1.Lean Software Development (LSD) is an agile Framework based on optimizing development time and resources, eliminating waste, and ultimately delivering only what the product needs. The Lean approach is also often referred to as the Minimum Viable Product strategy, in which a team releases a bare-minimum version of its product to the market, learns from users what they like, don’t like and want to be added, and then iterates based on this feedback.

LSD’s strengths include:

* Streamlined approach allows more functionality to be delivered in less time
* Eliminates unnecessary activity, and as a result can reduce costs
* Empowers the development team to make decisions, which can also boost morale

LSD’s weaknesses include:

* Heavily depends on the team involved, making it not as scalable as other frameworks
* Depends on strong documentation, and failure to do so can result in development mistakes

2. Extreme Programming (XP) is an agile software development framework that aims to produce higher quality software, and higher quality of life for the development team. XP is the most specific of the agile frameworks regarding appropriate engineering practices for software development.

* Dynamically changing software requirements
* Risks caused by fixed time projects using new technology
* Small, co-located extended development team
* The technology you are using allows for automated unit and functional tests

3. **Adaptive Software Development** is a method to build complex software and system. ASD focuses on human collaboration and self-organisation

**1. Speculation:**  
During this phase project is initiated and planning is conducted. The project plan uses project initiation information like project requirements, user needs, customer mission statement etc, to define set of release cycles that the project wants.

**2. Collaboration:**  
It is the difficult part of ASD as it needs the workers to be motivated. It collaborates communication and teamwork but emphasizes individualism as individual creativity plays a major role in creative thinking. People working together must trust each others to

* Criticize without animosity,
* Assist without resentment,
* Work as hard as possible,
* Possession of skill set,
* Communicate problems to find effective solution.

**3. Learning:**  
The workers may have a overestimate of their own understanding of the technology which may not lead to the desired result. Learning helps the workers to increase their level of understanding over the project.

4.

Top of Form

Feature-Driven Development (FDD) is a client-centric, architecture-centric, and pragmatic software process. The term "client" in FDD is used to represent what Agile Modeling (AM) refers to as [project stakeholders](http://agilemodeling.com/essays/activeStakeholderParticipation.htm#Stakeholders) or [eXtreme Programming (XP)](http://agilemodeling.com/essays/agileModelingXP.htm) calls customers. FDD was first introduced to the world in 1999 via the book [Java Modeling In Color with UML](http://www.amazon.com/exec/obidos/ASIN/013011510X/ambysoftinc/), a combination of the software process followed by Jeff DeLuca's company and Peter Coad's concept of features. FDD was first applied on a 15 month, 50-person project for a large Singapore bank in 1997, which was immediately followed by a second, 18-month long 250-person project. A more substantial description is published in the book [A Practical Guide to Feature-Driven Development](http://www.amazon.com/exec/obidos/ASIN/0130676152/ambysoftinc/) as well as the [Feature Driven Development](http://www.featuredrivendevelopment.com/) web site.

As the name implies, [features](http://agilemodeling.com/artifacts/feature.htm) are an important aspect of FDD. A feature is a small, client-valued function expressed in the form

Bottom of Form

FDD also defines a collection of supporting roles, including:

* Domain Manager
* Release Manager
* Language Guru
* Build Engineer
* Toolsmith
* System Administrator
* Tester
* Deployer
* Technical Writer