics

“Pairing involves having two programmers working at a single workstation. One programmer "drives," operating the keyboard, while the other "navigates," watching, learning, asking, talking, and making suggestions. In theory, the driver focuses on the code at hand: the syntax, semantics, and algorithm. The navigator focuses less on that, and more on a level of abstraction higher: the test they are trying to get to pass, the technical task to be delivered next, the time elapsed since all the tests were run, the time elapsed since the last repository commit, and the quality of the overall design. The theory is that pairing results in better designs, fewer bugs, and much better spread of knowledge across a development team, and therefore more functionality per unit time, measured over the long term.” (Version One)

Works Cited

"Pair Programming." Guide to Agile Practices. N.p., n.d. Web. 25 Feb. 2015.

"Pair Programming." VersionOne Product Blog. N.p., n.d. Web. 25 Feb. 2015.