**Lean Software Development**

**Introduction:**

Lean software development was created by Tom and Mary Poppendieck in 2003.

The history of Lean software development methodology began in the middle of the 20th century. At that time a famous Japanese motor corporation Toyota had big problems with product delivery. Its manufacturing chains were too long and included lots of unnecessary stages. The managers of the company found the solution for this problem, inventing a new project management system originally called Toyota production system. The main idea was to improve the terms of product delivery by waste elimination allowing the enterprise to make its manufacturing chains shorter and to deliver final products faster. The company came out with a simple and effective definition of waste, which was: anything that didn’t impact the functionality of the final product was considered a waste. Later it became popular around the world and changed its name to Lean manufacturing.

**Implementation of the lean methodology on software development:**

In 2003 Lean Methodology was applied to software development. Tom and Mary Poppendieck published their famous book “Lean Software Development”. It was a practical guide on the Issue of Lean implementation on software engineering. Modern researchers define seven Lean principles, being flexibility of project tasks and respect to the team members the main ones. Lean is and agile methodology. Its projects have iterative structure.

**Values:**

In 2011 the Lean Systems society published a set of values. Those values are:

1. **Accept the human condition:** To develop any product we need people and process. Human behaviour is led by their emotions, thoughts and traits and it is not constant. We must accept and embrace human condition rather than to denying it and expecting robot like behaviour.
2. **Accept that complexity and uncertainty are natural to knowledge work:** In most of the project initial stages complexity and uncertainly is unknown and is known up to some extent after several studies. But Variability can’t be anticipated in advance.
3. **Work towards a better economic outcome:** Investors and owners of businesses deserve a ROI (Return on investment). Employees and workers deserve a fair rate of pay performing this work. Customers deserve a food product/service for a fair price.
4. **Seek, embrace and question ideas from a wide range of disciplines:** Teams need to seek continuously for new ideas, brainstorm on it and embrace them to improve quality of process and product. This kind of culture needs to be developed in all disciplines involved in software product development.
5. **A values-based community enhances the speed and depth of positive change:** Share your new ideas/innovations and discuss with a value-based community which in turn gives more improvements as many people practice your ideas in their projects.

**Principles:**

**Eliminate waste:** To reduce waste it is critical that development teams be allowed to self-organize and operate in a manner that reflects the work they’re trying to accomplish. Finished product must not have waste.

**Build in quality:** Your process should not allow defects to occur in the first place, but when this happens you should work in such a way that you do a bit of work, validate it, fix any issues found and iterate.

**Create knowledge:** Planning is useful, but learning is essential. Promote strategies, such as iterative development, helping teams discover what stakeholders really want.

**Defer commitment:** Support the business effectively through flexible architectures that are change tolerant and by scheduling irreversible decisions to the last possible moment.

**Deliver quickly:** By limiting the work of a team to its capacity, which is reflected by the team’s velocity (number of “points” of functionality which a team delivers each iteration), it can be established a reliable and repeatable flow of work.

**Respect people:** Sustainable advantage is gained from engaged, thinking people. Enable IT teams don’t control them.

**Optimize the whole:** Manage programs of interrelated systems so a complete product can be delivered to stakeholders. Always look at the bigger picture.

**Practices:**

* Cumulative Flow Diagrams
* Visual Controls
* Visual Kanban Systems
* Small Batch Sizes
* Automation
* Kaizen Events
* Daily stand up meetings
* Retrospectives
* Operations Reviews

**Roles**:

There has been a traditional belief in most businesses about the decision-making in the organization – the managers tell the workers how to do their own job. In this case the roles are turned - the managers are taught how to listen to the developers, so they can explain better what actions might be taken, as well as provide suggestion for improvements. The lean approach follows the Agile Principle "find good people and let them do their own job”, encouraging progress, catching errors, and removing impediments, but not micro-managing.

**Tools:**

Lean software development tools are any application that is used by Lean developers to manage their projects or simplify their work is. Lean tools are aimed at saving your time for other tasks by automating the process of project management. Lean project management tools are subdivided into two categories: paid applications and free tools. Both can visualize the workflow of Lean team in various forms like charts, tables, diagrams.

**Pros and Cons:**

Lean software development has its advantages and disadvantages like any other method. Some advantages include the elimination of superfluous activity, therefore saving time and money. Enables more functionality to be delivered in a shorter period and empowers the development team in the decision-making process, improving motivation to do the best job possible. It is also easily scalable, which makes it a good alternative to more conventional software development methods.

However, it isn’t the perfect method. It is heavily team dependent, which means that you’d better have assembled an excellent team with a high skill level and immense knowledge in the field; learning on the go is impossible and unacceptable. Since the overall development team has so much responsibility spread over several smaller sub teams, it can be relatively easy to lose focus. Lean also requires excellent documentation, meaning that any area that is poorly documented can be underdeveloped or developed incorrectly.

Nevertheless, the pros generally outweigh the cons, especially when it comes to upgrades and additions.

**Pros and cons in a nutshell:**

**Pros:**

1. Allows for delivery of product early.
2. Lower budget and time requirements.
3. The team is motivated to make every product feature perfect.

**Cons:**

1. The workability of the team decides success of software development process.
2. The approach is suitable only for highly skilled developers.
3. Excessive flexibility leads developers to lose focus.

**Reference:**

<https://leankit.com/learn/lean/principles-of-lean-development/?fbclid=IwAR1HYMwFhNao457j5zNlTOFXaXAXMSDFXaIuKXNC_wJhZx8YGJYZOv87kiY>

<http://www.disciplinedagiledelivery.com/lean-principles/?fbclid=IwAR1-farQBGnqhQsUbX5X9yYBUC5DXb8VbZ0fXkkTS5Ms6ZoRoGaBlmJiNMM>

<https://www.linkedin.com/pulse/lean-software-development-values-principles-balaji?fbclid=IwAR3fYTcBN9uDWm4HpILtrjCc_LuAZQ5RwOaXcGwlIONcz3x05ZP9UxjCM8o>

<https://hygger.io/blog/lean-tools-overview/>

<https://bugwolf.com/blog/the-pros-and-cons-of-lean-software-development>

<https://medium.com/globalluxsoft/5-popular-software-development-models-with-their-pros-and-cons-12a486b569dc>

<https://acodez.in/12-best-software-development-methodologies-pros-cons/>