Jacob Steckel

CSC402

Article Review 2

Spring 2018

Strengthening the Case for Pair Programming

Pair programming is when two users work side by side on the same code in hopes that it will improve efficiency and quality. Programmers have used pair programming in the past because they feel that it creates a better finished product. This article digs deeper into pair programming and gathers real evidence showing that two programmers working side by side does improve software quality while also increasing time efficiency. Their experiment consisted of both professional programmers and advanced undergraduate students so their results cover both novice and expert level programmers.

In 1998, John Nosek, a professor at Temple University conducted a study where he gave 15 experienced programmers 45 minutes to do a challenging problem. 5 of them had to do it individually and the other 10 worked in pairs. The results showed that the paired groups completed the task 40% faster than their counterparts while also producing more efficient algorithms and code. The paired programmers admitted that they didn’t think working in pairs would be very enjoyable. They thought that they may butt heads with their partner. After the experiment, the paired programmers said they enjoyed the process more and had greater confidence in their final product.

After explaining some early studies on pair programming, the article goes into the development cycle that is involved. To maximize the efficiency of paired programming, the two users must work together as one unified organism. They must always be on the same page and basically work with one combined mind. One user controls the pencil, mouse, and keyboard and he is the one who is literally writing the code. This user is called the driver. The other partner watches the driver carefully looking for problems with the code, looking up helpful resources and alternative methods, and considers other strategies that may improve the program. The partners typically switch roles periodically so each person has a chance to input different ideas.

For the University of Utah study, they tested the effectiveness with pair programming as well. It took a little while for the pairs to jell, but when they began to learn how to cooperate, the results were clear. For the programmers who coded individually, they scored 73.4%, 78.1%, 70.4%, and 78.1% on programs 1-4, respectively. On the same programs, the pairs scored 86.4%, 88.6%, 87.1%, and 94.4%, respectively. You can clearly see the distinction between the two and with nearly a 10% higher average score, I think you can say that pair programming is an effective strategy. Also, by working in tandem the pairs completed the programs between 40% and 50% faster than the opposition. Pair programming should be something that companies consider since it can increase time efficiency along with improving the overall code.

Obviously, there can be potential risks when pairing two programmers together. Such as if a partner is an awful teammate, or has too big of an ego to collaborate effectively with their partner. When picking partners, these are factors that must be considered. But, when you can find two users who are willing to collaborate, the benefits of pair programming are extraordinary and shouldn’t be ignored by corporations.