Introduction to Agile Software Development

Andrea Polini

Software Project Management
MSc in Computer Science
University of Camerino

What is agile?

Observations

- No single recipe that results in perfect software every time. Agile teams have ideas and ground rules that help to guide the team to make the right choices and avoid problems, or deal with them when they emerge.
- A good developer almost always has opinions about the whole direction of the project
- Changes are unavoidable
- Software is an highly added value artefact, quality is strongly dependent from people

Bibliography

Textbook


Andrew Stellman and Jennifer Greene
Learning Agile Understanding Scrum, XP, Lean, and Kanban
O'Reilly 2015.

What is agile?

Set of methods and methodologies

- more effective work
- more efficient work
- make better decisions

Promises

- On time
- high quality
- highly maintainable
- user happy
- working normal hours

What is agile?

Different mindset, based on ideas, values, and principles

- focus on teams over individuals
  - sharing knowledge
  - taking responsibilities
  - taking decisions
  - feeling commitment

Note

Manager and Developers generally have different perspectives on the same project. This affect the introduction of novel approaches to development. e.g. introduction of a daily standup meeting

What is agile?

Different mindset, based on ideas, values, and principles

- focus on teams over individuals
  - sharing knowledge
  - taking responsibilities
  - taking decisions
  - feeling commitment

Note

Manager and Developers generally have different perspectives on the same project. This affect the introduction of novel approaches to development. e.g. introduction of a daily standup meeting

Agile values

A group of highly skilled and innovative people started agile as a revolution against the “waterfall mindset”.

Agile values

- Individuals and interactions **over** processes and tools
- Working software **over** comprehensive documentation
- Customer collaboration **over** contract negotiation
- Responding to change **over** following a plan

Can waterfall work?

- Good communication
- Good practices
- It's more important the creation of the plan than sticking to it

The big issue . . . Requirements up-front!

Can waterfall work?

- Good communication
- Good practices
- It's more important the creation of the plan than sticking to it

The big issue . . . Requirements up-front!

Better-than-not-doing-it

For a “waterfall” team one of the most complex issue in PM is the transition toward an Agile mindset

Going agile is not equal to becoming an agile team

- Not just tools, techniques, and practices (can just lead to a better-than-not-doing-it effect)

Agile tools, practices, techniques

- test driven development, automated build script, build server, scrum, iterations, task board, velocity, burndown charts, user stories, product owner, release plan

In a fractured perspective everyone has a different view of the agile practice

Often team’s members just improve their individual capabilities in activities for which they were already good at

Better-than-not-doing-it

For a “waterfall” team one of the most complex issue in PM is the transition toward an Agile mindset

Going agile is not equal to becoming an agile team

- Not just tools, techniques, and practices (can just lead to a better-than-not-doing-it effect)

Agile tools, practices, techniques

- test driven development, automated build script, build server, scrum, iterations, task board, velocity, burndown charts, user stories, product owner, release plan

  In a fractured perspective everyone has a different view of the agile practice

Often team’s members just improve their individual capabilities in activities for which they were already good at

Better-than-not-doing-it

For a “waterfall” team one of the most complex issue in PM is the transition toward an Agile mindset

Going agile is not equal to becoming an agile team

- Not just tools, techniques, and practices (can just lead to a better-than-not-doing-it effect)

Agile tools, practices, techniques

- test driven development, automated build script, build server, scrum, iterations, task board, velocity, burndown charts, user stories, product owner, release plan

  In a fractured perspective everyone has a different view of the agile practice

Often team’s members just improve their individual capabilities in activities for which they were already good at

Individuals and interactions **over** processes and tools

A great tool can sometimes help you do the wrong thing faster

It is important to understand people in the team:

- how they work together
- how each person's work impact everyone else

the Value and the practices

Used practices in line with the value:

- Daily stand-up meeting
- retrospectives
- user stories
- ...

Working software **over** comprehensive documentation

Often complex software documents have no readers. Agile methodologies aim at providing working software that adds value to the organization

Obviously this does not mean that no documentation should be provided. Instead documentation should save more time and effort than it costs

the value and the practices

- comments
- javadocs
- test-driven development
- demoing at the end of each iteration
- ...

Customer collaboration **over** contract negotiation

The objective is to provide valuable software to the customer, so software that he/she really needs.

Issues

Up-front requirements reduce customer involvement and the possibility to revise the plan after the contract is signed

Agile methodologies foster inclusion of the customer in the development team and strict cooperation.

Give customers what they really need, and not just what they ask for

If I had asked people what they wanted, they would have said faster horses
Henry Ford

Responding to change **over** following a plan

A plan provides a comfortable path toward the development of a possible wrong software. Agile methodologies ask for taking into consideration any change that could emerge.

Task board

The use of a task board is a practice helping the team to take the right decision when a change emerge. Three sections each one contaning user stories (in general) in on of the possible three different states (To do, In progress, Done).

- electronic format vs. paper based

Principles over practices

> *Without concrete practices, principles are sterile; but without principles, practices have no life, no character, no heart. Great products arise from great teams–teams who are principled, who have character, who have heart, who have persistence, and who have courage.* — *Jim Highsmith*

A team becomes agile if it shares the values and not just if they adopt the practices

Interesting questions

- When the agile manifesto talks about not having comprehensive documentation, does that mean we don't have to write anything down?
- I've definitively heard that agile means doing any planning, and instead jumping straight into programming. Isn't that more efficient?
- Can I have the developers on the team go agile, but leave the rest of the team alone?
- If I'm not using Scrum, XP, Lean, Kanban, does that mean my team isn't agile?

Interesting questions

- When the agile manifesto talks about not having comprehensive documentation, does that mean we don't have to write anything down?
- I've definitively heard that agile means doing any planning, and instead jumping straight into programming. Isn't that more efficient?
- Can I have the developers on the team go agile, but leave the rest of the team alone?
- If I'm not using Scrum, XP, Lean, Kanban, does that mean my team isn't agile?

Interesting questions

- When the agile manifesto talks about not having comprehensive documentation, does that mean we don't have to write anything down?
- I've definitively heard that agile means doing any planning, and instead jumping straight into programming. Isn't that more efficient?
- Can I have the developers on the team go agile, but leave the rest of the team alone?
- If I'm not using Scrum, XP, Lean, Kanban, does that mean my team isn't agile?

Interesting questions

- When the agile manifesto talks about not having comprehensive documentation, does that mean we don't have to write anything down?
- I've definitively heard that agile means doing any planning, and instead jumping straight into programming. Isn't that more efficient?
- Can I have the developers on the team go agile, but leave the rest of the team alone?
- If I'm not using Scrum, XP, Lean, Kanban, does that mean my team isn't agile?

Agile principles

Principle motivations

The Agile manifest signers identified ground rules and ideas that help the team to make the right choices and avoid problems. 12 principles were then defined.

Principles can be organized according to four sections:

- delivery
- communication
- execution
- improvement

Principles list – Delivering the project

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage
   - Nobody get's in trouble when there's a change
   - We are all in this together, including the customer
   - We don't sit on change until it's too late
   - Changes are not solutions to previous mistakes
   - Learn from the changes
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
   - Practice of timeboxed iterations

Principles list – Delivering the project

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage
   - Nobody get's in trouble when there's a change
   - We are all in this together, including the customer
   - We don't sit on change until it's too late
   - Changes are not solutions to previous mistakes
   - Learn from the changes
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
   - Practice of timeboxed iterations

Principles list – Delivering the project

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage
   - Nobody get's in trouble when there's a change
   - We are all in this together, including the customer
   - We don't sit on change until it's too late
   - Changes are not solutions to previous mistakes
   - Learn from the changes
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
   - Practice of timeboxed iterations

Principles list – Communicating and working together

Objective is somehow to suggest the definition of as much documentation as you need to run the project. And this depends from communications habits. Comprehensive documentation leads to unnecessary changes and wasted effort. Waterfall real practices on changes often do not reflect theory

4. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
5. Businesspeople and developers must work together daily throughout the project.
6. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
   - rewards on bug metrics for individuals is not a good idea
   - reward BAs on the amount of documentation is not a good idea
   - CYA attitude

Principles list – Communicating and working together

Objective is somehow to suggest the definition of as much documentation as you need to run the project. And this depends from communications habits. Comprehensive documentation leads to unnecessary changes and wasted effort. Waterfall real practices on changes often do not reflect theory

4. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
5. Businesspeople and developers must work together daily throughout the project.
6. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
   - rewards on bug metrics for individuals is not a good idea
   - reward BAs on the amount of documentation is not a good idea
   - CYA attitude

Principles list – Communicating and working together

Objective is somehow to suggest the definition of as much documentation as you need to run the project. And this depends from communications habits. Comprehensive documentation leads to unnecessary changes and wasted effort. Waterfall real practices on changes often do not reflect theory

4. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
5. Businesspeople and developers must work together daily throughout the project.
6. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
   - rewards on bug metrics for individuals is not a good idea
   - reward BAs on the amount of documentation is not a good idea
   - CYA attitude

Principles list – Project execution

7 Working software is the primary measure of progress.

8 Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

9 Continuous attention to technical excellence and good design enhances agility.

Principles list – Project execution

7 Working software is the primary measure of progress.

8 Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

9 Continuous attention to technical excellence and good design enhances agility.

Principles list – Project execution

7 Working software is the primary measure of progress.

8 Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

9 Continuous attention to technical excellence and good design enhances agility.

Principles list – Costantly improving the project and the team

10 Simplicity – the art of maximizing the amount of work not done – is essential.

11 The best architectures, requirements, and designs emerge from self-organizing teams.

- the work generally starts from user stories
- incremental design, instead of big design architecture covering all requirements

12 At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Principles list – Costantly improving the project and the team

10 Simplicity – the art of maximizing the amount of work not done – is essential.

11 The best architectures, requirements, and designs emerge from self-organizing teams.

- the work generally starts from user stories
- incremental design, instead of big design architecture covering all requirements

12 At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Principles list – Costantly improving the project and the team

10 Simplicity – the art of maximizing the amount of work not done – is essential.

11 The best architectures, requirements, and designs emerge from self-organizing teams.

- the work generally starts from user stories
- incremental design, instead of big design architecture covering all requirements

12 At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

FAQs

- I'm a project manager, and I'm still not clear on how I fit into an agile team: What's my role in all of this?
- If the whole team plans together, then does that mean nobody is in charge? How are decision made?

Bibliography

Andrew Stellman and Jennifer Greene
Learning Agile Understanding Scrum, XP, Lean, and Kanban
O'Reilly 2015.

- Chapters 1, 2 and 3