Pair Programming

All significant development is done in pairs. We have found that progress is faster, we can work longer without losing headway, and quality is higher. Typically the person types who has the best feel for where the code is going. The observer catches detail errors (spelling, formatting, method names) and at the same time tends to have a wider overall view of how the development is going.

It is as if the typist is crafting the individual methods, while the observer is monitoring strategy and tiny detail at the same time.

We have found that with very little practice, most people adapt quickly to pair development. They get to like it, because progress is discernibly faster.

From a project viewpoint, the practice ensures that at least two people are very familiar with all the code. This pays off very quickly as teams go on to other tasks.

On the contrary, some complex tasks require concentration and developers need privacy to concentrate.

- We do allow people to prototype on their own. The recommended practice is then to throw the prototype away and develop it anew with a partner. If we develop something alone (such as over the weekend), we will review it with a partner. This is OK if kept in check. Do not think of it as a clever way to avoid pairing.
- If a task is so complex that it requires that much concentration, it is almost certain that we're doing the wrong thing. We have found no valid exceptions to this guideline in our entire project: simplicity has always been possible, and always been better.

On the contrary, I just like to work alone.

- That's fine. You just can't work with us.

      (Alistair Cockburn accused Kent and me of playing "Do it My Way or Get Out". Follow the above link for some discussion of that.)

Pair Programming Screenplay

      An exchange recently on comp.lang.smalltalk, though it actually involved three people, showed a good example of how pair programming works. Check it out.