I am from Portland, Maine which is about 2 hours north of Boston. In college I studied finance and economics with a minor in computer science. I started my career as an analyst with Wells Fargo in San Francisco and began studying Systems Analysis and Design at UC Berkeley. After a couple of years, I was working in Silicon Valley during the .com boom of the mid 1990’s. This was a very exciting and interesting place to be at the time. At this time I focused my career on business intelligence and data warehouse design. I eventually found my way back to the Boston area, got married and have been living just north of Boston for the past 14 years. In that time I have spent over 10 years as an independent contractor doing software development and the last 5 years working for the Federal Reserve System where I manage a team of developers. During my career I have worn many hats. I have been a scrum master, a project manager, a product owner, a subject matter expert, and a software developer.

**Successes, Challenges, Lessons Learned.**

Working as a contractor provided me with the unique opportunity to work on many different projects across many different organizations and industries. I could see what worked and what did not. I could also see the common challenges different companies face with software development. I like to think that I was doing agile development before the term was really applied in the software world. The most common challenge I saw in developing software was not achieving the technical capacity to deliver on a solution, but the constantly changing requirements during the development cycle. As a developer, you are put in the awkward position of having to tell the people paying your salary that they have to stop otherwise they are going to sink the project. Having a backlog would allow me to never have to say no, but at the same time, it would force project sponsors to prioritize requirements in the face of time and budget constraints. Keeping them constantly engaged and part of a feedback cycle would guard against any surprises when the solution would be delivered. This is in contrast to the more traditional waterfall methodology where requirements would be documented in detail and a few months later the solution would be presented to stakeholders with the hope that they would be happy with the end result. Over many years and many projects I have seen the needle vacillate between a more structured approach like waterfall and a less structured approach like agile. I think both methodologies have their strengths and weaknesses and some projects are better suited to agile while some are better suited to waterfall. Agile lends itself to small project teams where the desired outcome is not well defined or there are a lot of stakeholders involved. Waterfall seems to be better for teams with a lot of people and interdependencies where the solution is well defined and agreed upon up front.

**What I hope to learn**

I am looking forward to learning from other people’s experiences. I would like to learn more about agile from the product owner’s perspective and become more familiar with the agile ceremonies and artifacts. I tend to hear “that is not really agile” a lot from different colleagues, so, I would like to be more in depth with my knowledge of agile and hopefully be able to call myself an expert or at least a qualified practitioner. We are also applying a lot of agile principles with my current team and I hope to be able to apply what I learn here to the projects I am working on.