

Agile Manifesto

- <http://agilemanifesto.org/principles.html>
- February, 2001
- XP, SCRUM, DSDM, Adaptive Software Development, Crystal, Feature-Driven Development, Pragmatic Programming

Continuous Delivery

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software
 - Sustainable process
 - Feature boxed

Changing Requirements

- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage
 - Particularly appropriate for situations with volatile requirements

Frequent Delivery

- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale
 - Very small increments
 - Implies increased *release* overhead

Customer Involvement

- Business people and developers must work together daily throughout the project
 - Implies identified customer (or surrogate)
 - And big-time customer commitment
 - Suitable for in-house projects

Motivated Participants

- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done
 - All processes want motivated individuals
 - Agile office design

Meetings

- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
 - Questionable
 - Have to identify who should be attending

Software Focus

- Working software is the primary measure of progress
 - Deemphasis of infrastructure, both architectural and process

Sustainability

- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely
 - All processes would like this
 - No crises requiring overtime

Amortization

- Continuous attention to technical excellence and good design enhances agility
 - Amortization of design and quality activities throughout development
 - Localizes design decisions

Simplicity

- Simplicity--the art of maximizing the amount of work not done--is essential
 - Avoid generalizing

Self Organization

- The best architectures, requirements, and designs emerge from self-organizing teams
 - Conway's law

Reflection

- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly
 - Amortization of process