# Agile Variations and Lean Software Development

## Agile practices

### [recall] Some of the most popular Agile practices

- Extreme programming and Scrum – 66% of the software development methodologies – according to a survey conducted by Version One in 2015
- Scrum – nearly the most common
- ...

### More agile practices

"Agile is always changing and evolving."

### Variations on Agile practices

Lean software development: more and more popular in the software development world

## Practices which have evolved out of Agile

### Agile Unified Process / Agile UP

Agile UP = Agile Unified Process

- parallel development phases
- philosophies of Agile: in order to increase developer morale and improve upon collaboration on projects
- test-driven-development: small, incremental releases to improve upon the basic Unified process.
- focuses on following a process over specific tools
  & on simplicity, empowering people, and a customization of the methodology to suit each project's needs

### other variations

- Dynamic Systems Development
- feature driven development
- Scrumban and behavior-driven development

## Quiz

**Which of the following are traditional Agile methodologies?**

- A. Unified Process,
- ✓ B. Scrum,
- ✓ C. Extreme Programming,
- ✓ D. Dynamic Systems Development.

## a variation of Agile: Lean software development

### Lean & Toyota

» to reduce waste in the production process and increase the quality of their vehicles

- in the manufacturing industry
- proposed in the mid 90s as a way of reducing risk within projects.
  - reducing waste,
  - starting new work only when necessary
  - reducing process bottlenecks
  - working only on the things that will add value to the project

### Lean & software development

in 2003, a book on Lean by Mary and Tom Poppendieck

### Lean – a mindset

» a set of core principles to reduce your project's risks and deliver a high quality product to your clients quickly

## Lean software development

seven principles of software development:

- eliminating waste
- amplifying learning
- deciding as late as possible,
- delivering as fast as possible
- empowering the team
- building quality in,
- seeing the whole.

## 1: eliminating waste

**waste**

“In Lean, anything that doesn’t add value is considered waste.”
» In Lean, anything that is considered waste should be ruthlessly removed from the development process.

**example of waste in car manufacturing**

- car manufacturing: producing car body and tires at full capacity
- keep working?
  » typical mindset in software development is that in order to maximize your team’s productivity, all of your developers should be working on something at any given time.

## Quiz

**Which of the following is not a principle of lean?**

- A. eliminating waste.
- B. delivering as late as possible.
- C. deciding as early as possible.
- D. deciding as late as possible.

## Quiz -A

**Which of the following is not a principle of lean?**

- A. eliminating waste.
- ✓ B. delivering as late as possible.
- ✓ C. deciding as early as possible.
- D. deciding as late as possible.

## 1. eliminating waste

### problems of keep developers busy:

- developpers become ineffective
- Instead of developing core features, by the development team constantly code new extra features.
- » result in low quality product & cause issues with the main product & leave features incorporated at all

### a key example of how software production can be wasteful

- people always think that keeping some of the development team idle in the product will be wasteful
- by forcing the dev team to be constantly on, that might force them to create unnecessary work.

## 1. eliminating waste

**the following can also be wasteful:**

- process delays
- unnecessary meetings
- unclear requirements
- product defects
- ...

## 2. amplifying learning

### Why amplifying learning

» “to make sure that your team builds the right product through developing alternatives, continuous feedback and refinement.

### an example, a start

- Your development team creates a very basic list of user requirements.
- They quickly get things off the ground.
- They can then very briefly plan out one possible approach for this product and begin to develop that basic functionality.

## 2. amplifying learning

### How? 1/3, try alternatives

In order to amplify learning though, a good practice is to stop at this stage. Don't do anymore work on this version.

- "Don't focus too deeply on one idea before you've fully explored another."
- "Explore ideas thoroughly by thinking through alternatives before proceeding with actions."
- "Involving the client in trying different approaches, your development team learns how best to tackle their specific problem."

## 2. amplifying learning

**Could this be done another way?**

- trying different approaches
- The development team briefly plans and develops these alternative approaches as well.
- The product is given its best chance at meeting your client's needs in the most satisfying way.

## 2. amplifying learning

### Run integration and unit tests after each build of the product

- » “Tests can bring key insights to your team which can be used to inform other versions of the product in future planning.”
- to ensure that your code is well written
- learn how your product should function.

### in short iterations

Consider others after really seeing one’s flaws?

- trail and error in short iterations
- » to fail and learn quickly.
- avoid spending too much time on one design or one feature,

## 2. amplifying learning

**How? 3/3, involve clients in a special way - natural selections**

- development teams create different versions of their product before focusing in on one version
- these versions should be constantly presented to the client.

## 2. amplifying learning

### How? 3/3, involve clients in a special way - benefits

- allows the development team to learn through trial and error, what the client really needs?
- also allows the client to learn how they want their product to adjust to their specific problem.

» the development team learns how best to tackle their specific problem

» the development team to decide on what features they want to see built into the product which then brings focus in refinement to the final product.

## Quiz - Q

**context**

Audrey is building a software product with her development team using the Lean software development practices.
She's asked her development team to only focus on one aspect of the project at a time while creating different alternatives for their client.

**By asking her team to focus on one aspect of the project at a time, Audrey is implement which principle of Lean?**

- A. develop small, incremental releases.
- B. amplifying learning.
- C. focusing on frequent delivery.
- D. eliminating waste.

## Quiz - A

By asking her team to focus on one aspect of the project at a time, Audrey is implement which principle of Lean?

- A. develop small, incremental releases.
- B. amplifying learning.
- C. focusing on frequent delivery.
- ✓ D. eliminating waste.

- The answer is D, eliminating waste.
- Amplifying learning is what Audrey is doing, by having her team build different alternatives.
- Developing small incremental releases and focusing on frequent delivery are principles of agile not Lean, although Lean projects can certainly make use of them.

## 3. deciding as late as possible

### deciding as late as possible & amplifying learning

- an excuse to delay the product for more development time?
- making decisions when the data and information you need is available.

### the right process

- Work hard to ensure that there are many alternatives.
- Decisions made before you have enough information are bound to result in poor outcomes.
- Deciding as late as possible, ensures you have the information you need first.

### build the first prototype as final? without developing alternatives

- the development team could have built a poor product.
- they wouldn't even known that they were building a poorer product.

# 3. deciding as late as possible

## an example

- Imagine a software team that builds a basic piece of software for their client.
- The client then sees this prototype and says yep, that's exactly what I need. Build that.

» Well, initial prototype is satisfying! Great!

- Before you decide,
  the development team being a Lean software development team explores different approaches in order to amplify learning.
- Build alternative versions of the product, integrating what the client likes about the prototype.
- Gather feedback and know the appealing characteristics of this prototype, build into the final

## 4. delivering as fast as possible

### [recall] deciding as late as possible

- get the necessary client feedback in order to focus in on what the client needs
- how much time for the development of each potential prototype?
- but not delaying development
  - » delivering as fast as possible
  - product not only evolves at a rapid rate
  - the client to give feedback at frequent intervals throughout the process

### What happens when the client changes their mind about a feature?

- development happened in rapid iterations
- issue would have been caught and corrected early.
- » minimizes waste and increases the amount of quality built into your product quickly.

## 5. empowering the team

**a quote from Teddy Roosevelt**

"The best executive is one who has sense enough to pick good people to do what he wants done, and self-restraint enough to keep from meddling with them while they do it."

**efforts of a software product manager go to**

- understanding your client's needs
- remove your developers' roadblocks
- ...

## 5. empowering the team

### trus the team

“If you believe your development team has the skills necessary to get the job done right, then why would you spend your time trying to tell them how to do it?”

- let the team members sense the trust
- software product manager are encouraged to listen to their developers and to let developers imagine solutions that satisfy client requirements.
- Your goal should be to listen to what your developers have to say and offer suggestions.
- allows your team to work how they want to work.
- Trust them to make the right technical decisions.
  » Not only does this make sure that your developers are using the most productive method for them, it also serves as a huge boost to their morale.

## Quiz

What is one of the main reasons that a software development team, and client, should decide as late as possible when developing a software product?

- A. it buys them more development time.
- B. it allows them to explore other product designs.
- C. the client never knows what they want.
- D. it allows the team to build working software quickly.

## Quiz

What is one of the main reasons that a software development team, and client, should decide as late as possible when developing a software product?

- A. it buys them more development time.
- ✓ B. it allows them to explore other product designs.
- C. the client never knows what they want.
- D. it allows the team to build working software quickly.

## 6. building quality in

### building quality in

Aims to ensure that your development team is using the best practices available to them in order to avoid errors.
Ways to build integrity in:

- TDD, focus the actual problem that the code is meant to solve, not the existing code
  - Building your tests first means you're actually testing if your code solves your problem. And not only if your code works.
- Pair programming: increase the quality of their product also very costly in terms of developer time
  » it's a wonderful tool for developing great software, don't rely on it to build your entire product.
- other ways: well commented code, good documentation, refactoring code to be more efficient, and automated testing.

## 7. seeing the whole

### seeing the whole

- what the end user sees should not be just some features strung together.
- a cohesive product with components that flow into each other logically and smoothly

### the big picture

- think about your product as a whole before releasing it.
- developers must be constantly aware of how their piece of the puzzle fits into the big picture.

context

- Ross is building a software product for medical research.
- Since the work he does everyday saves lives, he is continually trying to improve his software development processes.
- He recently came across Lean software development, and has begun implementing some of its principles.
- He tries to encourage his team to see the project as part of a larger purpose, and gets them to create many different versions of the software before committing and focuses on building quickly.
- In order to save time, he has asked his development team not to develop unit tests, document code, or use any specific programming practices.

What aspect of Lean Software Development is Ross actively omitting from his project?

A. eliminating waste. B. amplifying learning. C. deciding as late as possible. D. building quality in.

## context

- Ross is building a software product for medical research.
- Since the work he does everyday saves lives, he is continually trying to improve his software development processes.
- He recently came across Lean software development, and has begun implementing some of its principles.
- He tries to encourage his team to see the project as part of a larger purpose, and gets them to create many different versions of the software before committing and focuses on building quickly.
- In order to save time, he has asked his development team not to develop unit tests, document code, or use any specific programming practices.

## What aspect of Lean Software Development is Ross actively omitting from his project?

A. eliminating waste. B. amplifying learning. C. deciding as late as possible. [✓] D. building quality in.

## Lean Software Development principles

beyond the principles outlined in Mary and Tom Poppendieck's book – "Lean Software Development: An Agile Toolkit". written in 2003

principles added by the community of Lean software developers

- use the scientific method when building a system
- encouraging leadership
- ...

## use the scientific method when building a system

### acting upon data

- carrying on with a project's design using hunches, guesses, and experience ?
- encourages developers to base their development off real data.
- software product manager: need to initiate experiments to test ideas, and collect data.
- Analyzing and acting upon this data, allows software developers and clients, to make informed decisions about the product.
- gain credibility by backing potential decisions with data.
- decide as late as possible principle

## encouraging leadership

**encouraging leadership – an extension on the principle of empowering the team**

- aims to bring out the best in each person on the team.
- enabling initial individual developers to be courageous, innovative, inspirational, and collaborative.

## Lean Software Development principles

these principles are (names could be different):

- eliminating waste,
- amplifying learning,
- deciding as late as possible,
- delivering as fast as possible,
- empowering the team,
- building quality in,
- and seeing the whole.

## Lean Software Development principles

**Warranty at the end of Poppendieck's book**

"Lean principles are warranted to be tried and proven in many disciplines, and when properly applied, they are warranted to work for software development."

**Proper application**

All of the Lean principles are employed, and that thinking tools are used to translate them into agile practices, appropriate for the environment.

This warranty is invalid, if the practices are transferred directly from other disciplines or domains, without thinking.

- Lean Software Development further expands the theoretical foundations of Agile software development.
- A tool kit / methodology for organizations whether big or small