**Lean software development**

There are seven principles to the lean software development:

1. *Eliminate waste:*

Anything that is not of use to the customer is eliminated. There are seven wastes:

* Unnecessary code functionality
* Starting more that can be completed
* Delays in the coding development process
* Unclear or constantly changing requirements
* Bureaucracy
* Slow or ineffective communication
* Partially done work
* Defects or quality issues
* Switching tasks in development

1. *Amplify learning*

This principle focuses on the communication between all parties in an effort to encourage positive feedback and suggestions towards the end product. This principle also encourages trying different software methods as opposed to creating more deliverable documentation, ensuring users expectations are met through the use of intuitive software.

1. *Decide as late as possible*

Delaying decisions until they can be made based on facts minimises the chance of vendor expectations not being due to decisions being made based on assumptions or predictions. This approach ensures the software has the most possible capacity for change built into it and is imperative in the successful development of software through an iterative (agile) development.

1. *Deliver as fast as possible*

The sooner the end product can be delivered without major issues, the sooner feedback can be received and incorporated into the next iteration.

1. *Empower the team*

Ensuring requirements and user expectations for the project are met by having developers and managers in strong communications throughout the project. This means developers have the opportunity to explain different courses of action and the space to provide suggestions for improvements.

1. *Build integrity in*

Developing the software in a way that ensures that the system’s separate software work well together as a whole and with a balance between flexibility, maintainability, efficiency and responsiveness.

1. *See the whole*

Decomposing the task into smaller ‘bit-size’ and more manageable tasks and by standardising the different stages of the development process, the root causes of defects are more likely to be found and eliminated. The larger a system or collection of projects is, the more important the collaboration and communication between vendors.

**XP (eXtreme programming)**

Extreme programming (XP) is an agile software development methodology which embraces changing user requirements to improve software quality and responsiveness, advocating improve productivity using checkpoints for new requirements.

It involves:

1. Programming in pairs or doing extensive code reviews.
2. Unit testing of all code.
3. Avoiding programming of features until they are actually needed.
4. A flat management structure.
5. Code simplicity and clarity.
6. Expecting changes in the customer's requirements as time passes and the problem is better understood.
7. Frequent communication with the customer and among programmers.

It takes from the idea that the beneficial elements of traditional software engineering practices are taken to "extreme" levels.

**Crystal**

A software development methodology that died. <http://blog.scrumstudy.com/what-is-crystal/>

*“Introduced by Alistair Cockburn, Crystal Methods, which is a collection of Agile software development approaches, focuses primarily on people and the interaction among them while they work on a software development project. There is also a focus on business-criticality and business-priority of the system under development. Unlike traditional development methods, Crystal doesn’t fix the tools and techniques of development, but keeps people and processes at the core of the development process. However, it is not only the people or the processes that are important, rather the interaction between the two that is most important.”*

**The Scaled Agile Framework (SAFe)**

Fundamentally, SAFe is a large knowledge database that can be used to assist large scale software development groups, whether they are groups of 40-125 or in the thousands. SAFe encourages collaboration between multiple agile teams, it was developed for and by practitioners. The framework describes roles, responsibilities, artifacts and activities necessary to implement Lean-Agile development in an aim to help businesses help deliver enterprise-class software and systems in the shortest sustainable time.