Ian Hoyos

Prof. Perez  
CSC 431: Software Engineering

**Successful Software**

There is a common misconception that all you need is a good idea. Not to say that this isn’t true, but a good idea will only get you so far. In the software industry, a good idea will get you to the door, maybe some investment to get started, but after that it’s all up to how efficient your development is. You could be Bill Gates, but if your software engineering practices are poor, you will fail. This is because success in the software world depends on how well you can work, and part of that depends on which methodology you exercise and how well you exercise it.

So, this begs the question, what are good software engineering practices? While code is big part of it, how that code gets created is equally as important. This is where methodologies come in. The industry trails two predominant styles, the Agile methodology and the Waterfall methodology. Agile is more of a loose and flexible approach, while waterfall is more rigorous and depends largely on detailed planning. But for the purposes of this paper, and being an Agile fan myself, I will refer to the Agile disciplines.

So how does agile define good engineering? In the Agile methodology, there are four key values which if followed will enable you to successfully develop your software. *Communication* is the first. Communication is key when developing good software because regular interactions between team members enables you to cohesively make decisions and solve problems quickly. A lot of the Agile process is to keep tasks small and keep a good pace, and to do this team members need to constantly keep each other in the loop of what is going on.

*Simplicity* is the second. The Agile methodology particularly relies on simplicity, on having individual teams complete only the work that a client asks for. By avoiding building anything that is unnecessary you reduce the amount of code, documentation and simplify what could otherwise be complex systems. By keeping tasks simple teams consistently produce good working code. This also paves the way for nice, on time deliveries, which clients love.

Clients also love to be in the loop, and for a team to be able to deliver exactly what the client wants, consistently, it is important to seek feedback as much as possible. Therefore, *feedback* Is the third. Establishing constant communication will bring to light any changes that may need to be made, earlier rather than later. This also helps your team stay focused on what needs to be done.

The fourth key value is *Courage*. It doesn’t just take an idea, or coding skills to make good software. It also takes the courage to speak openly, honestly, and be willing to accept other’s opinions and views, and accept that things may need to changed. A good team will enforce this, and in return all members will rightly feel as an integral part of the team. This promotes motivation and reminds all members that good software engineering is a unified effort.

These four key values are an insight into good software engineering. In tech, often people will think that successful companies are simply the ones with the most money or talent, and while to some extent those things help, successful companies arise out of good engineering practices. If you wish to create successful software and earn a reputation of good engineering, then the iteration of these four key values will help you succeed in attaining these. That is why it is imperative for all software engineers and all employers looking to make successful software, to apply these values to their development processes.