# Table of Contents

# Agile Values and Principles

The granddaddy of them all, there is no self-respecting agile book that doesn't reference these in some way.

The original is at http://www.leansystemssociety.org/ and is best read there. Following list is maintained for my convenience.

# Values

The Lean Systems Society published its credo at the 2012 Lean Software & Systems Conference. This was based on a set of values published a year earlier. Those values include:

- Accept the human condition
- Accept that complexity & uncertainty are natural to knowledge work
- Work towards a better Economic Outcome
- While enabling a better Sociological Outcome
- Seek, embrace & question ideas from a wide range of disciplines
- A values-based community enhances the speed & depth of positive change

Not directly related to software so here is a slightly different take "Learning Agile: Understanding Scrum, XP, Lean, and Kanban" - Andrew Stellman, Jennifer Greene:

- “Eliminate waste: Find the work that you’re doing that doesn’t directly help to create valuable software and remove it from the project.
- Amplify learning: Use feedback from your project to improve how you build software.
- Decide as late as possible: Make every important decision for your project when you have the most information about it — at the last responsible moment.
- Deliver as fast as possible: Understand the cost of delay, and minimize it using pull systems and queues.
- Empower the team: Establish a focused and effective work environment, and build a whole team of energized people.
- Build integrity in: Build software that intuitively makes sense to the users, and which forms a coherent whole.
- See the whole: Understand the work that happens on your project — and take the right kind of measurements to make sure you’re actually seeing everything clearly, warts and all.

# The Seven Wastes of Software Development

The Poppendiecks took a more concrete approach to the concept of “waste” and applied it to software, coming up with a list of the Seven Wastes of Software Development

- Partially Done Work: Not “done” work that can be delivered to the customer
- Extra Processes: The waste assciated with a requirements document that is never read, for example, or dependencies on 3rd parties that are not addressed.
- Extra Features: Working on something no-one wants.
- Relearning: Something that you really should have already known.
- Handoffs: For example, when teams don't sit together.
- Delays: Or waiting. When people could be being productive
- Task Switching: Multi-tasking is just bad.
- Defects: No customer wants a defect - so its just not valuable, plus it takes time away from creating valuable stuff.

The Poppendiecks came up with an idea called the Seven Wastes of Software Development. Like much of Lean, this is adapted from ideas developed at Toyota in the middle of the last century. Mary and Tom Poppendieck’s mapping of Shigeo Shingo’s “Seven Wastes of Lean Manufacturing” into the software development world.

Learning, Reference, Basics, FirstSprint, Values, Principles, Lean

~~LINKBACK~~ ~~DISCUSSION~~