AGILE-PARADIGM SHIFT IN SDLC

Session - 6

Agile Software Development - I

Objectives

- Describe Agile development and its evolution
- Explain Agile SDLC
- Differentiate between Agile and other traditional SDLC models
- Explain the values behind the Agile manifesto
- Describe the key features of Agile software development
- Describe the most popular Agile software development methodologies
- List the applications of Agile software development
- Describe a case study on Agile

Introduction

Traditional Process Models

- Failure of clear understanding of requirements means ending up developing a piece of software that has less relevance with the actual need of the customer.
- To overcome some of these constraints, **AGILE Development Methods** are followed by businesses today.

AGILE

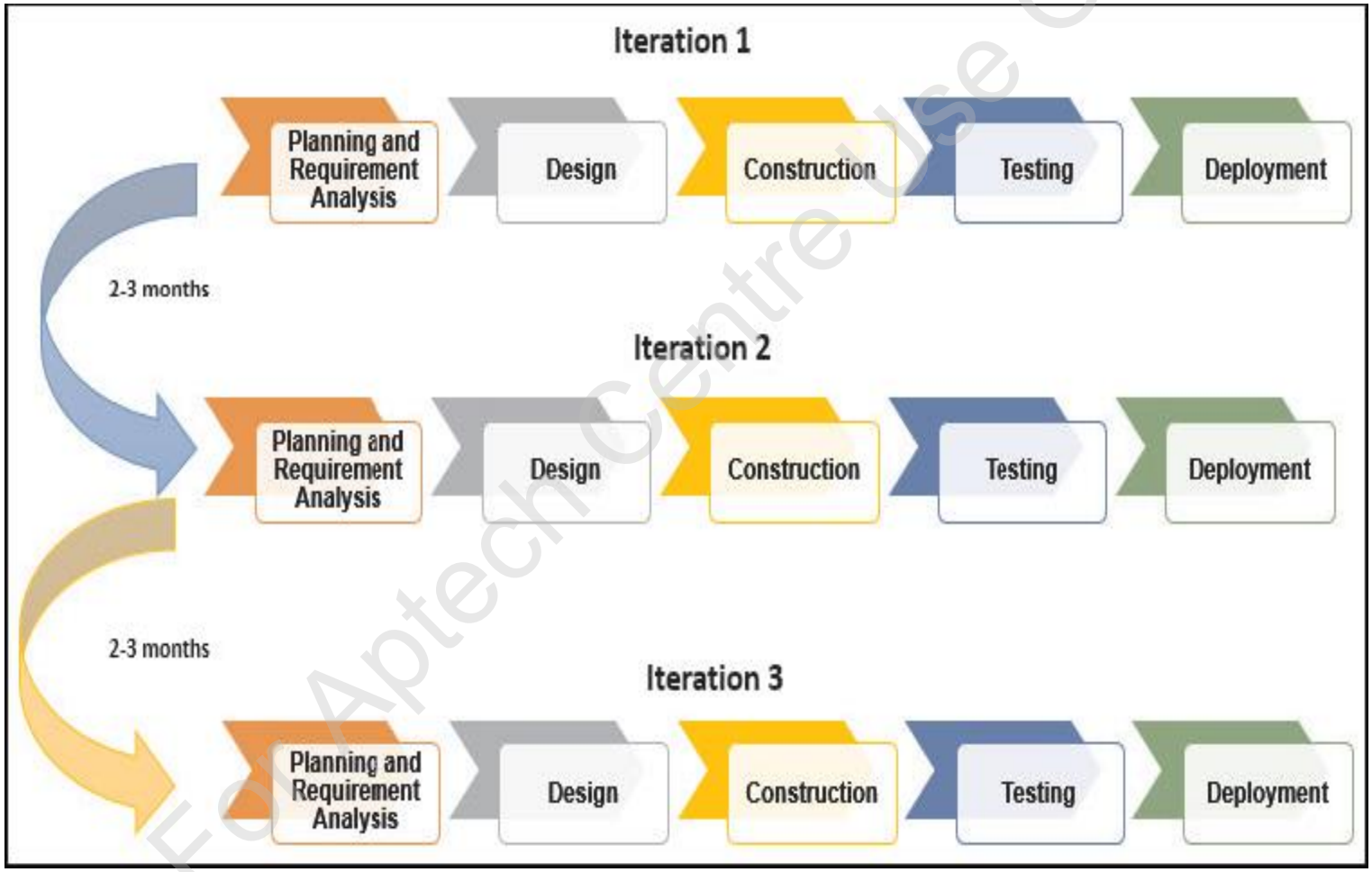
- Is the solution for all the eager business communities asking for a lightweight, fast, and easy to understand development processes.
- Reduces both development cost and time.

Evolution of Agile

- Agile methodologies are made up of several management, customer, and engineering practices.
- These practices put together, guides teams through the process of rapidly planning and delivering finished software products.
- Each aspect of development is continually revisited throughout the life cycle.
- The team stops and re-evaluates the direction of a project periodically, and if required changes the direction.
- The 'inspect-and-adapt' approach in Agile development reduces to a great extent both development costs and time to market.

Agile SDLC [1-2]

- Figure shows the Agile SDLC approach in software development.



Agile SDLC [2-2]

- Agile SDLC development follows the principle of increments and iterations to deliver software products.
- This method breaks down tasks into small increments with minimal planning.
- Iterations are of short time frames known as time-boxes.
- Iterative approach is taken to build software that can be delivered in each iteration.
- Each iteration adds or increments the software with the features laid by the customer.

Key Features of Agile SDLC [1-2]

Iterative	• Entire application is divided into fixed units called iterations and is a mini-increment of the functionality.
Active customer involvement	• There is a lot of client involvement and face-to-face interaction involved and each iteration is tested and approved by the client.
Feature driven	• The 80/20 rule is applied to decide the 20% features that would be used 80% of the time.
Fixed time	• There is a fixed time span for each of the iteration in which it is delivered.
Adaptive	• Agile is very adaptive and hence the developed application is able to accommodate the inflow of new requirements throughout its development with ease.
Empowered teams	• Agile teams are small and the interactivity level is high. As the entire team is actively involved, it is empowered to take decisions.

Key Features of Agile SDLC [2-2]

People centric	• Agile gives more importance to suitably-skilled people to do the development than on following the processes.
Priority based delivery	• Features are prioritized based on client requirement, development risk, and so on and the priorities are re-evaluated after each iteration.
Rapid development	• Agile projects use light-weight development technologies to complete the development rapidly.
Discipline	• Agile projects involve a lot of team and self-discipline as the deliverables have to be rapidly and accurately delivered which requires highly organized and skilled team members.
Simplicity	• In Agile projects, the emphasis is on being open to change and keeping things as simple as possible.

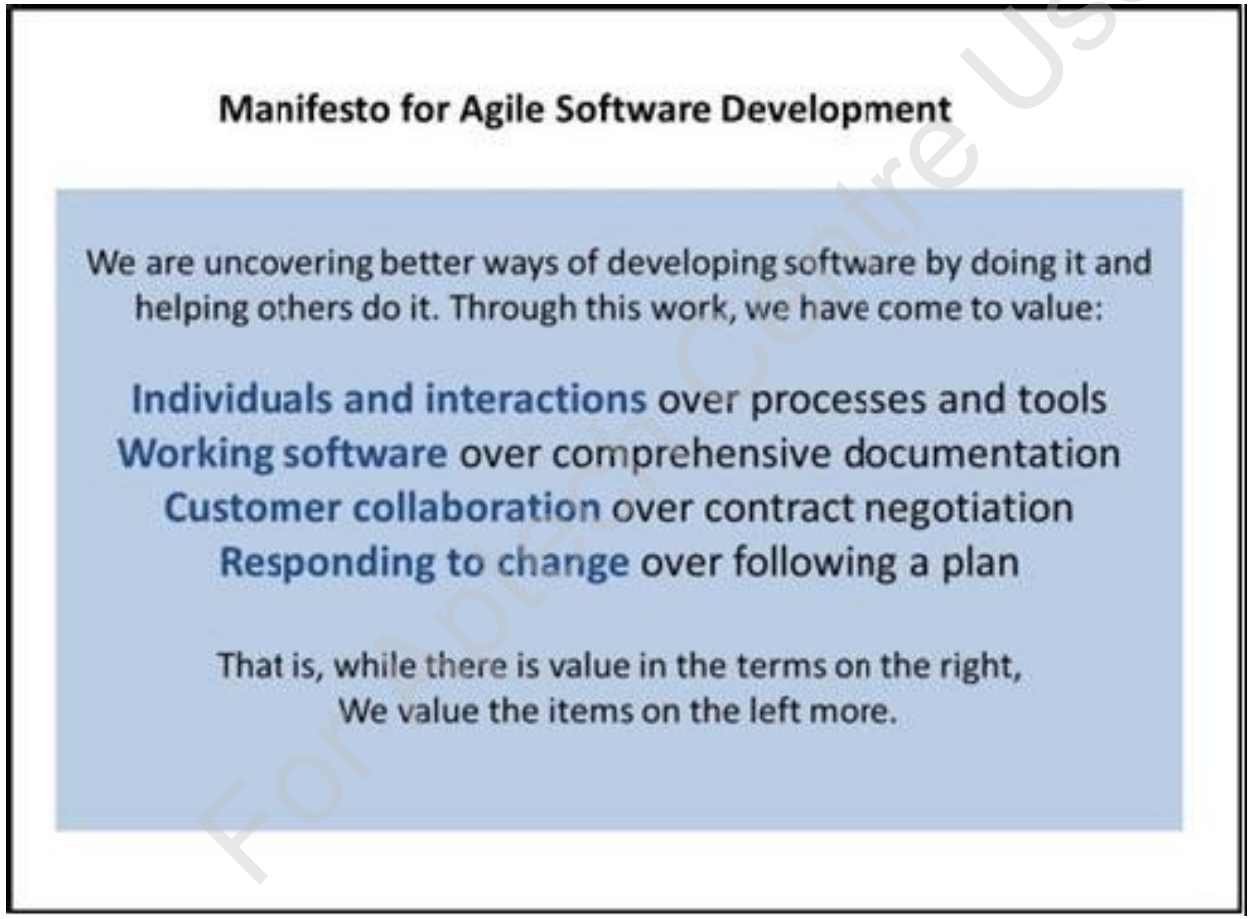
Comparing Agile with Traditional Models

- Table lists the differences between traditional software development and Agile.

Parameter	Traditional	Agile
Requirements	Fixed	Evolve during the lifecycle
Time and People	May vary	Fixed
Customer Involvement	Initially and at the end	During
Negotiable factors	Estimates	Schedule
Testing	At the end of development	In iterations
Feedback	After	During
Concentration on	Processes and Reviews	Working software
Focus	Plan-driven	Value-driven
Stages	Requirements, Design, Code, Test, and Feedback	Adaptive

Agile Manifesto

- The manifesto established a common set of values and principles for all Agile methodologies.
- Figure shows the four values provided by Agile manifesto.



Individual and Interactions Over Processes and Tools

- Is the first core value provided by Agile manifesto.
- States that the communication between the members of the team must be valued than involvement of processes and tools in project development.
- Rely on periodic inspect-and-adapt cycles and these work well only when team members exhibit the following key behavior:

Respect for each individual in the team.

Truth in all communications.

Transparency of all data, decisions, and actions.

Trust that each one of them in the team will show their involvement and contribute to their fullest.

Engagement in the team and working towards achieving the team's goals.

Working Software Over Comprehensive Documentation

Agile stresses on delivering small increments of working software to the customer at regular intervals.

Working software is more effective to attract the customer and eases the client's understanding than comprehensive documentation.

Customer Collaboration Over Contract Negotiation

Getting the customer involved is a very importance step in the success of projects.

Agile methodologies encourages having a customer representative work along with the development team.

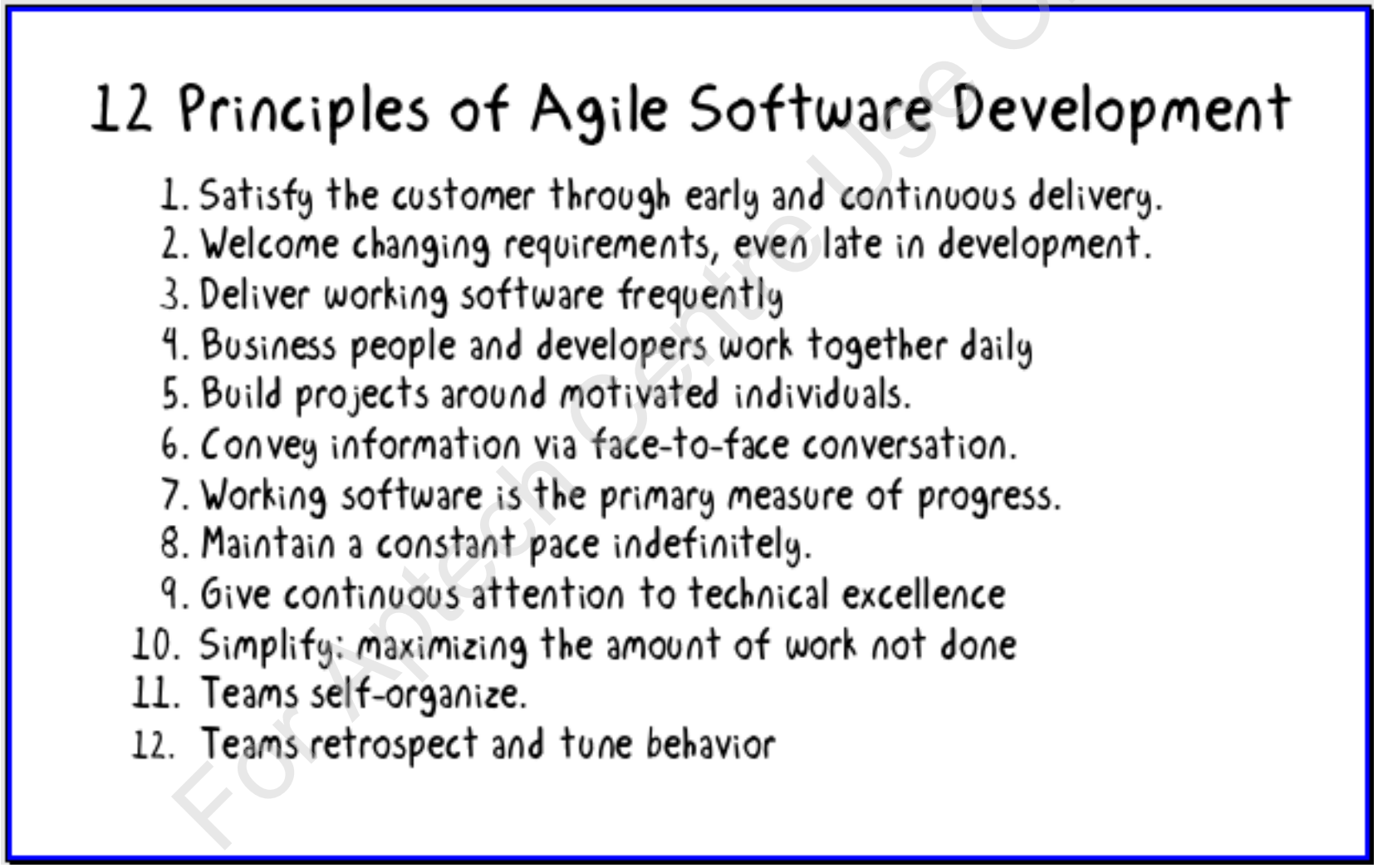
Daily customer collaboration has doubled the success rates of traditional projects worldwide.

Responding to Change Over Following a Plan

- To create products that provide business value and satisfy customers, teams must respond to change.
- All Agile methodologies have built-in processes to alter their plans periodically based on feedback from the customer or customer representative.
- These plans are designed to deliver the highest business value first.
- For this, there are processes, such as reviews that are designed to reconsider priorities regularly based on business value and customer feedback.

Principles of Agile

- Figure shows the 12 principles that have been laid by Agile manifesto.



Agile Methodologies [1-3]

- Some popular Agile software development methodologies also referred to as Agile process models are as follows:

Extreme Programming

- Extreme programming methodology promotes frequent releases in short iterations which improves productivity and enables new customer requirements to be accommodated.

Scrum

- Scrum follows an empirical approach accepting that problems cannot be fully understood or defined, and focusing on maximizing the ability of the team to deliver quickly and respond to new requirements.

Agile Methodologies [2-3]

Adaptive Software Development (ASD)

- ASD is used in business transactions involving considerable risks and continues team cooperation and learning loops to develop the application.

Feature Driven Development (FDD)

- Main goal of this methodology is to deliver working software periodically in fixed intervals and milestones are identified and progress is tracked based on achievement of milestone.

Lean Software Development

- Eliminate waste. Strengthen learning by doing and testing things rather than documenting and implementing them.

Agile Methodologies [3-3]

Agile Unified Process (AUP)	• Focuses on building up a model, implementing, testing, deploying, followed by configuration management. The target is on covering the high priority tasks rather than investing time on all possible things.
Crystal Group	• People-centric and emphasizes on enhancing the work of people. Focuses on efficiency and habitability as components of project safety.
Dynamic Systems Development Methods(DSDM)	• Based on RAD. Mainly used for projects which have a rigid schedule and budget.

Application of Agile Development [1-2]

Incremental and iterative approach is used in modeling.

Changes in requirements are acted upon and there is no requirement freezing.

Customers are actively involved in modeling.

Prioritization of tasks is done by the key stakeholders and the team works on the requirement of the highest priority.

The team members actively participate in the project.

The content of the model is given more importance than the representation or format.

Application of Agile Development [2-2]

Agile is not appropriate for projects having ample time for development.

Simple projects with repeated ideas need not use Agile.

Customers have limited or no involvement in the modeling or development phases.

Documents are to be prepared at each stage of the project and have to be signed off by the key stakeholders.

After a stage is completed and the associated documents are handed over the next team starts work on the project.

Case tool is used to specify the design of the software, but not to generate it.

Case Study [1-3]

- The case study discusses two start-up companies A and B, each with their own software development models. To make their processes leaner and more efficient, both companies decided to go off the beaten track.
  - **Company A** decided to adopt a team approach to software development. The entire team would use whiteboard to brainstorm and solve programming challenges together. It turned out to be more productive than using traditional methods of Computer-Aided Software Engineering (CASE) tools.
  - **Company B** decided to hire a chief software architect to come up with the design. The architect talked to each team member to get their inputs and designed the software on her own. He incorporated their suggestions and improvements in the next design cycle to produce better results.
- Both companies realized that process-wide documentation was not really helping them and in fact hampered their speed of development. They learned to separate design from documentation.

Case Study [2-3]

- **Company A** came up with their design faster as people worked parallely. As the development team participated in the design process, their design found wider acceptance and considered to be more robust.
- **Company B** succeeded in lowering costs the most as only one person, the architect was involved in the design process. However, as only one person was in charge, this approach could lead to bottle-necks while implementing projects.

Case Study [3-3]

- The learning from both these approaches are listed:
  - Developers are more important than processes, it is important to recognize the ideas and approaches of the development team for coming up with creative software solutions quickly.
  - The team should comprises technically sound experienced professionals.
  - Documentation is not as vital as previously believed. Although some amount of documentation is essential to any software development process, it should not be allowed to weigh down the process itself.
  - As the focus shifts away from documentation, the need for greater communication arises automatically. This in turn translates into more effective problem-solving.
  - While there are different modeling tools available, software design benefits from a more interactive approach involving multiple people.
  - Developers may need to use more than one tool for software development.
  - An iterative model allows developers to introduce improvements and fine-tune design in successive cycles.
  - Using different sources of solutions and ideas, such as using open source, reduces the effort spent on development, while making sure the focus is on innovation and not reinventing the wheel.

Summary [1-3]

- Agile software development is a software development methodology based on iterations increments in software development process.
- The keys of the Agile manifesto are:
  - Individuals and interactions over processes and tools
  - Working software over comprehensive documentation
  - Customer collaboration over contract negotiation
  - Responding to change over following a plan
- The features of Agile SDLC are:
  - Iteration
  - Active customer involvement
  - Feature-driven
  - Fixed time
  - Priority based delivery

Summary [2-3]

- Adaptive
- Empowered teams
- People centric
- Rapid development
- Discipline
- Simplicity

- Some of the most popular Agile software development methodologies are:
  - Extreme Programming
  - Scrum
  - Adaptive Software Development
  - Dynamic System Development Methods
  - Feature Driven Development
  - Lean Software Development

Summary [3-3]

- Agile Unified Process
- Crystal

- The learning from earlier efforts to make software development leaner, led to new approaches that later came to define Agile philosophy.
- Documentation is not as vital as previously believed. Although some amount of documentation is essential to any software development process, it should not be allowed to weigh down the process itself.
- Agile is not appropriate for projects having ample time for development.