

Introduction to Agile

Lesson 1: Agile Process Framework

▪ Software Development Model and SDLC

▪ “Waterfall Model” – An Overview

▪ “Waterfall Model” – Advantages

▪ “Waterfall Model” – Disadvantages

▪ Agile Development Model

▪ Graphical Illustration of Agile Development Model



▪ Why use Agile?

▪ Agile Manifesto and Principles

▪ 12 Principles of Agile Methods

▪ Agile Values

▪ What is NOT an Agile software development?

▪ Foundation of an Agile software development Method



▪ Common Characteristics of Agile Methods

▪ Agile Methods and Practices

▪ When to use Agile Model?

▪ Advantages of Agile Model

▪ Difference between Agile and Waterfall Model

▪ Agile – Myths and Reality

▪ Agile Market Insight

▪ Software development models are various processes or methodologies

used to develop the product

▪ Software developments models help improve the software quality as well as the development process in general

▪ There exists various software development models and each one of them fulfill certain objectives of software development

▪ Software Development Life Cycle (SDLC) is an environment that

describes activities performed in each stage of the software development process

▪ The SDLC contains detailed plan which basically describes how the development and maintenance of specific software is conducted

▪ Most people involved with software development are very much familiar with the traditional software development methods like:

• Waterfall or the sequential method

• V-model

▪ The classic waterfall model was introduced in the 1970s by Win Royce

▪ The Waterfall Model was the first Process Model to be introduced

▪ It is also referred to as a linear-sequential life cycle model

▪ The waterfall model is a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards like a waterfall through the phases of SDLC

▪ Every stage has to be completed separately at the right time so you can

not jump stages

▪ Documentation is produced at every stage of a waterfall model to allow people to understand what has been done

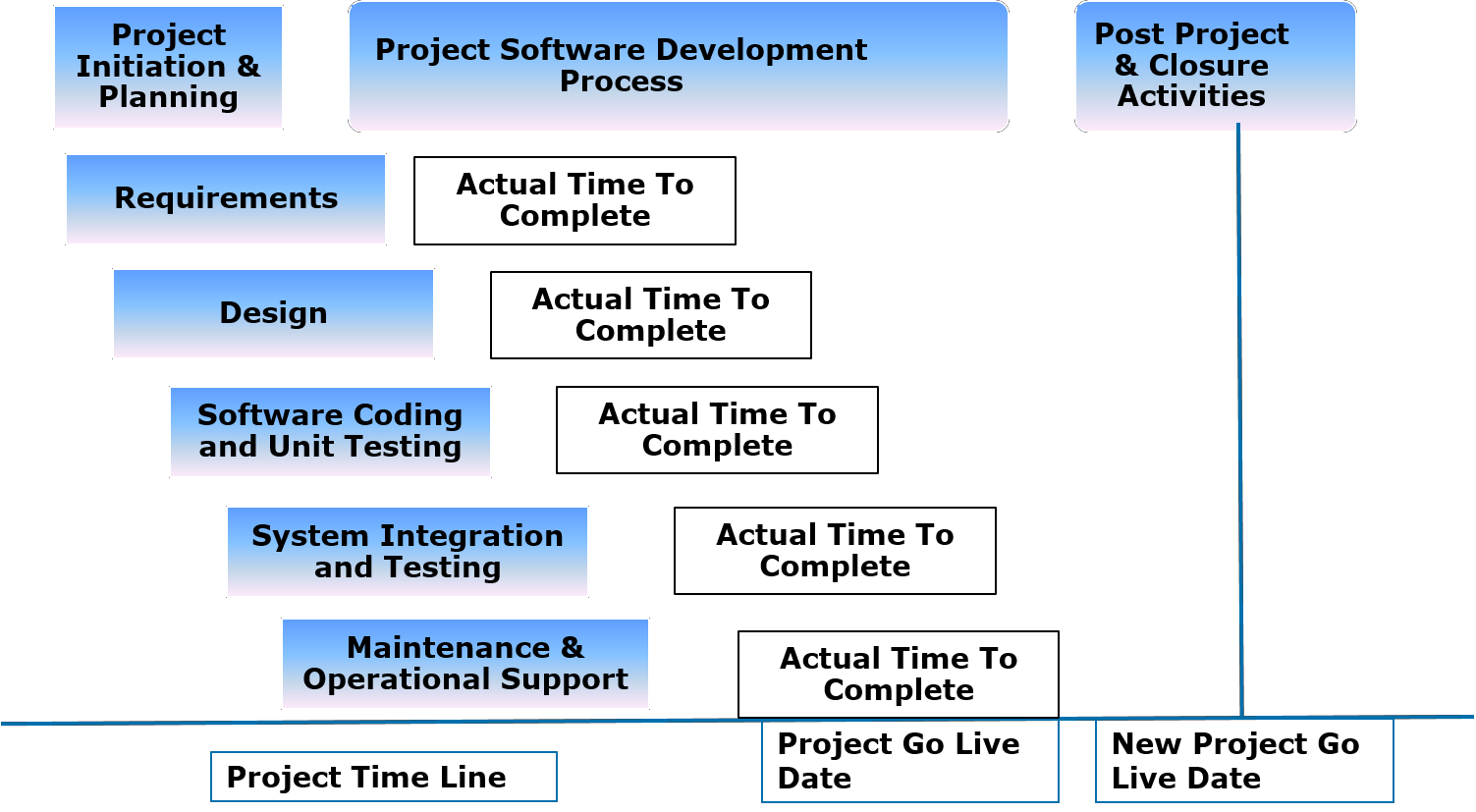
▪ Testing is done at every stage

▪ The waterfall approach assumes that requirements are stable and frozen

across the project plan

▪ However, this is usually not true in case of large projects where requirements may evolve across the development process

“Real Life” – Waterfall Model



▪ Simple and easy to understand and use

▪ Easy to manage due to the rigidity of the model, each phase has specific

deliverables and a review process

▪ The project requires the fulfilment of one phase, before proceeding to next

▪ Works well for smaller projects where requirements are very well

understood

▪ Various stages of the software development can be clearly defined in waterfall model

▪ Well understood milestones

▪ A schedule of activities can be created with deadlines for each stage of

development

▪ Product development progresses from vision, through design, implementation, testing, and ends up at operation and maintenance

▪ Each phase of development proceeds in strict order

▪ Once an application is in the testing stage, it is very difficult to go back

and change something that was not well-thought out in the concept stage

▪ The customer can experience the working model of the product only at the end

▪ Not suitable for complex & large projects

▪ Only a certain number of team members will be qualified for each phase,

which can lead at times to some team members being inactive

▪ It is difficult to follow the sequential flow in software development process

▪ Agile development model is an amalgamation of iterative and incremental process models focusing more on process adaptability and customer satisfaction by rapid delivery of functional software product

▪ Agile development model breaks the software into small incremental builds

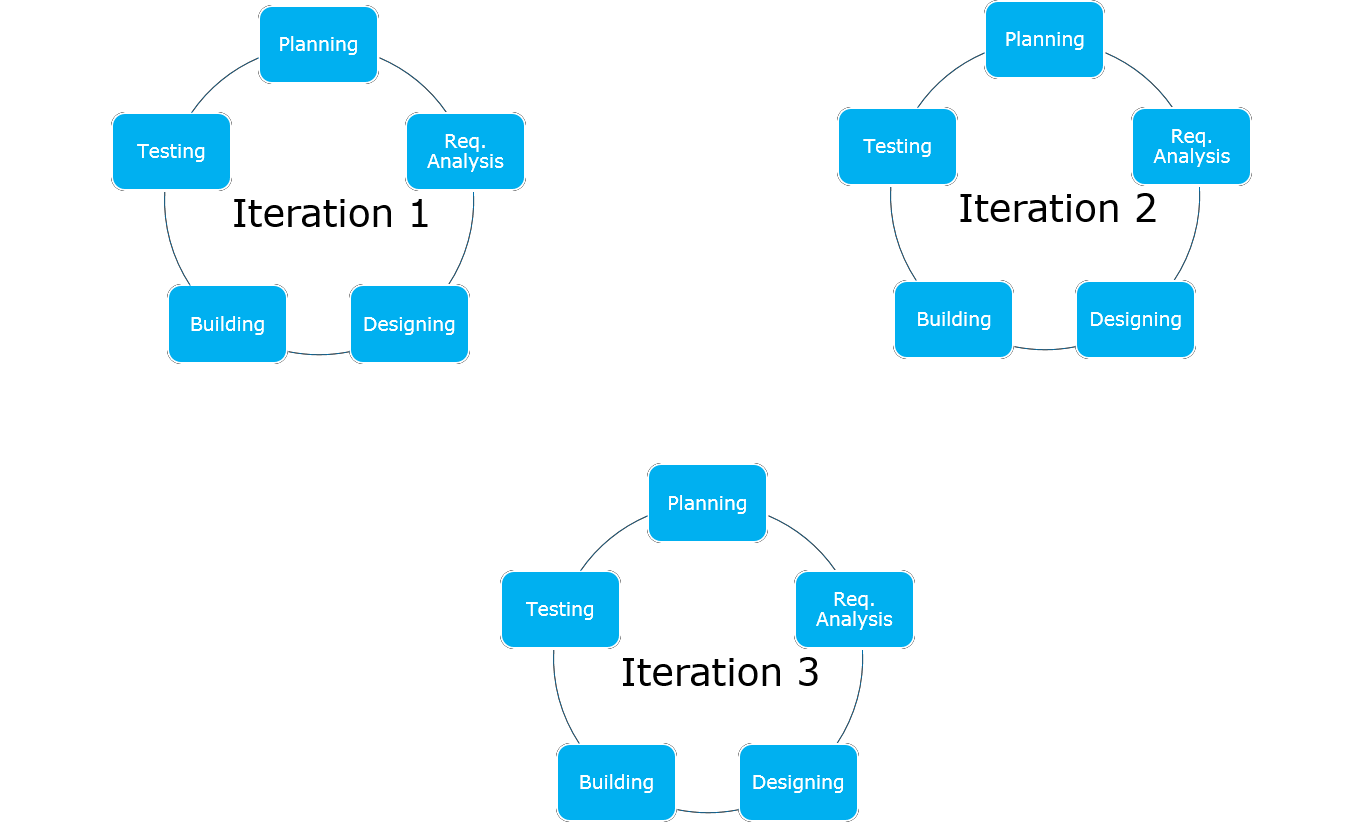
▪ These builds are provided in iterations

▪ Each iterations lasts from about one to three weeks

▪ Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing

▪ At the end of the iteration a working product is displayed to the customer and important stakeholders

Graphical Illustration of Agile Development Model



▪ Improved return on investment (RIO)

▪ Early detection and cancellation of failing products

▪ Higher quality software

▪ Improved control of a project

▪ Reduced dependence on individuals and increased flexibility

***Individuals and interactions* over processes and tools**

***Working software* over comprehensive**

**documentation**

***Customer collaboration* over contract negotiation**

***Responding to change* over following a plan**

▪ According to Kent Beck, the Agile Manifesto is based on twelve

principles:

• Customer satisfaction by rapid delivery of useful software

• Welcome changing requirements, even late in development

• Working software is delivered frequently (weeks rather than months)

• Working software is the principal measure of progress

• Sustainable development, able to maintain a constant pace

• Close, daily cooperation between business people and developers

• Face-to-face conversation is the best form of communication (co-location)

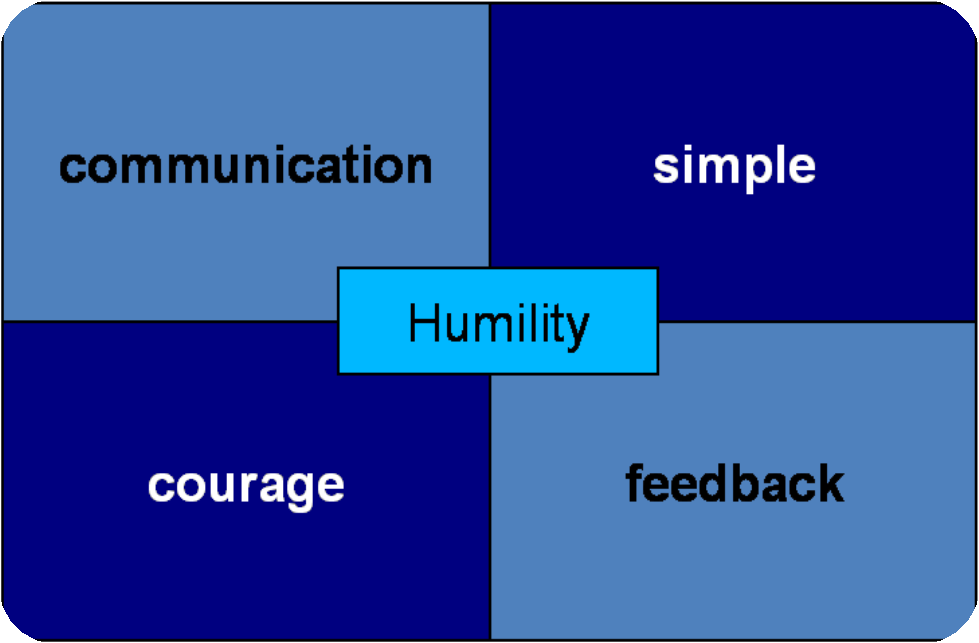
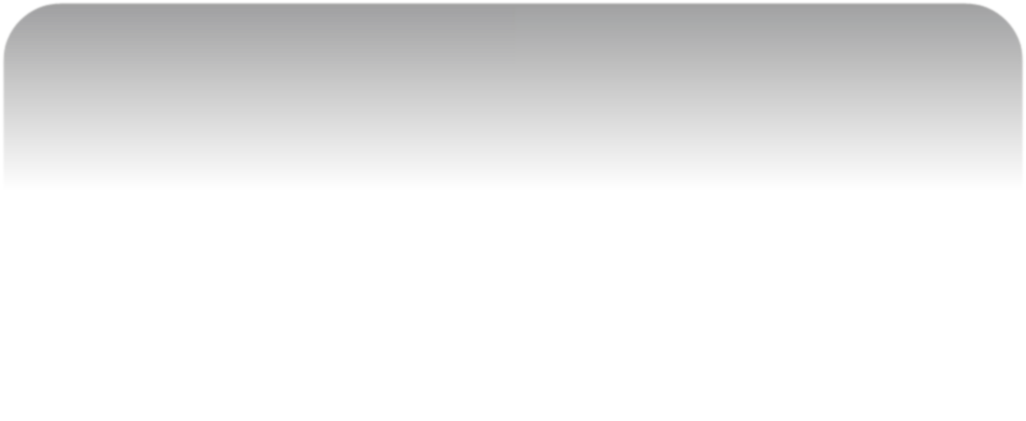
• Projects are built around motivated individuals, who should be trusted

• Continuous attention to technical excellence and good design

• Simplicity—the art of maximizing the amount of work not done - is essential

• Self-organizing teams

• Regular adaptation to changing circumstances



Agile Values

What is NOT an Agile software development?

▪ Compressing the project schedule

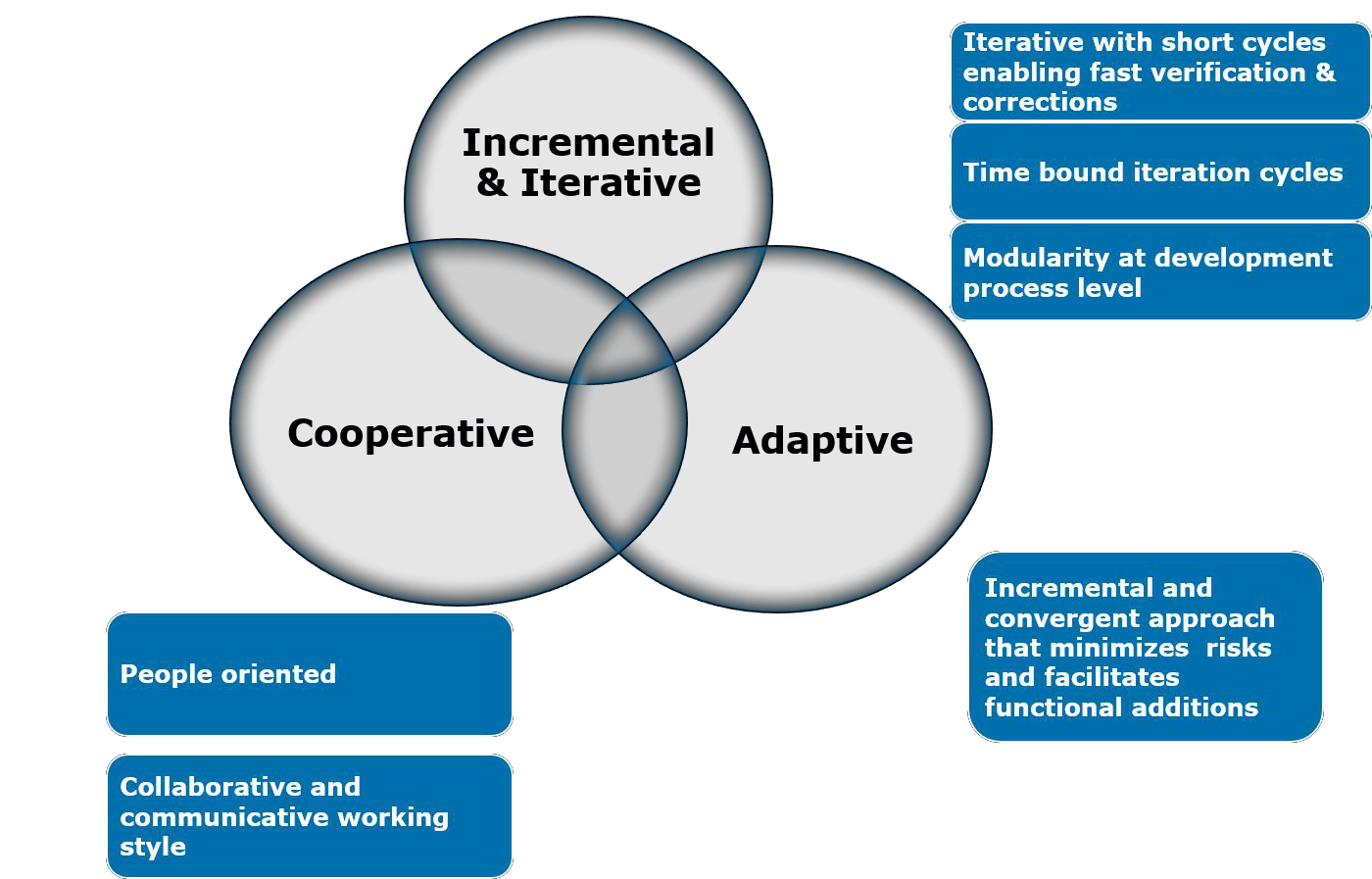
▪ Eliminating all existing software development models

▪ Eliminating all documentation

▪ Writing code up to the last minute

▪ An excuse for doing nothing

Common Characteristics of Agile Methods



▪ Scrum - Ken Schwaber, Jeff Sutherland, Mark Beedle

▪ Extreme Programming (XP) - Kent Beck, Eric Gamma, and others

▪ Dynamic System Development Method (DSDM) - Dane Faulkner

And others

▪ Agile Unified Process (or Agile RUP) - Scott Ambler

▪ Feature Driven Development - Peter Coad and Jeff Deluca

▪ Lean Software Development - Mary and Tom Poppendieck

▪ Kanban - David Anderson

▪ This model can be followed when:

• New changes must be implemented. The freedom agile gives to change is very important. New changes can be implemented at very little cost because of the frequency of new increments that are produced.

• To implement a new feature, the developers need to lose only the work of a few days, or even only hours, to roll back and implement it.

• Unlike the Waterfall Model, in the agile model, limited planning is required to get started with the project. Agile assumes that the end users’ needs are ever changing in a dynamic business and IT world. Changes can be discussed and features can be newly effected or removed based on feedback. This gives the customer the finished system they want or need.

• Both system developers and stakeholders alike, find that they also get more freedom of time and options than if the software was developed in a more rigid, sequential manner. Having options gives them the ability to leave important decisions until more or better data or even entire hosting programs are available; meaning the project can continue to move forward without fear of reaching a sudden standstill.

▪ Is a very realistic approach to software development

▪ Promotes teamwork and cross training

▪ Functionality can be developed rapidly and demonstrated

▪ Resource requirements are minimum

▪ Suitable for fixed or changing requirements

▪ Delivers early partial working solutions

▪ Good model for environments that change steadily

▪ Minimal rules, documentation easily employed

▪ Enables concurrent development and delivery within an overall planned context

▪ Little or no planning required

▪ Easy to manage

▪ Gives flexibility to developers

**Agile**

• Software development lifecycle is

carried out in the form of Sprints

• Agile method proposes incremental and iterative approach to software design

• It follows an incremental approach

towards solution development

• Agile methodology is known for its flexibility

**Waterfall**

• Software development process is

divided into distinct phases

• Development of the software flows sequentially from start point to end point

• It follows linear, sequential design approach towards software development

• Being a traditional software

• Agile can be considered as a collection of many different projects

development model, Waterfall exhibits characteristic of a structured model so most of the times it can be very rigid

• Software development will be

completed as one single project

**Myth**

• No Documentation

• Undisciplined

• No Planning

• Not Predictable

• Is a Fad

• Silver Bullet

• RUP isn’t agile

• Not Fixed Price

**Reality**

• Agile Documentation

• Requires great discipline

• Just-in-time (JIT) planning

• Far more predictable

• It’s quickly becoming the norm

• It requires skilled people

• RUP is as agile as you make it

• Agile provides stakeholders control over the budget, schedule, and scope

Summary

▪ In this lesson, you have learnt



• Various conventional software development models like

Waterfall Model, SDLC & V-Model

• The traditional software development models like Waterfall Model, V-Model are classified into the heavyweight methodologies

• These methodologies are based on progressive series of steps like requirements definition, design and architectural planning, development and testing

• Every traditional software development model has its own advantages and disadvantages

• We need to select the software development model which best suits our project requirement

• Introduction to Agile

• Agile Manifesto and Principles

• Agile Values

• Agile – Myths and Reality