Submitted By:

**Ayan Krishna Paul**

**Roll: 2017UCO1698**

**Scrum**

Scrum is one of many iterative and incremental software development agile process for managing product development. It is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value. It is based on multiple small teams working in an intensive and interdependent manner. The teams in the organization work together while constantly focusing on their common interest. Scrum has three roles:

* Product Owner
* Scrum Master
* Scrum Team

Product Owner

* The product owner creates a product backlog (essentially, a wishlist of tasks that need to be prioritized in a project).
* Decides release date and content.
* Prioritize features according to market value.
* Accept or reject work results.

Scrum Master

* The Scrum Master conducts daily meetings.
* Ensures that the team is fully functional and productive.
* Shield the team from external interfaces.
* Responsible for management of the project.

Scrum Team

* Typically 5 to 10 people.
* Teams are cross functional and self-organizing.
* Scrum team determines which items and in what order are to be executed.

**Advantages:**

* Large projects are broken down into mall sprints.
* Customers are involved, so best results are ensured.
* Budget friendly and time saving.

**Disadvantages:**

* Once decided the sprint backlog items cannot be changed.
* Requires strong commitment from all members of the team.

**Lean Development**

Lean Development is derived from the lean manufacturing principles. Lean development is the application of the lean manufacturing principles into a software development model. Robert Charette is the founder of LD. LD is the most strategic-oriented agile methodology. While most other agile methodologies are tactical in nature, in LD the major changes must be initiated from the top of the organization. LD’s goals are very stretch. An example would be one-third of the development time of an average organization. These aggressive goals challenges the managers to think about issues in entirely different ways.

7 Principles of Lean Development

* Eliminate Waste
* Amplify Learning
* Delay Commitment
* Empower the Team
* Deliver Fast
* Build Integrity in
* See the whole

7 wastes of software development

* Partial work
* Unnecessary features
* Additional features
* Mixing of tasks
* Waiting for development
* Difficulty in finding information
* Undetected errors

Delay Commitment

* Decision should be delayed until the last possible moment
* Benefits of delay commitment are
* More knowledge for decision
* More options for selection

**The LD Process**

The LD process has three high-level phases:

* Start-up
* Steady-state
* Transition and Renewal

**Contribution of LD**

* Strategic focus, a linkage to lean production, Risk entrepreneurship, stretch goals.

**Extreme programming (XP)**

 Extreme programming (XP) is a lightweight, efficient, low-risk, flexible, predictable, scientific, and fun way to develop a software. XP is designed to support small to medium sized teams that work under vague and rapidly changing requirements. It is called “Extreme Programming” because it takes 12 known proven software development principles and push them to extreme levels.

The five values of XP are:

* Communication
* Simplicity
* Feedback
* Courage and
* Respect

XP follows object oriented approach.

Activities in Extreme Programming:

* Coding
* Testing
* Listening
* Designing

The 12 key Practices of Extreme Programming:

* The Planning Game
* Small Releases
* SYSTEM Metaphor
* Simple Design
* Continuous Testing
* Refactoring
* Pair Programming
* Continuous Integration
* Collective Ownership
* On-site customer
* The 40-hour week
* Coding Standards

**Advantages:**

* Fewer documentation required.
* Collaboration with customers.
* Flexibility to developers.
* Easy to manage.

**Disadvantages:**

* Depends heavily on customer interaction.

**Adaptive Software Development (ASD)**

Adaptive Software Development (ASD) is a new software development methodology that focuses on rapid creation, application and evolution of software systems. It was derived from rapid application development (RAD). It has been proposed by Jim Highsmith as a technique for building complex software and system. It focuses on human collaboration and team self-organization.

The adaptive life cycle is divided into the following three main phases:

* Speculate
* Collaboration
* Learning

Speculate

* During speculation, the project is initiated and adaptive cycle planning is conducted.
* When we speculate, it’s not that we don’t define a mission to the best of our ability.

Collaboration

* Encompasses communication and teamwork but also emphasizes individualism.
* If we can’t predict, then we can’t control in the traditional management sense.
* If we can’t control, then a significant set of current management practices is no longer operable.

Learning

* Challenges all stakeholders.
* Examine their assumptions and use the results of each development cycle to learn the direction of the next.

Speculate- Collaboration- Learning

* For many project leaders and project teams, adaptive development is a terrifying prospect.

**Advantages:**

* Software incremental adjustment.
* Rapid and complex software development.

**Disadvantages:**

* There is lack of emphasis on necessary designing and documentation.

**Feature Driven Development**

Feature Driven Development (FDD) is an agile software development methodology. FDD employs a short-iteration model. It is designed to be used in large projects and teams. It combines many of the best practices of other agile methodologies.

FDD consists of a minimalist and five-step process that has the following steps:

* Develop an Overall Model.
* Build a Features List.
* Plan by Features.
* Design by Features.
* Build by Features.

For technical reviews, mandated code inspections are performed in FDD. Customer involvement is realized in the first step and in the fourth step. When the number of new features exceeds ten percent, the project manager asks top management.

FDD has two important roles:

* Chief architect and
* Chief programmer

Chief architect leads to draw the overall model and chief programmer leads the programmers through the feature sets. For a feature, there are here important parts:

* Action
* Result of the action
* Object related with the action

Feature Teams

Feature teams are temporary groups of developers formed according to the features. Usually there are multiple feature teams. A developer can take role in more than one feature teams.

Class Ownership

Class ownership is important in FDD. Every class belongs to a specific developer. Class owners have the responsibility for all changes in implementing new features. Quality classes are developed using class ownership. Class owners become expert for their classes.

**THE END**