Fundamentals of Agile Webinar

Trainer

Jeffery Payne

Jeffery Payne is CEO and founder of Coveros, Inc., a software company that helps organizations accelerate the delivery of secure, reliable software. Coveros uses agile development methods and a proven software assurance framework to build security and quality into software from the ground up. Prior to founding Coveros, Jeffery was Chairman of the Board, CEO, and co-founder of Cigital, Inc. Under his direction, Cigital became a leader in software security and software quality solutions, helping clients mitigate the risk of software failure. Jeffery is a recognized software expert and popular speaker at both business and technology conferences on a variety of software quality, security, and agile development topics. He has also testified before Congress on issues of national importance, including intellectual property rights, cyber-terrorism, Software research funding, and software quality.

Agenda

- Definition of Agile Software Development
- History of Agile Development
- Agile Manifesto
- Principles of Agile
- Overview of Fundamentals of Agile Course

Definition of Agile Software Development

- Agile Development is an approach to software development that recognizes that building software is much more of a creative process than a manufacturing process
  - *Adaptive* – Embrace change as it is inevitable
  - *Iterative* – Focus on building working software
  - *Collaborative* – Constant team and customer communication
  - *Productive* – Continuous improvement process

- Agile principles / practices have been applied to business activities beyond software development
  - Program / project management
  - IT management
  - Executive management
  - Product development

Agile Practices Have Been Around for a LONG Time

> Much of present-day software acquisition procedures rests upon the assumption that one can specify a satisfactory system in advance. . . . I think **this assumption is fundamentally wrong**, and that many software acquisition problems spring from that fallacy. --Fred Brooks, author of *Mythical Man Month*, 1986

- Agile methods picked up steam in the 1990's as many became disheartened with on-going software project failures
- 'Agile' term was coined at 2001 at a Snowbird, Utah workshop where the *Agile Manifesto* was created

Agile Manifesto

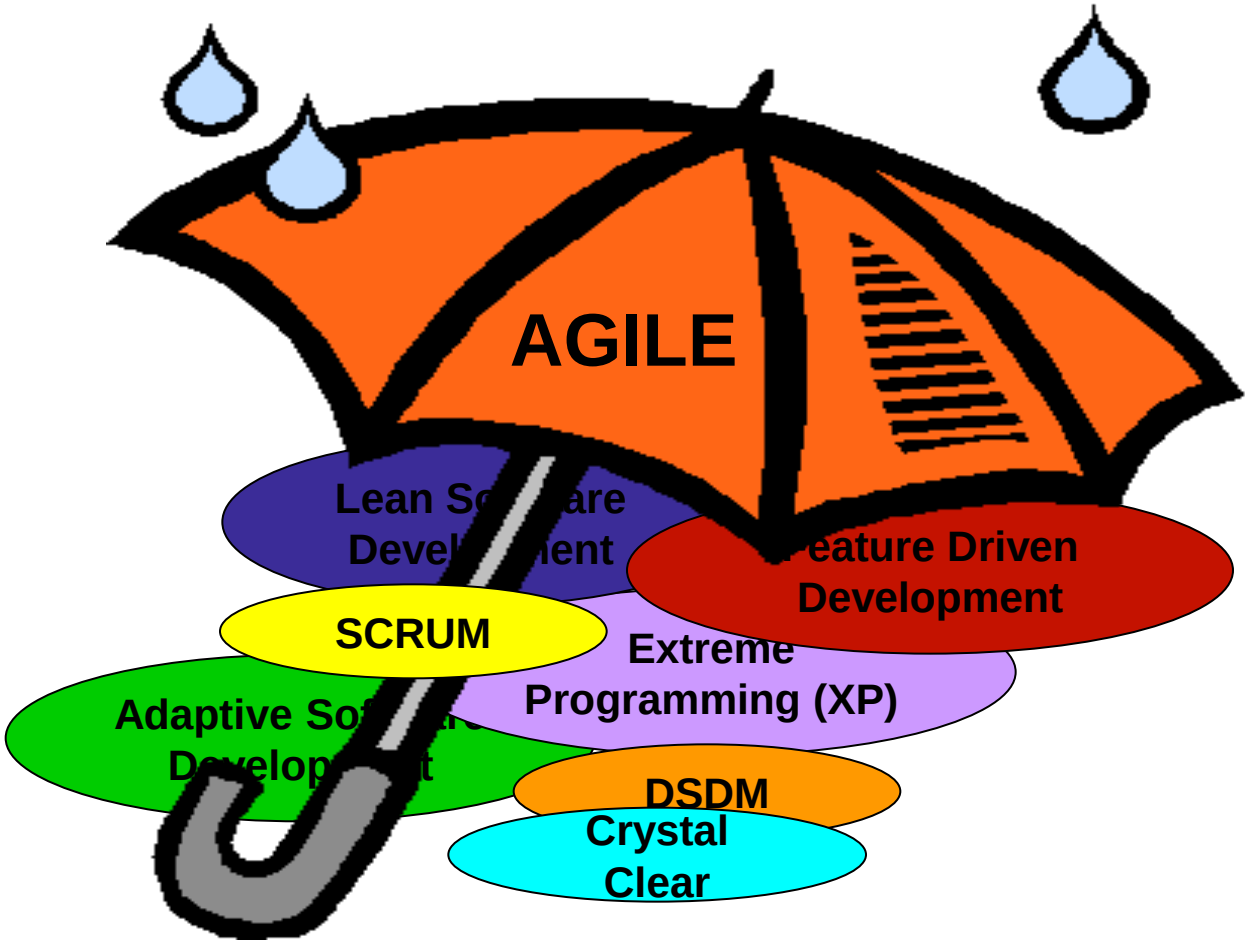
"We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

1. Individuals and interactions over processes and tools
2. Working software over comprehensive documentation
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more."

What is Agile?

Agile is not one set of practices … it is a set of principles



http://www.agilemanifesto.org

Agile Principles

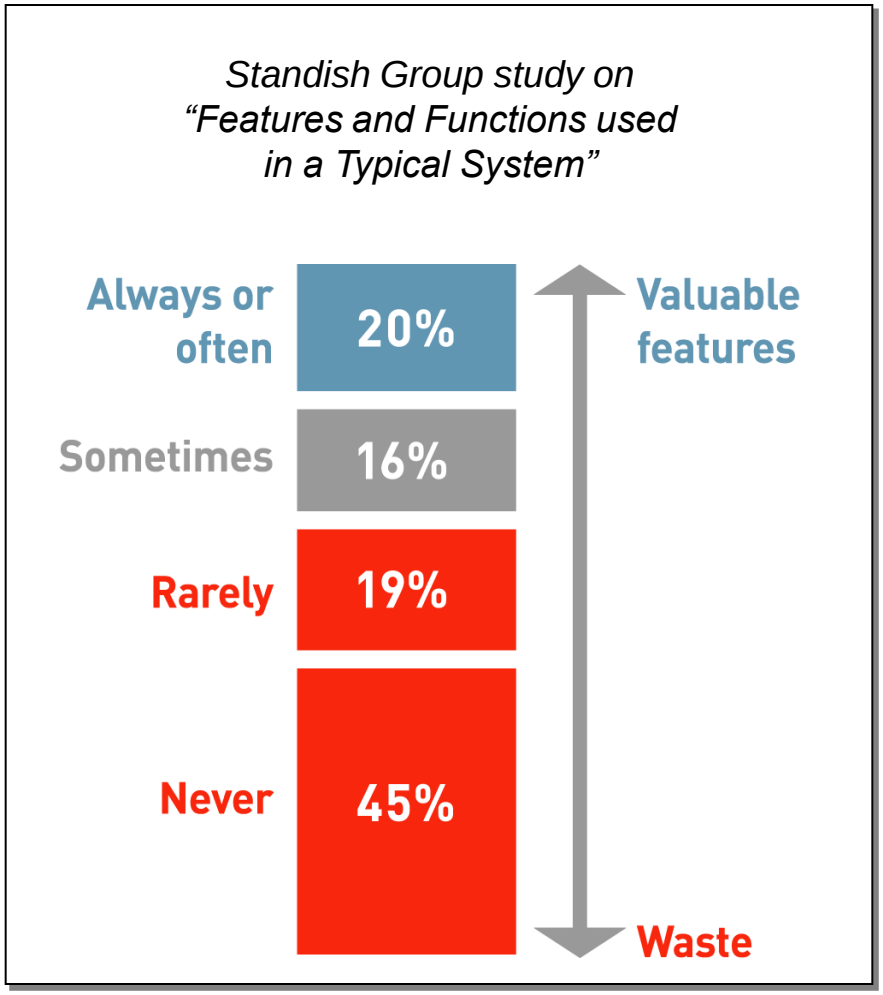
- **Customer value** is the number one priority
- **Embrace change**, even late in development
- Early and continuous delivery of **working software**
- **Frequent conversation** to convey information efficiently
- Working software as the **primary measure of progress**
- **Sustainable** development
- Continuous attention to **technical excellence** and good design
- **Simplicity** – maximizing the amount of work not done
- **Self-organizing** teams
- At regular intervals, the **team reflects on, tunes, and adjusts** its behavior

Agile Principles: Customer Value is #1

- Why?
  - Value to customers will make or break product success
  - To prioritize work so everything isn't the most important
  - Technologists like to build "cool things" using "cutting-edge technology", often to the detriment of customers

- How is this achieved?
  - Requirements are prioritized based upon customer value
  - Customers are part of the entire planning and development process (either as part of the team or through a customer proxy)
  - User acceptance testing provides a tight customer feedback loop
  - Working software is delivered after each Sprint

Why Projects Fail: Poor requirements management



Agile Principles: Embrace Change

- Why?
  - Change cannot be controlled so it must be expected and dealt with
  - Change is inevitable so there is no reason not to embrace it
  - Change is often necessary within the team based upon context

- How is this achieved?
  - Detailed requirements planning only for those features that are inevitable (and of highest value!)
  - Continuous customer feedback on product functionality and capabilities to identify changing requirements early
  - Continuous integration to accelerate Sprints
  - Fixed time / team, variable scope project planning

Agile Principles: Continuous Software Delivery

- Why?
  - Delivering continuous working software guarantees progress is made
  - Working software allows users to provide constructive feedback based upon a tangible product
  - Puts the product release decision into the hands of the business

- How?
  - Short Sprints that deliver working software
  - User acceptance testing at the end of each Sprint
  - Comprehensive testing of all features / functions developed

Addressing these Issues with Agile Software Delivery

- Deliver incremental change in order to maximize feedback.
- Accept change continuously in order to minimize waste.

THE AGILE BET

If the cost of change can be kept low over time, the cost savings that result from **early feedback will far** outweigh the added costs of early change.

Agile Principles: Frequent Communication

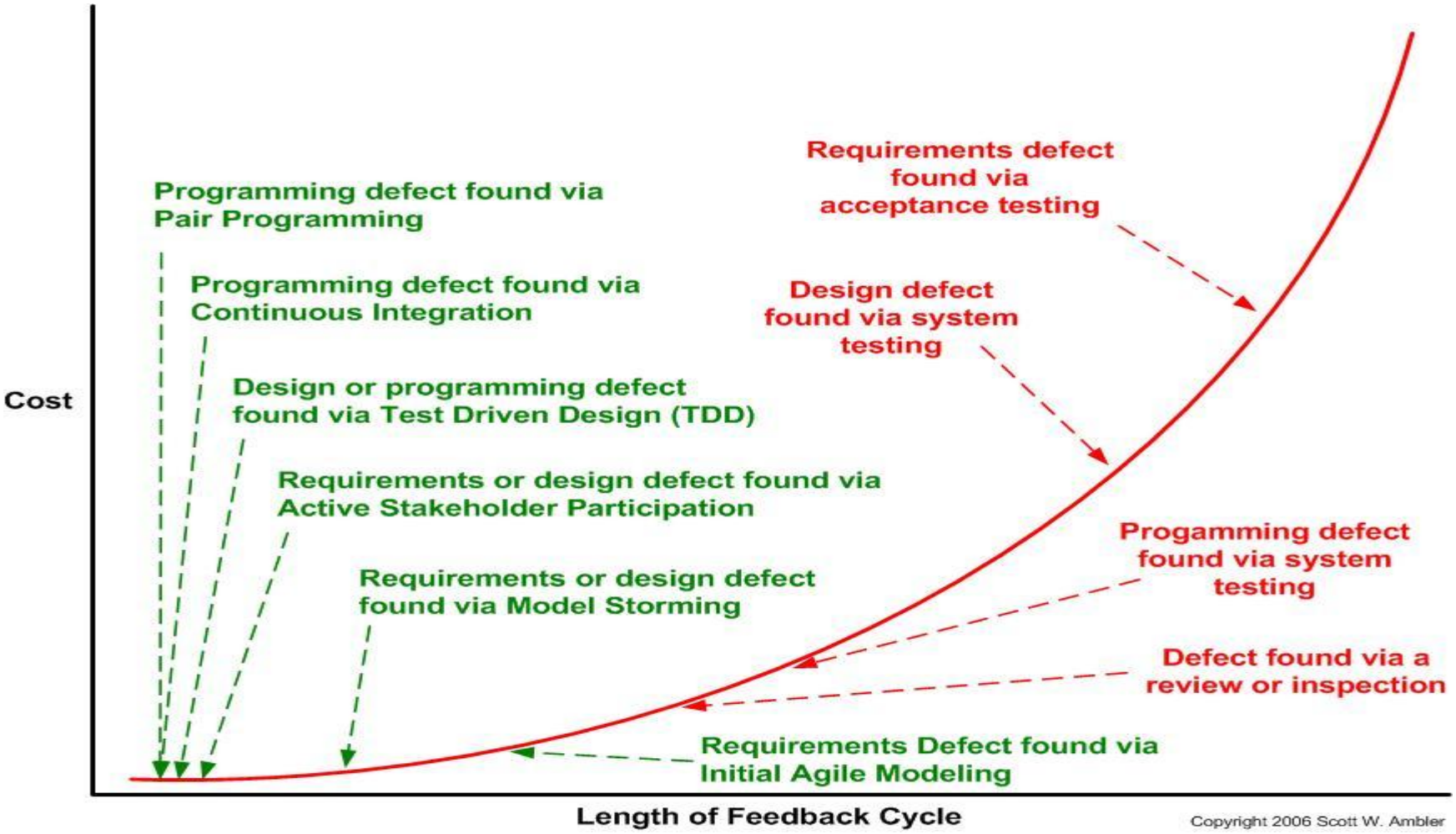
- Why?
  - Effective team communication is absolutely critical to assuring the correct product is produced
  - Frequent communication replaces heavy documentation typically used to communicate progress
  - Provides peer pressure to assure everyone on the team is carrying their weight
  - Assures risk and issues are raised early in the process when they are cost effectively fixed
- How?
  - Meeting rhythm that includes huddles, etc. during Sprints
  - Pairing of developers, testers, developers/testers
  - User acceptance testing

Agile Principles: Working Software is Measure of Progress

- Why?
  - True progress is exhibited when features are delivered
  - Working software can easily be demonstrated to customers
  - Working software can easily be released if needed

- How
  - End-to-end development of “steel threads” of functionality during each Sprint
  - Integrated testing during each Sprint
  - Burndown charts to track feature/function delivery as a metric

Why Projects Fail: Cost of change



Agile Principles: Sustainable Development

- Why?
  - Death marches result in burnout, programming mistakes, and ultimately staff turnover
- How?
  - Project scoping done by those who are asked to develop the functionality
  - Team works together to complete tasks that turn out to be more difficult than expected
  - Everything that must be done is scoped
  - Estimates for work include ramp-up time and expected administrative time

Agile Principles: Technical Excellence

- Why?
  - Reducing technical debt assures on-going success of the product
  - Focusing on excellence reduces the risks associated with turnover
- How?
  - Refactoring
  - Coding standards
  - Pairing
  - Team-based design

Agile Principles: Simplicity

- Why?
  - Trying to “boil the ocean” often results in analysis paralysis or over engineering of solutions
  - Simplicity assures we do not spend precious time and effort doing things that are never needed
- How
  - Team-based design
  - Refactoring

Agile Principles: Self Organized Teams

- Why?
  - Team members are often the best judge of who is capable of doing what
  - Makes teams accountable for their success
  - Increases peer pressure
  - Builds trust and community

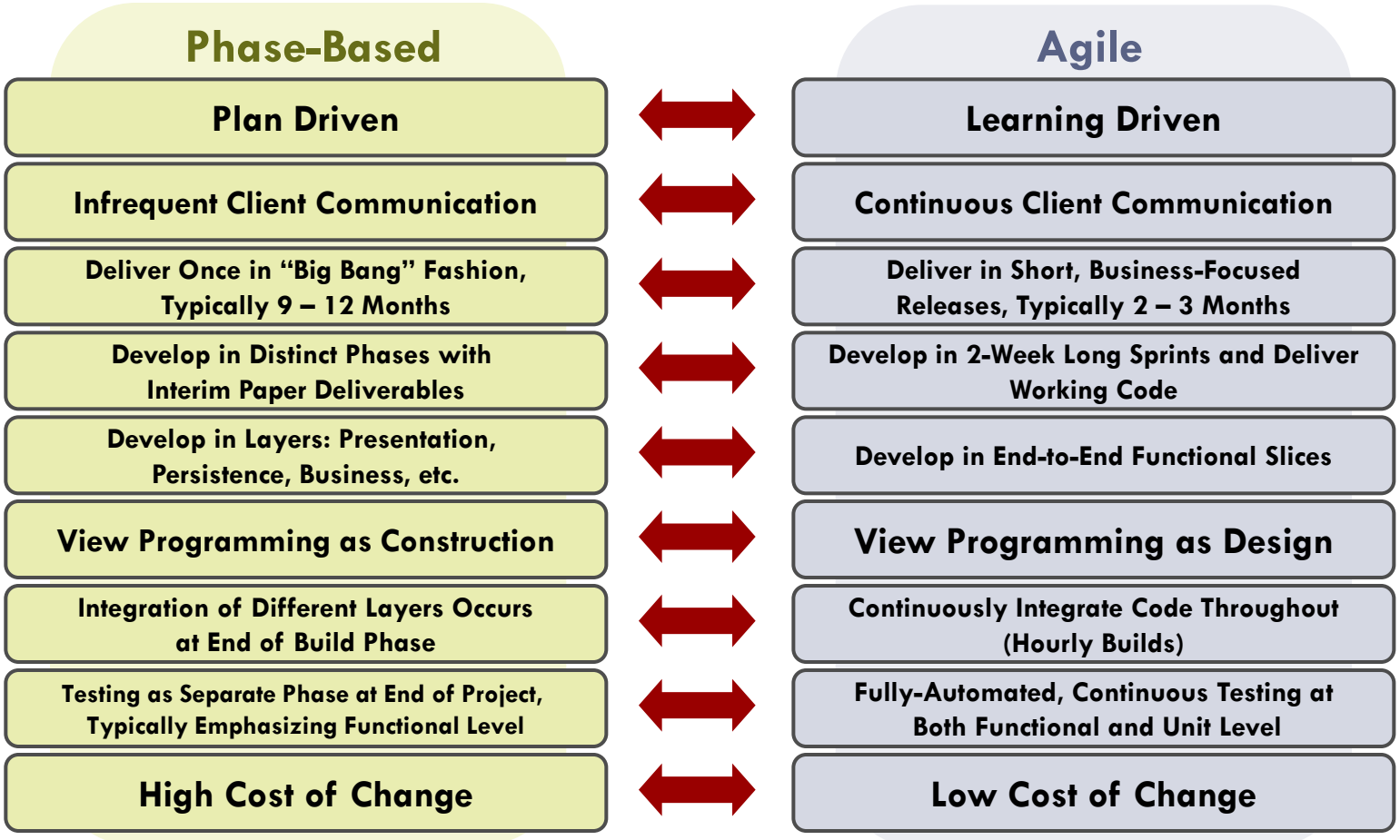
- How?
  - Teams scope the work for each Sprint
  - Teams decide who works on what during Sprint Planning sessions
  - Teams work toward a common goal / definition of done

Agile Principles: Continuous Improvement

- Why?
  - Adaptive software development methods must adapt!
  - Every team is different and will gel / improve across iterations
- How?
  - Pairing
  - Retrospectives
  - Iterative planning
  - User Acceptance Testing

What is Agile?

The agile approach is a different way of *thinking* about SW projects



Fundamentals of Agile Training

- Pragmatic approach to understanding agile management and implementation techniques
  - What is Agile?
  - Benefits of Agile
  - Myths about Agile
  - Agile Development Process
  - Agile Planning
  - Agile Development and Testing
- Completion of this course results in a “Certified ICAgile Professional” designation

Thank You!

Contact Information:

Jeffery Payne
jeff.payne@coveros.com
703.431.2920

