PMI®—Agile Certified Practitioner (PMI-ACP)®

Agile Methodologies–I

PMI® & ACP are the registered marks of Project Management Institute, Inc.

Copyright 2014, Simplilearn, All rights reserved.

Objectives

After completing this lesson, you will be able to:

- Explain some of the Agile methodologies and techniques

Copyright 2014, Simplilearn, All rights reserved.

Agile Methodologies, Frameworks, and Processes

A number of core Agile methodologies share the same philosophy expressed in the Agile Manifesto, however, there are different implementations with their own practices, processes, and techniques. The PMI-ACP certification references several core methodologies:

- Scrum
- Extreme Programming (XP)
- Lean Kanban
- Crystal
- Dynamic Systems Development Method (DSDM Atern)
- Feature Driven Development (FDD)
- Agile Project Management (APM)
- OpenUP

Copyright 2014, Simplilearn, All rights reserved.

Scrum

Scrum is one of the leading Agile techniques developed in the 1990s by Ken Schwaber and Jeff Sutherland.

Features that make scrum popular:

- Simplicity and proven results
- Enables other Agile engineering techniques
- Emphasizes small teams and team empowerment
- Welcomes changes to requirements
- Allows working from a single source of prioritized work items
- Daily status meetings
- Team commitment to a potentially shippable increment during a ‘Sprint’

Copyright 2014, Simplilearn, All rights reserved.

Scrum Roles

The Scrum roles are important features of Scrum and not using these specific terms is often referred to as 'Scrum-but.'

Scrum defines three roles:

Product Owner

Responsible for the project's success by defining the project vision, requirements, and priorities.

Scrum Master

Accountable to the team to remove impediments that will prevent them from achieving the goals of the Product Owner.

Development Team

Team comprises people with a mix of roles and self-organizes to determine how to best meet the goals of the Product Owner.

Copyright 2014, Simplilearn, All rights reserved.

Scrum Roles—Tips and Tricks

Following are some points to be kept in mind for each Scrum role:

Product Owner

- Resists the temptation to "manage" the team and to add more important work after the sprint is already in progress.
- Be willing to make hard choices during the sprint planning meeting.

Scrum Master

- Works to assist both the team and the Product Owner.
- Guides the Product Owner how to maximize Return On Investment (ROI).
- Improves the lives of the Development Team by facilitating creativity and empowerment.

The Team

- The team has the autonomy to choose how to best meet the goals, and is held responsible for them.

Copyright 2014, Simplilearn, All rights reserved.

Scrum Vocabulary

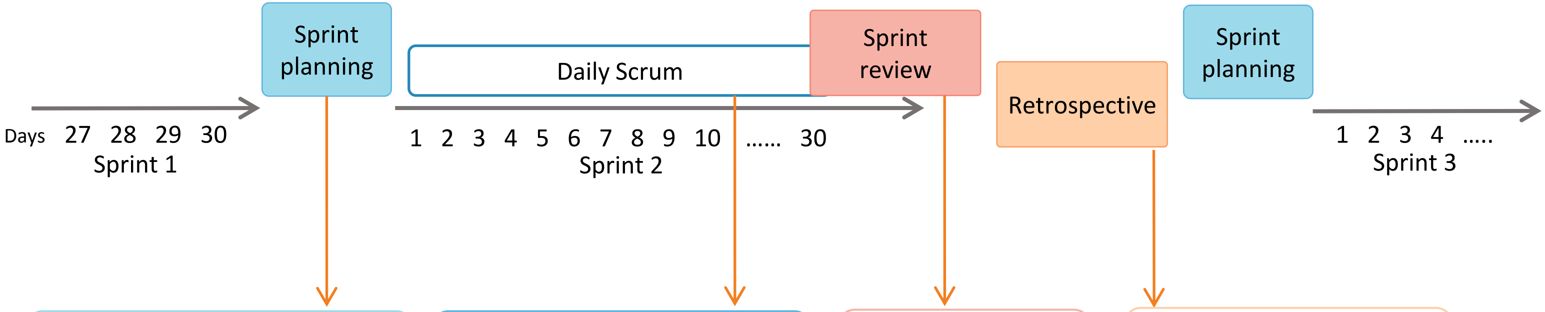
Following are some key Scrum terms:

- **Product Backlog**: All work to be performed in the foreseeable future, both well-defined and requiring further definition.
- **Sprint**: A period of 30 days or less within which a set of work will be performed to create a deliverable.
- **Sprint Backlog**: A well defined requirement that can be worked on with relatively little change over a period of 30 days or less and will result in a tangible, potentially shippable incremental deliverable.
- **Scrum**: A daily meeting at which progress and impediments to progress is reviewed.

Copyright 2014, Simplilearn, All rights reserved.

Scrum Meetings

Scrum defines four required Meetings.



Each sprint begins with an eight hour meeting to identify features and estimate effort. Four hours for selecting stories and four hours for estimating.

A 15 minute standing team meeting where three questions are answered:

- What was done yesterday?
- What will be done today?
- What blockers are there?

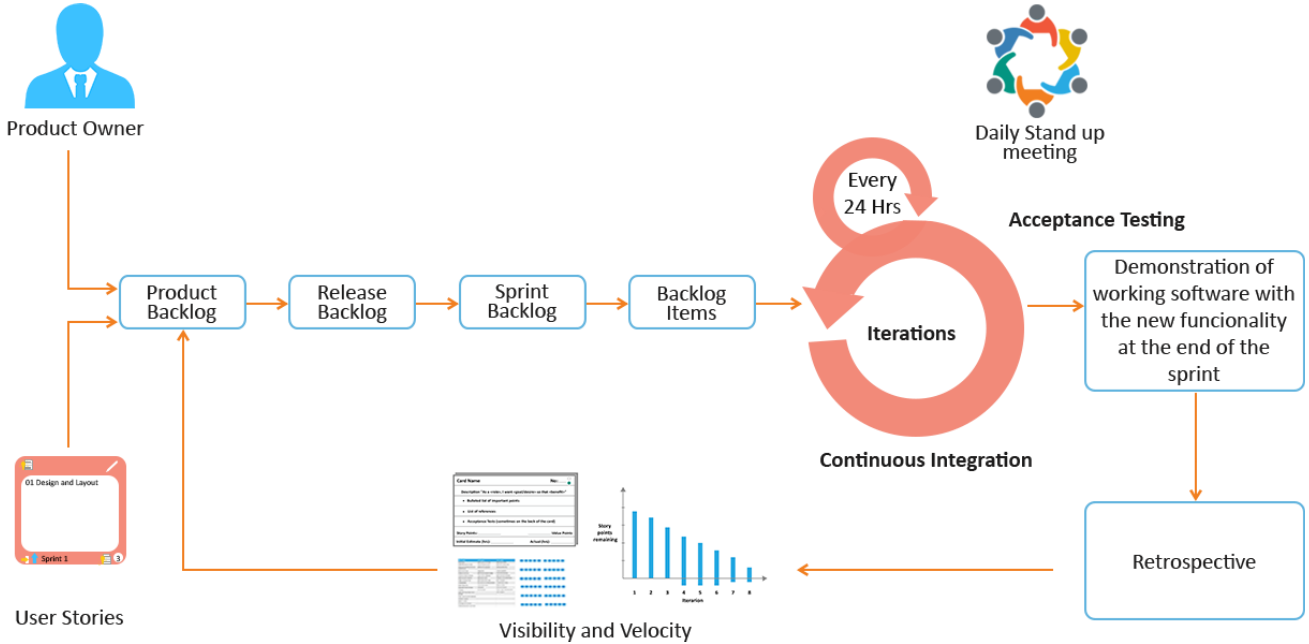
Team showcases what they have achieved in the form of potentially shippable deliverable.

At the end of each sprint the team reflects on what went well and what they would like to improve in the next sprint.

Copyright 2014, Simplilearn, All rights reserved.

Project Using Scrum—Example

Given below is a graphical representation of an ideal Agile project that uses Scrum.



Copyright 2014, Simplilearn, All rights reserved.

Extreme Programming

Extreme Programming (XP) was developed by Kent Beck and Ward Cunningham in the 1990s to:

- respond to the high cost of changing requirements; and
- institute strong engineering practices to improve software quality.

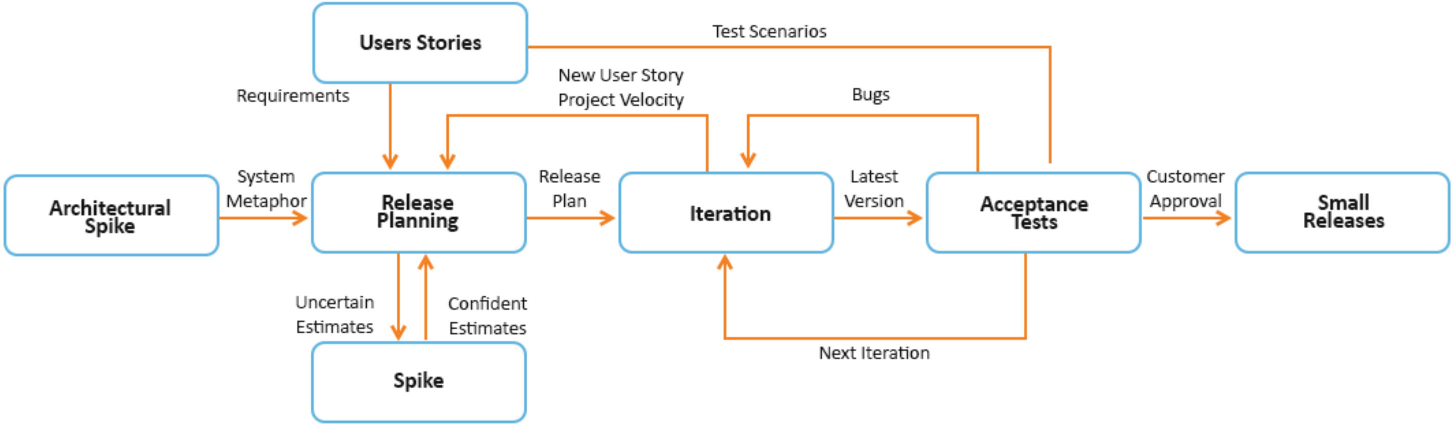
XP introduced many revolutionary concepts to software development that have now become standard practices:

- Test Driven Development;
- Continuous Integration;
- Iterations; and
- User Stories.

Copyright 2014, Simplilearn, All rights reserved.

Extreme Programming Process Diagram

The process of XP goes as follows:



Copyright 2014, Simplilearn, All rights reserved.

Five Core Principles of XP

Extreme Programing builds upon five core principles.

Communication

- Frequent collaboration between users and programmers.
- Use simple designs common metaphors and application of patterns.

Simplicity

- Focus on the simplest solution.
- Don’t build more functionality than required.
- Refactor complexity.

Feedback

- Unit Tests for feedback from the system.
- Acceptance Tests for feedback from the customer.
- The Planning Game for feedback from the team.

Courage

- Refactor code to make future changes easier.
- Throw code away that is obsolete.

Respect

- Respect for others;
- Self respect;
- Adopting the other four values; and
- Respect gained from others in the team.

Copyright 2014, Simplilearn, All rights reserved.

XP Practices

The XP Practices introduced a range of techniques that are now accepted as standard practices.

Fine-scale feedback	Pair programming; Planning game; Test-driven development; Whole team
Continuous process	Continuous integration; Refactoring or design improvement; Small releases
Shared understanding	Coding standards; Collective code ownership; Simple design; System metaphor
Programmer welfare	Sustainable pace

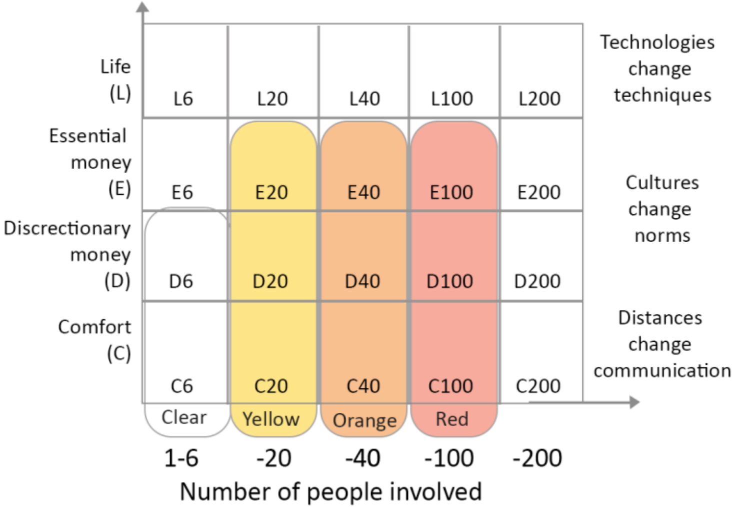
Copyright 2014, Simplilearn, All rights reserved.

Crystal Methodologies

Different levels of governance and visibility were required based on the ‘type’ of project.

Crystal Clear: For small teams working on projects with low risk to life and using discretionary monies. In the graph, the projects that fall on the far left belong to the category of ‘Crystal Clear.’

Crystal Red: For a larger project dealing with life and death implications which would have more governance, documentation, and control gates. The projects on the far right belong to the category of ‘crystal red’, in the graph.



Life (L)	L6	L20	L40	L100	L200	Technologies change techniques
Essential money (E)	E6	E20	E40	E100	E200	Cultures change norms
Discrectionary money (D)	D6	D20	D40	D100	D200	
Comfort (C)	C6	C20	C40	C100	C200	Distances change communication
	Clear	Yellow	Orange	Red		
	1-6	-20	-40	-100	-200	

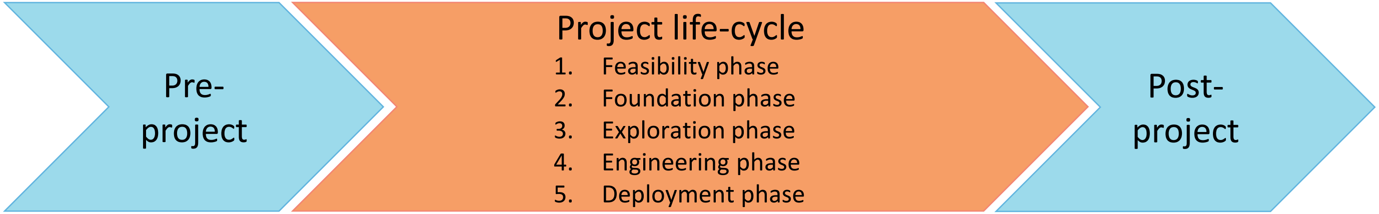
Number of people involved

Copyright 2014, Simplilearn, All rights reserved.

Dynamic Systems Development Method

DSDM was developed in the 1990s to provide more discipline to Rapid Application Development (RAD). The latest version is called **Atern**. DSDM uses a prioritization technique called **MoSCoW** (Must, Should, Could and Won't) to determine which requirements should be included in a release or iteration.

Following graphic represents phases of DSDM:



Pre-project

Project life-cycle

1. Feasibility phase
2. Foundation phase
3. Exploration phase
4. Engineering phase
5. Deployment phase

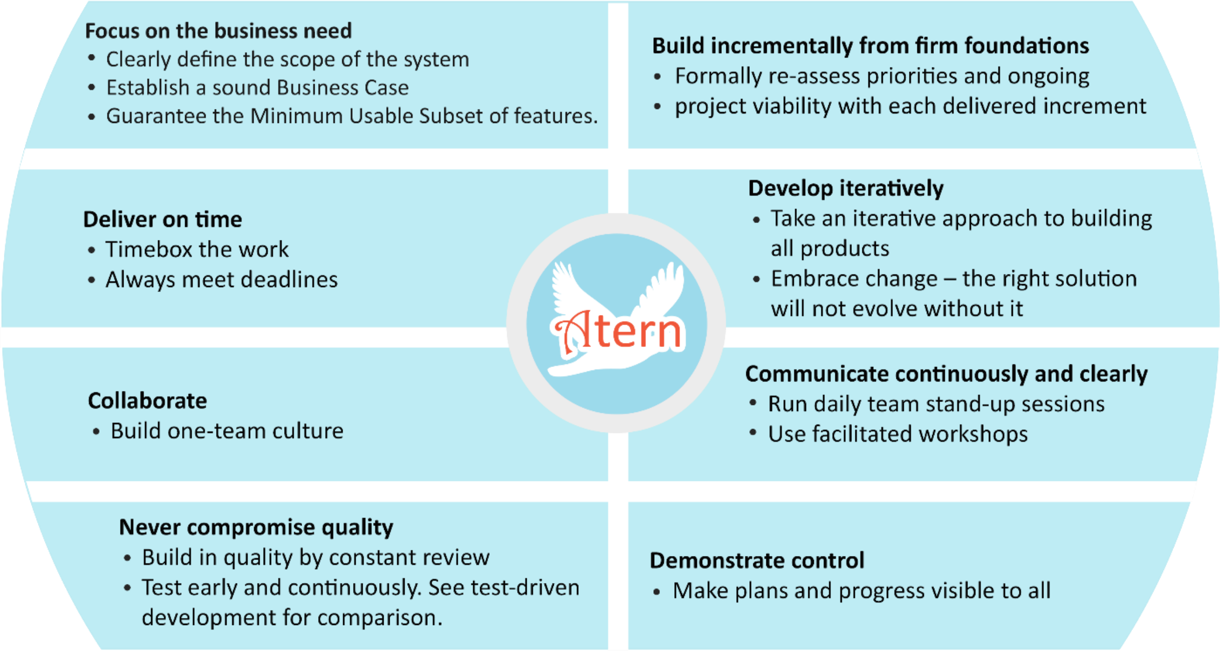
Post-project

Copyright 2014, Simplilearn, All rights reserved.

Principles of DSDM Atern

simplilearn
your pace, your place

Following are the eight DSDM Atern principles:



Focus on the business need
- Clearly define the scope of the system
- Establish a sound Business Case
- Guarantee the Minimum Usable Subset of features.

Build incrementally from firm foundations
- Formally re-assess priorities and ongoing
- project viability with each delivered increment

Deliver on time
- Timebox the work
- Always meet deadlines

Develop iteratively
- Take an iterative approach to building all products
- Embrace change – the right solution will not evolve without it

Collaborate
- Build one-team culture

Communicate continuously and clearly
- Run daily team stand-up sessions
- Use facilitated workshops

Never compromise quality
- Build in quality by constant review
- Test early and continuously. See test-driven development for comparison.

Demonstrate control
- Make plans and progress visible to all

Copyright 2014, Simplilearn, All rights reserved.

Feature Driven Development

Feature Driven Development (FDD) is an iterative and incremental approach to software development that was developed in the late 1990s by Jeff DeLuca and Peter Coad.

- Features are small pieces of client-valued functions expressed in the form:

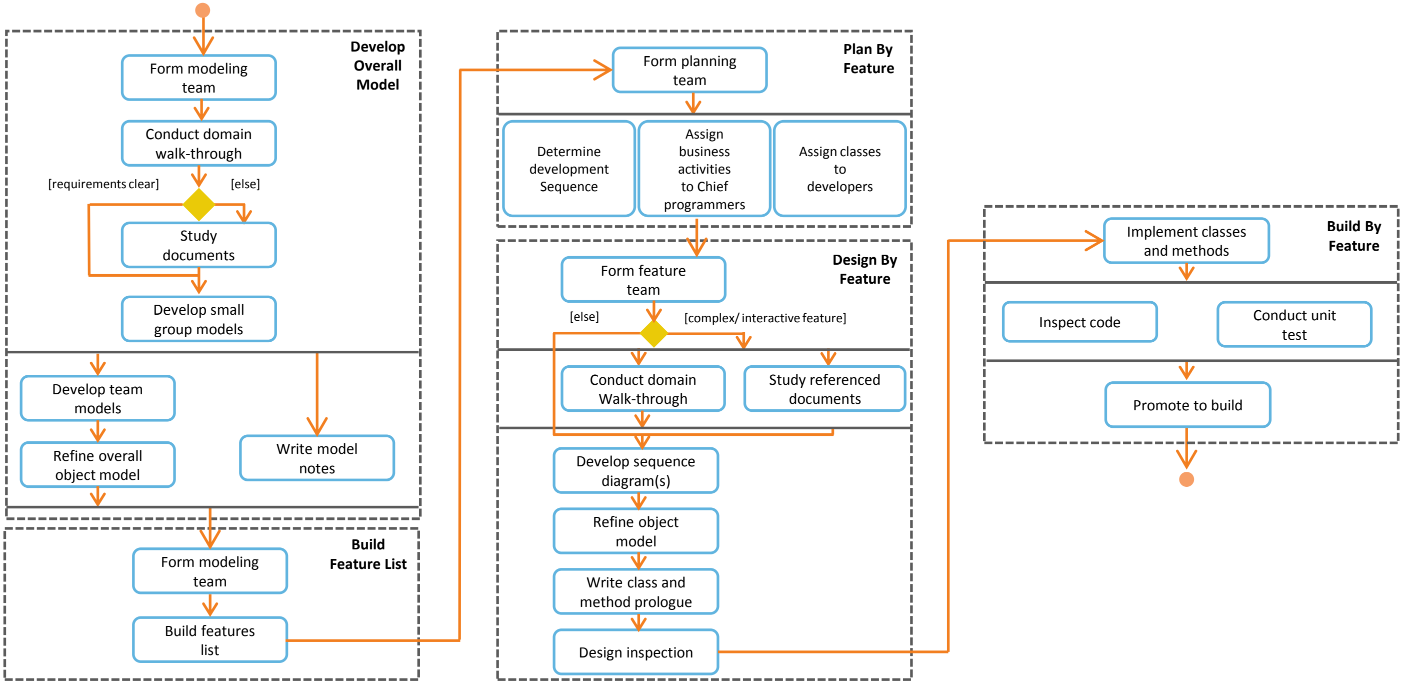
  "<action> <result> <object>"

- Through decomposition domain models are broken down into subject areas which are then expressed as business activities.
- Each step in a business activity is a feature.
- Features should not take more than two weeks to complete, or they should be broken down into smaller pieces.

Copyright 2014, Simplilearn, All rights reserved.

Feature Driven Development Cycle

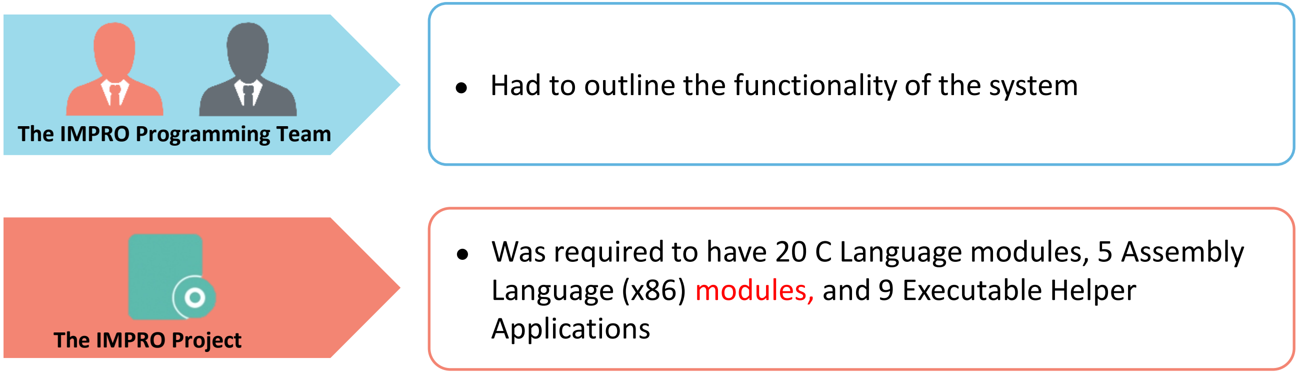
The steps of feature driven development cycle are as follows:


Develop Overall Model
Form modeling team
Conduct domain walk-through
[requirements clear]
[else]
Study documents
Develop small group models
Develop team models
Refine overall object model
Write model notes
Build Feature List
Form modeling team
Build features list
Plan By Feature
Form planning team
Determine development Sequence
Assign business activities to Chief programmers
Assign classes to developers
Design By Feature
Form feature team
[else]
[complex/ interactive feature]
Conduct domain Walk-through
Study referenced documents
Develop sequence diagram(s)
Refine object model
Write class and method prologue
Design inspection
Build By Feature
Implement classes and methods
Inspect code
Conduct unit test
Promote to build

Copyright 2014, Simplilearn, All rights reserved.

Success of XP Methodology—Real Life Example

The graphic below shows the problems faced during the development of IMPRO, an image processing environment.



The IMPRO Programming Team

- Had to outline the functionality of the system

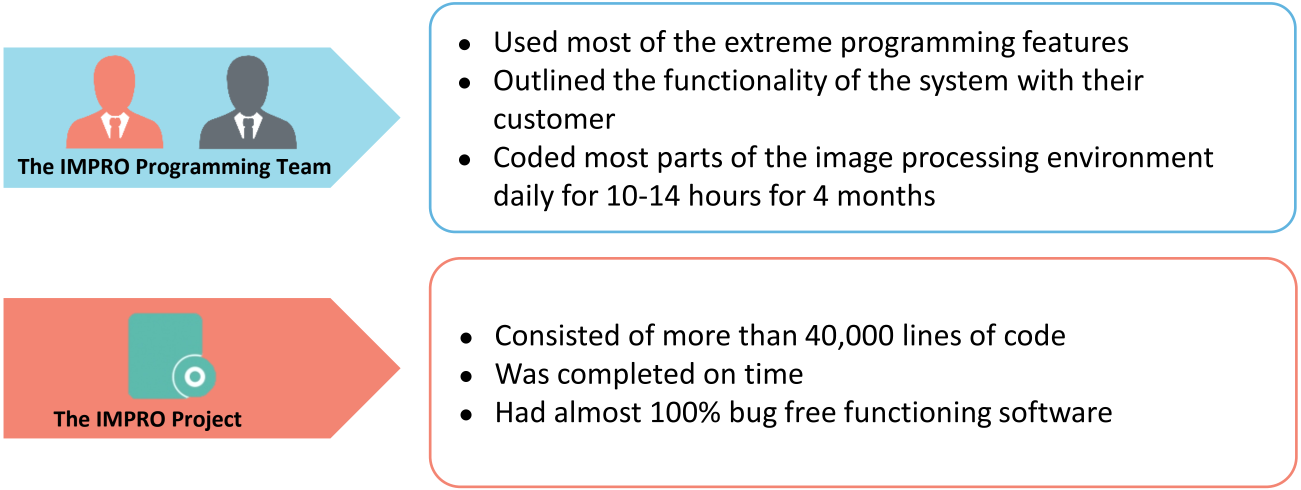
The IMPRO Project

- Was required to have 20 C Language modules, 5 Assembly Language (x86) modules, and 9 Executable Helper Applications

Copyright 2014, Simplilearn, All rights reserved.

Success of XP Methodology Example—Outcome

The graphic below shows the successful outcome of the IMPRO Project.



The IMPRO Programming Team

- Used most of the extreme programming features
- Outlined the functionality of the system with their customer
- Coded most parts of the image processing environment daily for 10-14 hours for 4 months

The IMPRO Project

- Consisted of more than 40,000 lines of code
- Was completed on time
- Had almost 100% bug free functioning software

Copyright 2014, Simplilearn, All rights reserved.

Quiz

Copyright 2014, Simplilearn, All rights reserved.

QUIZ 1

How many roles are there in Scrum?

a. None

b. 3

c. 12

d. 7 ± 2

Copyright 2014, Simplilearn, All rights reserved.

QUIZ

1

How many roles are there in Scrum?

a. None

b. 3

c. 12

d. 7 ± 2

Answer: b.

Explanation: Scrum identifies three roles: Product Owner, Scrum Master, and the Team.

Copyright 2014, Simplilearn, All rights reserved.

QUIZ 2

Which of the following values is XP based upon?

a. Reduce the cost of requirements changes

b. Visualize, Limit WIP, Manage flow, Make management policies, Improve collaboratively

c. Trust, Commitment, Value, Delivery

d. Simplicity, Communication, Feedback, Courage, and Respect

Copyright 2014, Simplilearn, All rights reserved.

QUIZ
2

Which of the following values is XP based upon?

a. Reduce the cost of requirements changes

b. Visualize, Limit WIP, Manage flow, Make management policies, Improve collaboratively

c. Trust, Commitment, Value, Delivery

d. Simplicity, Communication, Feedback, Courage, and Respect

Answer: d.

Explanation: XP is based on 5 values Simplicity, Communication, Feedback, Courage, and Respect. It was developed to respond to the high cost of requirements change. Answer b reflects the core practices of Lean Kanban.

✓

Copyright 2014, Simplilearn, All rights reserved.

QUIZ 3

Which of the following is not one of the XP Practices ?

a. Courage

b. Fine-scale feedback

c. Shared understanding

d. Programmer welfare

Copyright 2014, Simplilearn, All rights reserved.

QUIZ

3

Which of the following is not one of the XP Practices ?

a. Courage

b. Fine-scale feedback

c. Shared understanding

d. Programmer welfare

Answer: a.

Explanation: Courage is one of the XP Principles. The other XP Practice is Continuous Improvement.

✓

Copyright 2014, Simplilearn, All rights reserved.

Summary

Here is a quick recap of what was covered in this lesson:

- A number of core agile methodologies share the same philosophy expressed in the Agile Manifesto, however, there are different implementations with their own practices, processes, and techniques.
- Scrum and Extreme Programing (XP) are some of the Agile practices that are widely popular.

Copyright 2014, Simplilearn, All rights reserved.

THANK YOU

PMI® & ACP are the registered marks of Project Management Institute, Inc.

Copyright 2014, Simplilearn, All rights reserved.