**Exercise 1**

**Software engineering and Agile software development**

**What is Agile?**

Agile is the ability to create and respond to change. It is a way of dealing with, and ultimately succeeding in, an uncertain and turbulent environment.

**What is Agile Software Development?**

Agile software development is more than frameworks such as Scrum, Extreme Programming, or Feature-Driven Development (FDD).

Agile software development is more than practices such as pair programming, test-driven development, stand-ups, planning sessions, and sprints.

Agile software development is an umbrella term for a set of frameworks and practices based on the values and principles expressed in the Manifesto for Agile Software Development and the 12 Principles behind it. When you approach software development in a particular manner, it’s generally good to live by these values and principles and use them to help figure out the right things to do given your particular context.

One thing that separates Agile from other approaches to software development is the focus on the people doing the work and how they work together. Solutions evolve through collaboration between self-organizing cross-functional teams utilizing the appropriate practices for their context.

**What are Agile Methodologies?**

Agile methodologies are the conventions that a team chooses to follow in a way that follows Agile values and principles.

“Wait,” you’re probably saying, “I thought Scrum and XP were Agile methodologies.” Alistair applied the term framework to those concepts. They certainly were born from a single team’s methodology, but they became frameworks when they were generalized to be used by other teams. Those frameworks help inform where a team starts with their methodology, but they shouldn’t be the team’s methodology. The team will always need to adapt its use of a framework to fit properly in its context.

**What about Agile Project Management or Agile Business Analysis?**

Agile Software Development became more popular, people involved with software development activities but who didn’t personally develop software looked for some way to figure out how these Agile ideas applied in their line of work.

The Agile Manifesto and the 12 Principles were written by a group of software developers (and a tester) to address issues that software developers faced. When you think of Agile as a mindset, that mindset can be applied to other activities.

**What are the Key Agile Concepts?**

**User Stories:** In consultation with the customer or product owner, the team divides up the work to be done into functional increments called “user stories.” Each user story is expected to yield a contribution to the value of the overall product.

**Daily Meeting:** Each day at the same time, the team meets so as to bring everyone up to date on the information that is vital for coordination: each team members briefly describes any “completed” contributions and any obstacles that stand in their way.

**Personas:** When the project calls for it – for instance when user experience is a major factor in project outcomes – the team crafts detailed, synthetic biographies of fictitious users of the future product: these are called “personas.”

**Team:** A “team” in the Agile sense is a small group of people, assigned to the same project or effort, nearly all of them on a full-time basis. A small minority of team members may be part-time contributors, or may have competing responsibilities.

**Incremental Development:** Nearly all Agile teams favor an incremental development strategy; in an Agile context, this means that each successive version of the product is usable, and each builds upon the previous version by adding user-visible functionality.

**Iterative Development:** Agile projects are iterative insofar as they intentionally allow for “repeating” software development activities, and for potentially “revisiting” the same work products.

**Milestone Retrospective:** Once a project has been underway for some time, or at the end of the project, all of the team’s permanent members (not just the developers) invests from one to three days in a detailed analysis of the project’s significant events.