From Fragile to Agile: Practical Approaches to Adopting Agile

- Explore agile software development methodologies and approaches
- Understand differences between agile and traditional methodologies
- Learn how agile practices and principles improve the software development process
- Discover the major steps required to successfully plan and execute an agile software project
- Explore the leading agile development best practices

Organizations today are seeking ways to improve the efficiency of their software development efforts while still meeting quality objectives. Competitive pressures and customer demands continue to reduce software product release schedules, driving organizations to seek fresh new approaches to building software. Agile software development methods are often cited as a way to accelerate software delivery and get more done with less. This course will teach you how to avoid the common mistakes of agile adopters and answer some of the familiar myths and misuse:

- Believing that agile says you don't need to document your software
- Forgetting about doing unit testing as part of the development process
- Adopting Scrum without adopting appropriate agile development practices as well
- Believing that agile means you don't need to do architecture or design when building software

This course presents a roadmap for how to get started with agile along with practical advice. It will introduce you to agile software development concepts and teach you how to make them work. You will learn what agile is all about, why agile works, and how to effectively plan and develop software using agile principles. A running case study allows you to apply the techniques you are learning as you go through the course. Key concepts that will be introduced and discussed include:

- Managing requirements using agile
- Defining and estimating user stories
- Building a release plan
- Using Scrum-based project management
- Delivering software using Extreme Programming

Bring your specific issues and problems to the training course for discussion as well.

Coveros recommends this class both as a prerequisite for those seeking ICAgile's Agile Testing certifications and those practitioners who recognize the need to focus on "being" agile in addition to "doing" agile.

Who Should Attend

The audience includes software developers, software test professionals, project managers, business analysts, product managers, line of business owners. No specific prerequisites are assumed, however, attendees are expected to have some software experience.

Course Completion and Certification

Upon completion of this course the attendee will be certified by the International Consortium for Agile (ICAgile) and awarded the ICAgile Professional (ICP) designation. Additionally, the certified attendees will be listed on the ICAgile website, indicating their Professional designation and that they have completed all the learning objectives associated with the Fundamentals of Agile track. *The ICAgile certification fee is included with your registration for your convenience.*

About the ICAgile

The International Consortium for Agile's goal is to foster thinking and learning around agile methods, skills, and tools. The ICAgile, working with experts and organizations across agile development specialties, has captured specific learning objectives for the different agile development paths and put them on the learning roadmap. For more information visit www.icagile.com [1].

Course Outline

Introduction to Agile
What is agile?
Benefits of agile
Why does agile work?
Myths about agile
Who is using agile?

Agile Software Process
Overall agile development process
Agile best practices

Agile Planning
Introduction to Scrum
The planning process
-Backlogs
-Initial release planning
-Iterative sprint planning
Roles during initial planning
Building good user stories
Estimating work
Building a release plan
Key meetings and activities
-Sprint kickoff
-Daily Scrums
-Sprint planning
-User acceptance testing and reviews
-Retrospectives

Roles During Sprints

Agile Development Best Practices
Team-based design
Pair programming
Continuous integration
Test-driven development (TDD) and unit testing
Refactoring

Agile Testing Best Practices
Agile testing framework
Acceptance test-driven development (ATDD)
Exploratory testing
Agile test automation

Wrap Up Discussion