Introduction to Agile Software Development

Word Association

Write down the first word or phrase that pops in your head when you hear:

- Extreme Programming (XP)
- Team (or Personal) Software Process (TSP/PSP)
- Plan-driven software development
- Agile software development
Process Methodology Myths

✔ Agile Methods
  • *cowboys and hackers*
  • *undisciplined*
  • *low quality*

✔ Plan Driven Methods
  • *process worship*
  • *document laden*
  • *excessive discipline*

✔ It’s not that black and white. The process spectrum spans a range of gray.

Important Concepts

Plan-Driven
  • Process Improvement
  • Process Capability
  • Organizational Maturity
  • Process Group
  • Risk Management
  • Verification (building the product right)
  • Validation (building the right product)
  • System Architecture

Agile
  • Embrace Change
  • Frequent Delivery
  • Simple Design
  • Refactoring
  • Pair Programming
  • Retrospective
  • Tacit Knowledge
  • Test-Driven Development (TDD)
Plan-Driven Approach

Characteristics

• Systematic engineering approach
• Completeness of documentation
• Thorough verification - traceability
• Traditionally waterfall, but more incremental and evolutionary processes are the norm.

Examples

• Cleanroom (mathematically driven)
• PSP/TSP (Humphrey, SEI)
• SW-CMM (process improvement framework)

Agile Approach

Characteristics

• Short, iterative cycles
• Incremental delivery
• Evolutionary work artifacts (test, design, code)
• Active customer involvement
• Dynamic application domains (requirements)

Examples

• eXtreme Programming (XP) – (Beck)
• Crystal family (Cockburn)
• Scrum (Schwaber)
• Feature-Driven Development (Coad)
What Is Agile Software Development?

In the late 1990's several methodologies began to get increasing public attention. All emphasized:

- close collaboration between the programmer team and business experts
- face-to-face communication (as more efficient than written documentation)
- frequent delivery of new deployable business value
- tight, self-organizing teams
- ways to craft the code and the team such that the inevitable requirements churn was not a crisis.

2001: Workshop in Snowbird, Utah, Practitioners of these methodologies met to figure out just what it was they had in common. They picked the word "agile" for an umbrella term and crafted the

- Manifesto for Agile Software Development.
Manifesto for Agile Software Development

Statement of shared development values:

✓ **Individuals and Interactions** – over process 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 items on the right, we value the items on the left more.”

What traditional developers heard

Statement of shared development values:

✓ **Individuals and Interactions** = NO process
✓ **Working software** = NO documentation
✓ **Customer collaboration** = NO contracts
✓ **Responding to change** = NOT following a plan

© Scott Adams, Inc./Dist. by UFS, Inc.
Principles behind the Agile Manifesto

We follow these principles:

✓ Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
✓ Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.
✓ Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
✓ Business people and developers must work together daily throughout the project.
✓ Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
✓ The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Principles behind the Agile Manifesto

We follow these principles (continued):

✓ Working software is the primary measure of progress.
✓ Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
✓ Continuous attention to technical excellence and good design enhances agility.
✓ Simplicity—the art of maximizing the amount of work not done—is essential.
✓ The best architectures, requirements, and designs emerge from self-organizing teams.
✓ At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
**Traditional Approach**

- Project follows a waterfall process (plan driven)
- Teams produce artifacts at each phase of the life-cycle in a sequential manner.
- Significant upfront design effort
- Implementation delayed until later stages of the project
- Testing deferred until coding complete
- Teams make final presentation to the customer
- Teams participate in postmortem session
Traditional Challenges

- Lightweight application/heavyweight process
- Document intensive (perceived)
- Less flexible design
- Big bang approach to coding/integration
- Testing short-shifted
- One-shot presentation opportunity
- Lack of opportunity for process improvement
- Prone to “Analysis-Paralaysis”
  - “Ready, Aim, Aim, Aim, …”

Four Project Variables

- **Time** – duration of the project
- **Quality** – the requirements for ‘correctness’
- **Resources** – personnel, equipment, etc.
- **Scope** – what is to be done; the features to be implemented

Pick three, any three . . .
**Planning**

“The plan is nothing; the planning is everything”

✓ Dwight Eisenhower

✓ Allied supreme commander during World War II

✓ 34th President of United States (1953-61)

---

**Where are the risks?**

*Figure 2. Risk exposure (RE) profile. This planning model for a sample e-services company shows the probability of loss P(L) and size of loss S(L) for several significant factors.*

“Getting Ready for Agile Methods With Care”, Barry Boehm – IEEE Computer, 2002
Agile RE Profile

"Getting Ready for Agile Methods With Care", Barry Boehm – IEEE Computer, 2002

Plan-Driven RE Profile

"Getting Ready for Agile Methods With Care", Barry Boehm – IEEE Computer, 2002
Agile Characteristics

✓ Incremental development – several releases
✓ Planning based on user stories
✓ Each iteration touches all life-cycle activities
✓ Testing – unit testing for deliverables
✓ Testing – acceptance tests for each release
✓ Flexible Design – evolution vs. big upfront effort
✓ Reflection after each release cycle
✓ Several technical and customer focused presentation opportunities

Key Agile Contributions

✓ Team Skills
  • Collaborative Development
  • Reflections (process improvement)
✓ User Stories
  • Requirements elicitation
  • Planning – scope & composition
✓ Evolutionary Design
  • Opportunity to make mistakes
✓ Continuous Integration
  • Code (small booms vs big bang)
✓ Testing
  • Dispels notion of testing as an end of cycle activity
✓ Communication
  • Interacting with customer / team members
Agile Software Development

Agile Themes:
- Lightweight disciplined processes
- Feature / Customer Focused
- Small teams
- Short delivery cycles

Popular Agile Methodologies:
- XP (eXtreme Programming)
- Crystal Family
- Adaptive Software Process
- Scrum

Characteristics of Agile Methodologies

- Deliver working software frequently
- Incremental development cycles – release plan based on user stories.
- Evolutionary approach to design – design what you need for this release cycle
- Test – Test – Test (Unit & Acceptance)
- Customer participation
- Lightweight documentation
- Reflect at regular intervals – tune and adjust
Agile Benefits

- User stories drive planning and requirements in a manageable work units
  - Customer perspective
  - Risk management
- Frequent delivery of working software
  - Process reflection opportunities
  - Implementation refactoring
  - Positive feedback to team

Testing Focus
- Test early and often
- Change in attitude towards testing

Transitioning to Agile

- Agile is not a “Silver Bullet” that will cure all your development woes. It will however brightly illuminate your opportunities.
- Where are the opportunities for improvement in our current process? How does Agile address those issues?
- Trust, Transparency, Patience
- Individual Opportunity - “Generalizing Specialists”
- The adoption of Agile is neither completely top-down or bottom-up. It must be a balance of both with a strong level of trust and commitment between all levels of the organization
Common Issues

Typical issues/obstacles that arise include:

- Lack of business ownership and the inability to make decisions
- Limited business buy-in into the concept of Agile
- Team communication, individual skills, and team fit
- Lack of trust in the team by the business
- Focus only on Agile development practices

Agile permeates all levels of the organization
Methodology Distribution

<table>
<thead>
<tr>
<th>Methodology</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scrum</td>
<td>40%</td>
</tr>
<tr>
<td>Kanban</td>
<td>27%</td>
</tr>
<tr>
<td>Extreme Programming (XP)</td>
<td>10%</td>
</tr>
<tr>
<td>Test-Driven Development (TDD)</td>
<td>10%</td>
</tr>
<tr>
<td>Lean</td>
<td>3%</td>
</tr>
<tr>
<td>Business-Oriented Development</td>
<td>3%</td>
</tr>
<tr>
<td>Agile Methodology</td>
<td>3%</td>
</tr>
<tr>
<td>Six Sigma</td>
<td>3%</td>
</tr>
<tr>
<td>Capability Maturity Model/Integration (CMMI)</td>
<td>3%</td>
</tr>
<tr>
<td>Rational Unified Process (RUP)</td>
<td>3%</td>
</tr>
<tr>
<td>ISSM</td>
<td>3%</td>
</tr>
<tr>
<td>Agile</td>
<td>2%</td>
</tr>
<tr>
<td>Other</td>
<td>2%</td>
</tr>
<tr>
<td>Business-Oriented Development</td>
<td>2%</td>
</tr>
<tr>
<td>Unified Process (UP)</td>
<td>2%</td>
</tr>
<tr>
<td>Agile Data Method</td>
<td>2%</td>
</tr>
<tr>
<td>Microsoft Solutions Framework (MSF)</td>
<td>2%</td>
</tr>
<tr>
<td>Other</td>
<td>2%</td>
</tr>
<tr>
<td>Dynamic Systems Development Method (DSM)</td>
<td>2%</td>
</tr>
<tr>
<td>Cyneal</td>
<td>2%</td>
</tr>
</tbody>
</table>

Note: This methodology distribution is based on a survey of 141 technology industry professionals. All totals may not add up due to rounding.

Source: December 2008 Global Agile Company Online Survey

Resources

- Agile Software Development Portal: [agile.csc.ncsu.edu/](http://agile.csc.ncsu.edu/)
- Agile Alliance – [www.agilealliance.com](http://www.agilealliance.com)
- [www.extremeprogramming.org/](http://www.extremeprogramming.org/)
- Laurie Williams – North Carolina State: [collaboration.csc.ncsu.edu/laurie/index.html](http://collaboration.csc.ncsu.edu/laurie/index.html)