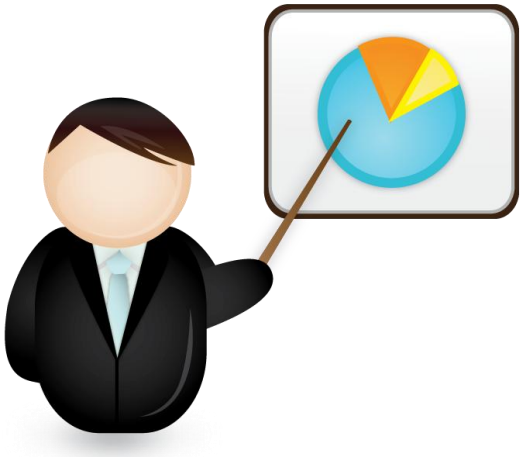
Distributed Agile Delivery Approach

By Ramana



About Trainer

- Working as Agile Practice Leader and RPA Manager
- Scrum Certified Trainer
- Having 11 Yrs Exp Test Manager
- Trained 10000 + Professionals

For further support, Please contact

B. Venkata Ramana

qtramana@gmail.com


Agenda!

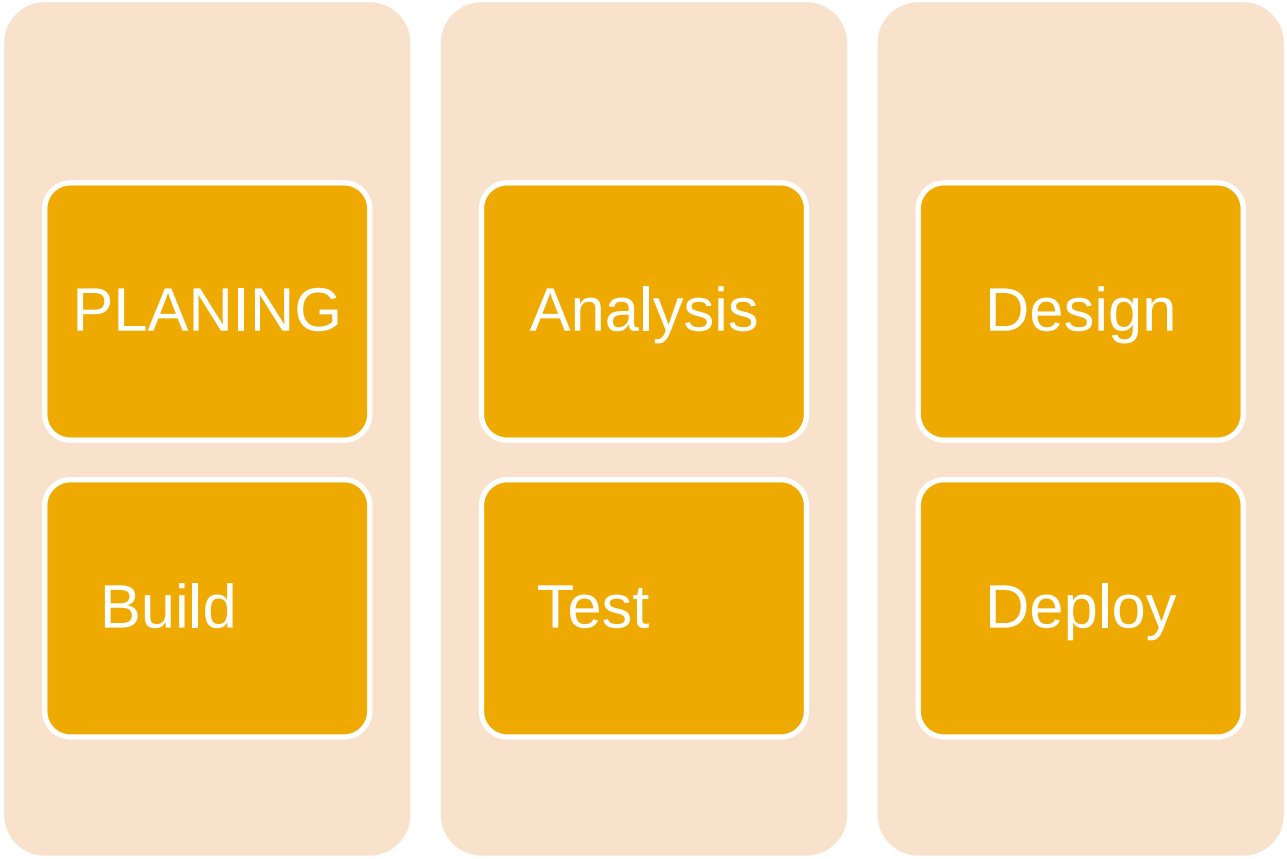
Agenda

- Basics of SDLC
- Plan-driven vs Agile approach
- Benefits and Challenges of Agile Approach
- Agile Manifesto
- Scrum Framework Introduction
- Scrum Roles
- Scrum Events
- Scrum Artefacts
- Advanced Agile
- JIRA-Agile Project Management Tool
- Scrum Framework vs Kanban Framework
- Advanced Agile Concepts

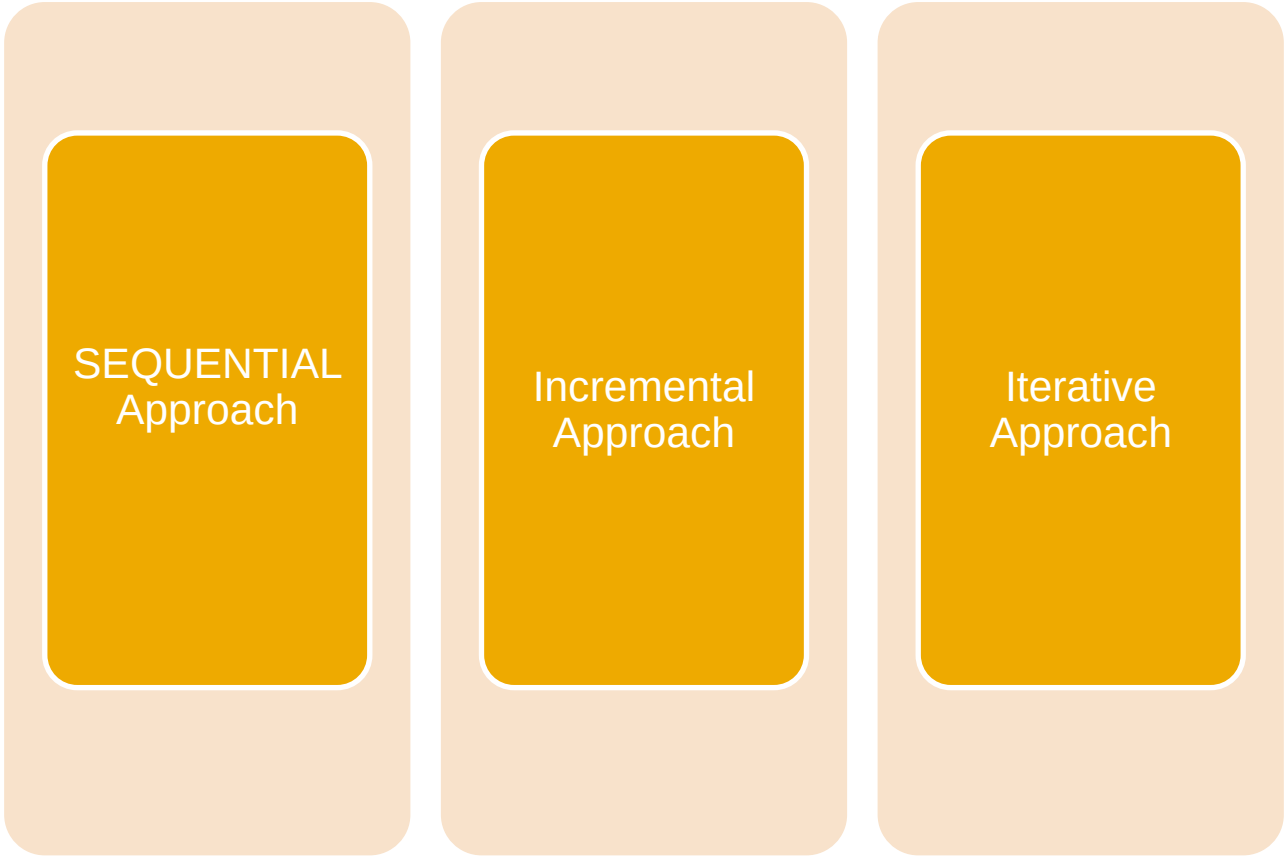
SDLC

(Software Development Life Cycle)

What is SDLC?



SDLC Approaches:



SDLC Models

Traditional Models

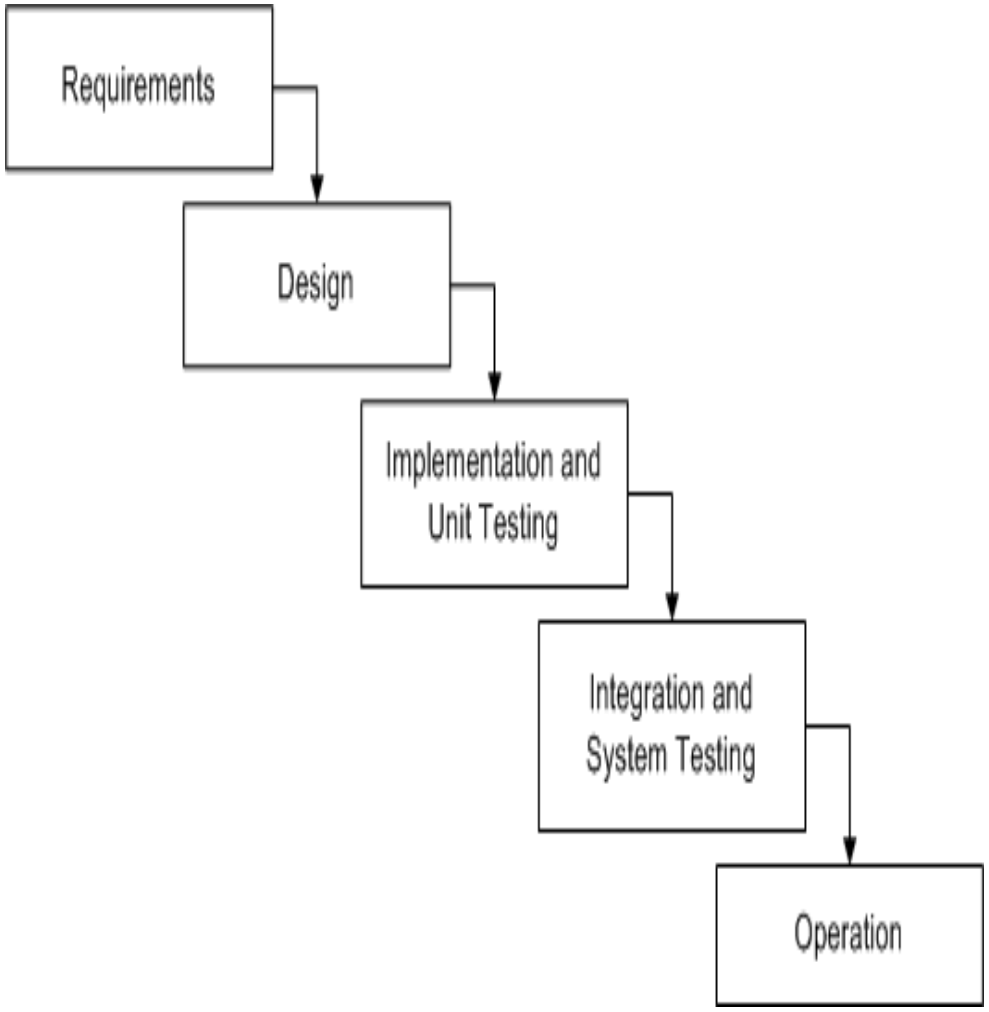
Ex:

Waterfall,

V - Model

Agile Models

Waterfall Model:



Disadvantages

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Poor model for complex and object-oriented projects.
- Poor model for long and ongoing projects. Works well for smaller projects where requirements are very well understood.
- Poor model where requirements are at a moderate to high risk of changing.
- Rework Would be more.

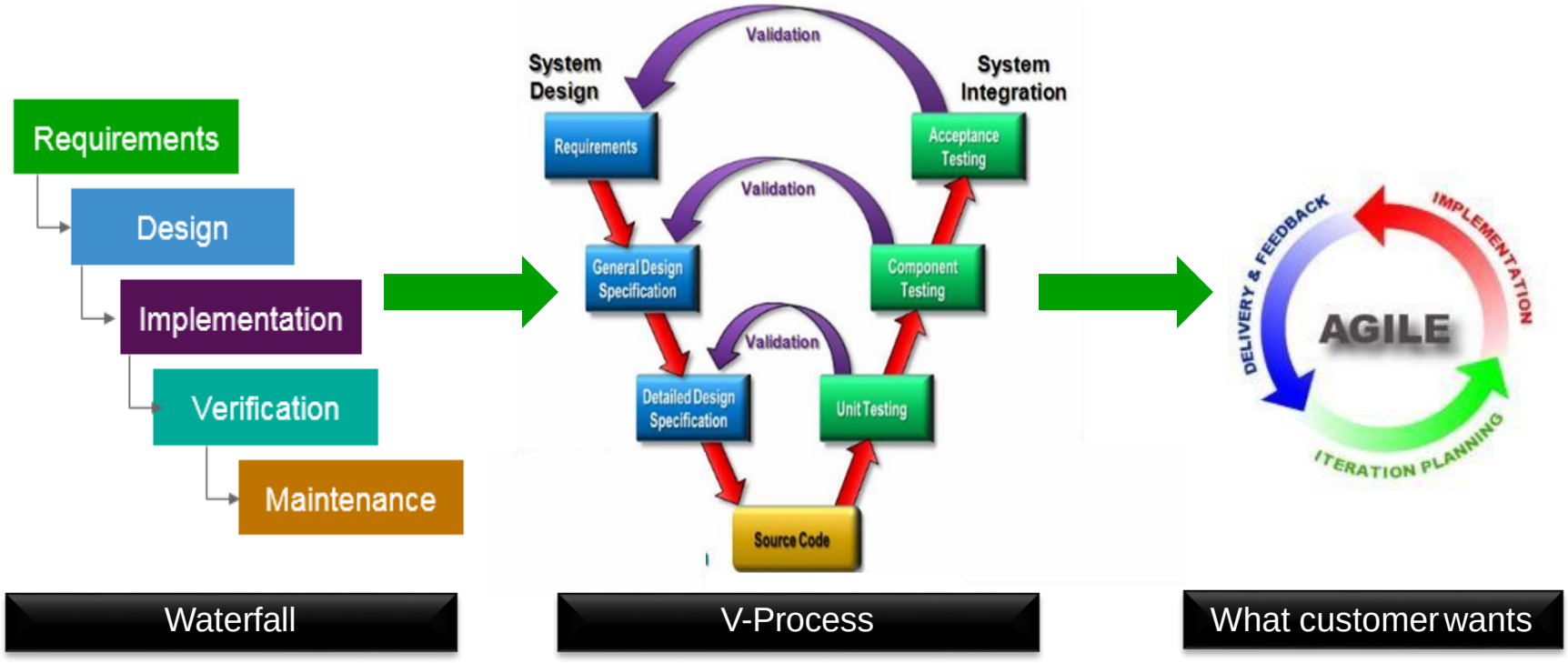
Iterative or Incremental Development Processes:

- Prototyping
- Rapid Application Development (RAD)
- Rational Unified Process (RUP)
- Agile Development


Agile

Why Agile?

- Traditional approaches like Waterfall, V-Process used sequential approach, and required lot of time to launch the product in the market.
- Due to the market dynamics, time to market and adapting to the changes has been very crucial



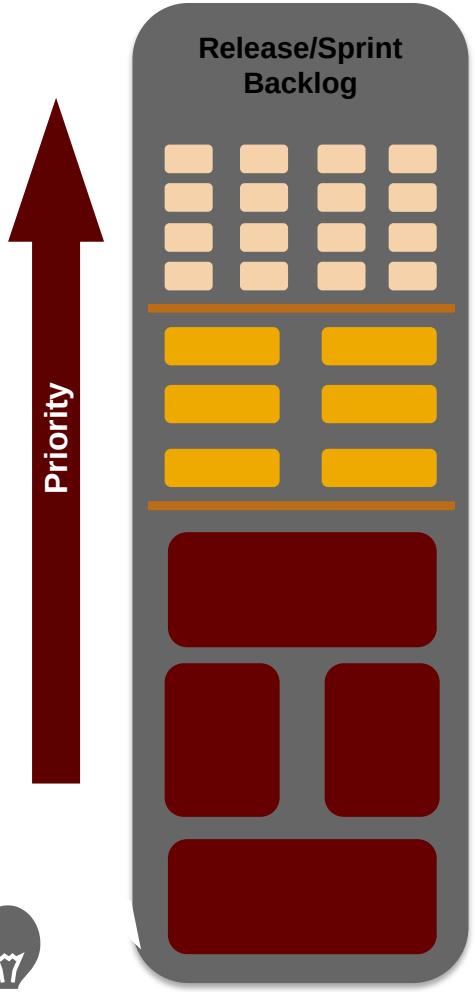
Agile Manifesto:

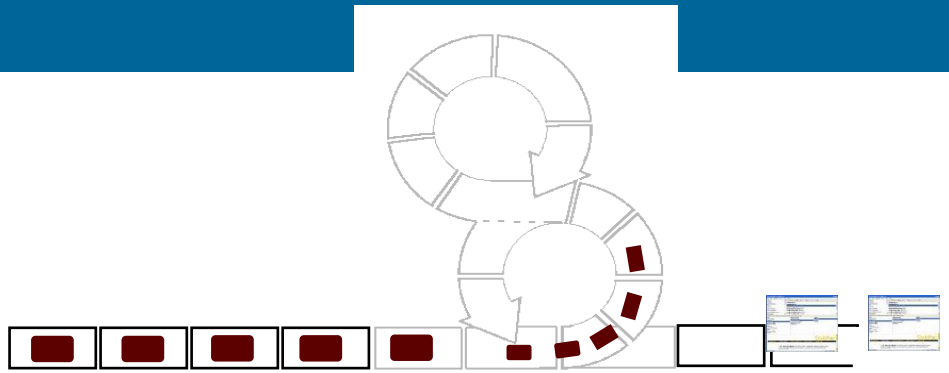
- Agile methods break tasks into small increments with minimal planning, and do not directly involve long-term planning.
- Iterations are short time frames ("time boxes") that typically last from one to four weeks.
- Each iteration involves a team working through a full software development cycle including planning, requirements analysis, design, coding, unit testing, and acceptance testing when a working product is demonstrated to stakeholders.
- Team composition in an agile project is usually cross-functional and self-organizing.
- Team members normally take responsibility for tasks that deliver the functionality an iteration requires.
- They decide individually how to meet an iteration's requirements..
- Most agile implementations use a routine and formal daily face-to-face communication among team members.

Conti…..

- Customer satisfaction by rapid, continuous delivery of useful software
- Working software is delivered frequently (weeks rather than months)
- Working software is the principal measure of progress
- Even late changes in requirements are welcomed
- Close, daily cooperation between business people and developers
- Face-to-face conversation is the best form of communication (co-location)
- Projects are built around motivated individuals, who should be trusted
- Continuous attention to technical excellence and good design
- Simplicity
- Self-organizing teams

Agile Delivery Lifecycle



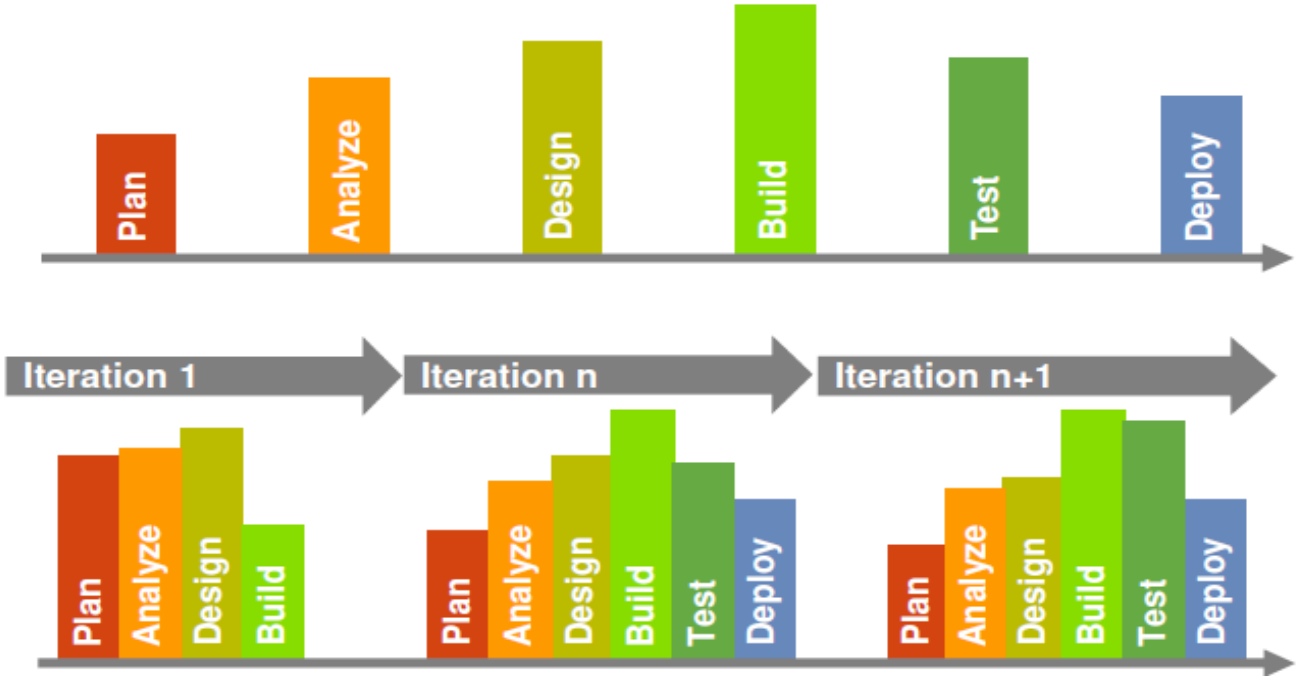


- Iterative Planning – progressively elaborate detailed analysis & planning of scope
- Re-prioritization based on discovery & analysis
- Time-boxed outcomes

What is demonstrable?

Plan-driven vs Agile approach

Agile Development is focused on an iterative (addressing all aspects of the lifecycle in each iteration) and flexible approach to software development



Plan-driven vs Agile approach

Plan-driven	Agile
Plan Driven	Learning Driven
Infrequent client communication	Continuous client communication
Big Bang releases (Typically 9-12 months)	Short releases of 3-6 months (focused on business priorities)
Requirements defined up-front	Requirements evolving throughout the project
Development in distinct phases with interim deliverables	Delivery of working code in short (2-4 weeks) iterations

Plan-driven vs Agile approach

Plan-driven	Agile
Develop in layers: Presentation, Business, Persistence etc.	Develop end-to-end functionality in iterations
View programming as Construction	View programming as Design
Integration of layers occurs at the end of build phase	Integrate code continuously
Testing occurs at the end	Testing occurs throughout the iteration at unit as well as functional level
High cost of change	Low cost of change

Benefits of Agile approach

- Agile provides for fast time to market doing **short releases to production**.
- The requirements are prioritized based on **the business values** and risks.
- Product is developed incrementally in **multiple iterations** →ability to address changing business requirements
- Testing is performed in each iteration.
- End-to-end testing in each iteration allows **critical defects to be identified** at the earliest point possible and enables the projects to schedule them based on priorities →**ability to address risks early**
- Each iteration produces a potentially shippable product.

→ **Deliver Business value and Test early and often**

Challenges of Agile Approach

Difficult to apply for large teams

• Agile methodologies depend on effective face-to-face communication and are, therefore, best suited for small teams.

Difficult to apply for distributed teams

• Since Agile methodologies place a strong premium on co-located faceto-face communication, distributed teams (both onshore and offshore) create significant challenges.

Require a cadre of responsible developers

• As teams are often self-organizing and empowered, more experienced developers with deeper skills are required.

Challenges of Agile Approach

Require an Agile champion

• Adoption of any methodology requires a champion to shepherd the process of adoption to ease the cultural change.

Some environments are not ready

• Some environments are not well suited for Agile methodologies – sometimes for cultural, historical, or political reasons. Realistically assessing an organization's readiness for Agile methodologies is key.

Agile Methods:

- Agile Modeling
- Agile Unified Process (AUP)
- Dynamic System Development Method (DSDM)
- Essential Unified Process (EssUP)
- Extreme Programming (XP)
- Feature Driven Development (FDD)
- Open Unified Process (OpenUP)
- Scrum