Comparison of Agile Methodologies

AGILE Manifesto

Values

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

That is, while there is value in the item on the right, we value the items on the left

The Agile Principles (1)

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage
3. Deliver working software frequently, from a couple of weeks to a couple of month, with a preference to the shorter time scale
4. Business people and developers must work together daily throughout the project

The Agile Principles (2)

1. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done
2. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation
3. Working software is the primary measure of progress
4. Agile processes promote sustainable development
5. The sponsors, developers and users should be able to maintain a constant pace indefinitely

The Agile Principles (3)

1. Continuous attention to technical excellence and good design enhances agility
2. Simplicity – the art of maximizing the amount of work not done – is essential
3. The best architectures, requirements and designs emerge from self-organizing teams
4. At regular intervals, the team reflects on how to become more effective, the tunes and adjust its behavior accordingly

The planning spectrum

(Boehm 2002)
Agile Methods Vs Plan-driven Methods (Bohem 2002)

<table>
<thead>
<tr>
<th>Area</th>
<th>Agile Methods</th>
<th>Plan driven methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developers</td>
<td>Agile, knowledgeable,</td>
<td>Plan-oriented,</td>
</tr>
<tr>
<td></td>
<td>collaborative</td>
<td>adequate skills,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>access to external knowledge</td>
</tr>
<tr>
<td>Customers</td>
<td>Dedicated, knowledgeable,</td>
<td>Access to</td>
</tr>
<tr>
<td></td>
<td>collaborative,</td>
<td>knowledgeable,</td>
</tr>
<tr>
<td></td>
<td>representative and</td>
<td>collaborative,</td>
</tr>
<tr>
<td></td>
<td>empowered</td>
<td>representative and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>empowered</td>
</tr>
<tr>
<td>Size</td>
<td>Smaller teams and products</td>
<td>Larger teams and products</td>
</tr>
</tbody>
</table>

Agile Methods Process Comparison

- eXtreme Programming
  1. Exploration
  2. Planning
  3. Iterations to first release
  4. Production
  5. Maintenance

Agile Methods Process Comparison

- Scrum
  1. Pre-game
  2. Development
  3. Release

- Crystal Clear
  No defined Process

Agile Methods Process Comparison

- Feature Driven Development
  1: Develop an Overall Model
  2: Build a Features List
  3: Plan by Feature
  4: Design by Feature
  5: Build by Feature

Agile Methods Process Comparison

- Rational Unified Process
  1. Inception
  2. Elaboration
  3. Construction
  4. Transition

- Adaptive Software Development
  1. Speculate
  2. Collaborate
  3. Learn
Agile Methods Process Comparison

• Dynamic System Development Method
  1. Feasibility
  2. Business study
  3. Functional Model Iteration
  4. Design and build iteration
  5. Implementation

Agile Methods Roles & Responsibilities Comparison

• eXtreme Programming
  1. Coach
  2. Programmer
  3. Tester
  4. Customer
  5. Consultant

• Scrum
  1. Product Owner
  2. Scrum Team
  3. Scrum Master
  4. Chicken

Agile Methods Roles & Responsibilities Comparison

• Crystal Clear
  1. Sponsor
  2. Designer-Programmers
  3. Coordinator
  4. Business Expert
  5. Ambassador User
  6. Lead Designer
  7. Tester
  8. Writer

Agile Methods Roles & Responsibilities Comparison

• Feature Driven Development
  1. Project manager
  2. Chief architect
  3. Domain experts
  4. Development Manager
  5. Chief Programmers
  6. Class owners
  7. Environment Manager
  8. Tester

Agile Methods Roles & Responsibilities Comparison

• Rational Unified Process
  30 roles called workers

• Adaptive Software Development
  1. Project Manager
  2. Developer representative
  3. Facilitator
  4. Customer
  5. Scribe

• Dynamic System Development Method
  1. Project Manager
  2. Team Leader
  3. Developer
  4. Tester
  5. Scribe
  6. Advisor User
  7. Technical Coordinator
  8. Executive Sponsor
  9. Visionary
  10. Ambassador User
Agile Methods Practices
Comparison

- **eXtreme Programming**
  1. Planning game
  2. Small, short releases
  3. System metaphor
  4. Simple design
  5. Testing
  6. Frequent Refactoring
  7. Pair programming
  8. Collective ownership
  9. Continuous integration
  10. 40-hour week (sustainable pace)
  11. On-site customer
  12. Coding standards
  13. Open workspace
  14. Just in time

Agile Methods Practices
Comparison

- **Scrum**
  1. Self-directed and self-organizing team
  2. No external addition of work to an iteration, once chosen
  3. Daily standup meeting with special questions
  4. Usually 30-calendary day iterations
  5. Demo to external stakeholders at end of each iteration
  6. Each iteration, client-driven adaptive planning

Agile Methods Practices
Comparison

- **Crystal Clear**
  1. Frequent delivery
  2. Close communication
  3. Reflective improvement
  4. Personal safety
  5. Focus
  6. Easy access to experts users
  7. Technical environment

Agile Methods Practices
Comparison

- **Feature Driven Development**
  1. Domain Object Modeling
  2. Developing by Feature
  3. Individual Class (Code) Ownership
  4. Feature Teams
  5. Inspections
  6. Regular Builds
  7. Configuration Management
  8. Reporting/Visibility of Results

Agile Methods Practices
Comparison

- **Rational Unified Process**
  1. Develop Software iteratively
  2. Manage requirements
  3. Use component-based architectures
  4. Visually model software
  5. Continuously verify software quality
  6. Control changes to software

Agile Methods Practices
Comparison

- **Dynamic System Development Method**
  1. Active user involvement is imperative
  2. Teams must be empowered to make decisions
  3. The focus is on frequent delivery of products
  4. Fitness for business purpose
  5. An iterative and incremental approach
  6. All changes during development are reversible
  7. Requirements are base lined at a high level
  8. Testing is integrated throughout the lifecycle
  9. A collaborative and co-operative approach
Agile Methods Practices
Comparison

- Adaptive Software Development
  1. Iterative development
  2. Feature-based (component based) planning
  3. Customer focus group reviews

Agile Methods Scope
Comparison

- eXtreme Programming
  Small and medium size team (between 3 and 20 members)
- Scrum
  Small teams (less than 10 members)
- Crystal Clear
  Any size team but not cover life critical projects
- Adaptive Software Development
  Not built-in limitations

Agile Methods Scope
Comparison

- Feature Driven Development
  Effective on large projects with complex business logic
  Suitable for the development of critical systems
  Upgrading existing code, second version
- Rational Unified Process
  Not built-in limitations
- Dynamic System Development Method
  Has been applied in small and large projects
  More easily applied to business systems than to engineering or scientific application