Creating a High Maturity Agile Implementation
Copyright Notice 2015.

- Unless otherwise noted, these materials and the presentation of them are based on the methodology developed by Cape Project Management, Inc. and are copyrighted materials of Cape Project Management, Inc., which are used with its permission. Cape Project Management, Inc. reserves all its rights.
- All other brand or product names used in this guide are trade names or registered trademarks of their respective owners.
- Portions of this slide deck are attributed to Ken Schwaber and Jeff Sutherland: The Scrum Guide. 2013, http://www.Scrumguides.org and to Mountain Goat Software, LLC under Creative Commons. http://creativecommons.org/licenses/by-sa/4.0/.
- The Agile maturity matrix is based upon the Maturity Index for Cultural Agility developed by Vodafone UK and Hewlett Packard as presented in this paper to the UK’s National Audit office.
- The online Agile self-assessment was adapted from an original source developed by Henrik Kniberg and is licensed under a Creative Commons: http://creativecommons.org/licenses/by-nc-nd/3.0/.
About Your Instructor: Dan Tousignant

Project Management Consultant and Trainer
- Project Management Professional (PMP)
- Agile Certified Practitioner (PMI-ACP)
- Certified Scrum Professional (CSP)

20 years of experience managing projects with budgets of up to $60 million

Follow me on Twitter @scrumdan

Webinar Learning Outcomes
- Recognize the organizational environment and its suitability for Agile
- Characteristics of an Highly effective Agile Implementation
- Obstacles to an effective Implementation
Agile Practices Overview

- Key benefits of Agile approaches include short delivery cycles and business driven development
- Agile is an umbrella term that encompasses many processes and practices, such as Scrum, Extreme Programming, Lean Software Development, Kanban, and more...
Creating a High Maturity Agile Implementation

Agile Adoption

<table>
<thead>
<tr>
<th>Method</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scrum</td>
<td>56%</td>
</tr>
<tr>
<td>Scrum/XP Hybrid</td>
<td>10%</td>
</tr>
<tr>
<td>Custom Hybrid</td>
<td>8%</td>
</tr>
<tr>
<td>Scrumban</td>
<td>6%</td>
</tr>
<tr>
<td>Kanban</td>
<td>5%</td>
</tr>
<tr>
<td>Iterative</td>
<td>4%</td>
</tr>
<tr>
<td>I don't know</td>
<td>3%</td>
</tr>
<tr>
<td>Lean</td>
<td>2%</td>
</tr>
<tr>
<td>Other</td>
<td>2%</td>
</tr>
<tr>
<td>Agile Modeling</td>
<td>1%</td>
</tr>
<tr>
<td>FDD</td>
<td>1%</td>
</tr>
<tr>
<td>AgileUP</td>
<td>1%</td>
</tr>
<tr>
<td>DSDM</td>
<td>1%</td>
</tr>
<tr>
<td>XP</td>
<td>1%</td>
</tr>
</tbody>
</table>

94% of all organizations surveyed practice agile.
45% of them said the majority of their teams are agile.

Source: Version One, 9th Annual State of Agile Survey 2015

Agile Project Success Rates

<table>
<thead>
<tr>
<th>Method</th>
<th>Successful</th>
<th>Challenged</th>
<th>Failed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lean/Kanban/Scrumban</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iterative/RUP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scrum/XP/DAD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ad-Hoc</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Traditional</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Source: http://ambysoft.com/surveys/success2013.htm
Scrum Project Management

A Product Owner creates a prioritized wish list (product backlog).

During Sprint Planning, the team pulls a small chunk from the top of that wish list (Sprint Backlog) and decides how to develop those pieces.

At the end of the Sprint, the work should be shippable (ready to hand to a customer, put on a store shelf, or show to a stakeholder).

The team has a certain amount of time (Sprint) to complete its work, but meets each day to assess its progress, a Scrum.

Extreme Programming

XP Practices

Collective Ownership
Test-Driven Development
Pair Programming
Continuous Integration
Simple Design
Sustainable Pace
Small Releases

Planning Game
Refactoring
Coding Standard
Whole Team

Customer Tests
Extreme Programming Practices

**Business Practices:** Whole Team, Planning Game, Small Releases, Customer Tests, Sustainable Pace
- Every contributor to the project is an integral part of the Whole Team.
- The team forms around a business person called the Customer, who sits with the team and works with them daily.
- A simple form of planning called a Planning Game occurs to decide what should be done next and to predict when the project will be done.
- Focused on business value, the team produces the software in a series of small fully-integrated releases that pass all the tests the Customer has defined.
- Everyone works at a pace that can be sustained indefinitely.

**Technical Practices:** Metaphor, Simple Design, Pair Programming, Test-Driven Development, Design Improvement
- The team shares a common and simple picture of what the system looks like called the Metaphor.
- Extreme Programmers work together in pairs and as a group, with simple design and obsessively tested code, improving the design continually to keep it always just right for the current needs.
- Collective Code Ownership, Coding Standards, Continuous Integration
- Extreme programmers share the code base and code in a consistent style so that everyone can understand and improve all the code as needed.
- The team keeps the system integrated and running all the time.
Creating a High Maturity Agile Implementation

Lean Development

- **Eliminate Waste**
  - Create nothing but value. The less code you write, the less code you have to test.

- **Create Knowledge**
  - Maintain a culture of constant learning and improvement.

- **Decide as Late as Possible**
  - Schedule Irreversible Decisions at the Last Responsible Moment.

- **Deliver as Fast as Possible**
  - Work in small batches – reduce projects size, shorten release cycles, stabilize work environment.

- **Empower The Team**
  - Move responsibility and decision making to the lowest possible level.

- **Build Quality In**
  - Start worrying about it before you write single line of working code.

- **See the Whole**
  - Don't create local inefficiencies, see the whole and optimize the whole organization.
Kanban Values and Practices

- Visualize
- Improve Collaboratively
- Limit WIP
- Manage Flow
- Explicit Policies
- Feedback Loops

1. Start with what you do now.
2. Agree to pursue evolutionary change.
3. Respect the current process, roles, responsibilities.
4. Encourage acts of leadership at all levels.

Kanban Boards

<table>
<thead>
<tr>
<th>Backlog</th>
<th>Design</th>
<th>Develop</th>
<th>Test</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>Story 1</td>
<td>6/1</td>
<td>In Progres</td>
<td>Ready</td>
<td>6/3</td>
</tr>
<tr>
<td>Story 2</td>
<td>6/1</td>
<td>In Progres</td>
<td>Ready</td>
<td>6/5</td>
</tr>
<tr>
<td>Story 3</td>
<td>6/2</td>
<td>In Progres</td>
<td>Ready</td>
<td>6/7</td>
</tr>
<tr>
<td>Story 4</td>
<td>6/3</td>
<td>In Progres</td>
<td>Ready</td>
<td></td>
</tr>
<tr>
<td>Story 5</td>
<td>6/5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Story 6</td>
<td>6/7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

WIP Limit: 20 hrs, 80 hrs, 40 hrs

- Visualize Flow
- Set Work in Progress (WIP) Limits
- Manage Cycle Time

Lead Time = 6 days
Cycle Time = 4 days
Holistic View of Agile

- Strategic
- Evolutionary
- Lean
- Kanban
- Engineering Practices
- Framework
- XP
- Scrum

The Agile Manifesto
Creating a High Maturity Agile Implementation

The Agile Manifesto: A statement of values

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

<table>
<thead>
<tr>
<th>Individuals and interactions</th>
<th>over</th>
<th>Process and tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working software</td>
<td>over</td>
<td>Comprehensive documentation</td>
</tr>
<tr>
<td>Customer collaboration</td>
<td>over</td>
<td>Contract negotiation</td>
</tr>
<tr>
<td>Responding to change</td>
<td>over</td>
<td>Following a plan</td>
</tr>
</tbody>
</table>

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

Source: www.agilemanifesto.org

We value:

"Individuals and Interactions over processes and tools“

- Reduce organization dependency to tools like Microsoft project, requirements tracking, etc.
- Do not build your Agile implementation around a tool.
We value:
"Working software over comprehensive documentation"
- Documentation needs to be prioritized, to those artifacts that provide value to the customer
- The primary focus for software development teams must be on delivering something that works for the customer

We value:
"Customer Collaboration over contract negotiations"
- Listening to what the customer wants is more critical than meeting the language of a contract.
- Software development needs to be a collaboration between the user and the developer not the review of a specification.
We value:

“Responding to change over following a plan”

- Plans are notoriously out of date as soon as they are written.
- By embracing change as opposed to trying to prevent it will deliver products that the customer is actually wanting.

12 Principles of the Agile Manifesto

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 months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity—the art of maximizing the amount of work not done—is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
The Principles

Our highest priority is to satisfy the customer through early and frequent delivery of valuable software.

- Early delivery allows for quick wins and early feedback. The paradigm, “fail sooner” is a valuable benefit.
- Don’t just develop iteratively, deliver iteratively.
- Success in the marketplace is often driven by innovation and frequent releases.

Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage

- The Agile approach strives to accommodate change as easily and efficiently as possible.
- Changes in technology, regulations, resources and requirements are a reality of software development projects.
The Principles

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the smaller time scale.

- It is more rewarding to both the customer and the team to deliver frequently.
- Feedback that occurs after shorter delivery cycles is a key advantage of iterative development.

The Principles

Business people and developers work together daily throughout the project

- There can no longer be an “us” and “them” between business and IT.
- Involve the customer every step of the way.
- Consider Agile approaches that embed the customer with the team.
The Principles

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done

- One of the cornerstones of Agile is the focus on the value of the motivated team.
- Trust first is a major shift in leadership style needed to empower teams.

The Principles

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

- Non-verbal communication is more than 80% of all interactions.
- Informal communication leads to relationship building and creativity.
The Principles

Working software is the primary measure of progress

- Many well run projects often fail because the focus is removed from the actual deliverable.
- This is the number one principle and measurement of a successful Agile team.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

- Software development is a creative process and cannot be forced.
- Burnout and decreased productivity are the outcomes of a long hours and mandatory overtime.
The Principles

Continuous attention to technical excellence and good design enhances agility

- Nothing can replace a clean design and architecture.
- Don’t ignore quality in the rush to deliver more quickly.

The Principles

Simplicity—the art of maximizing the amount of work not done—is essential

- This is a key Lean principle as well – minimize waste.
- Focusing on delivering business value is an important part of this principle.
The Principles

The best architectures, requirements, and designs emerge from self-organizing teams

- This is a combination of some other principles: it is almost impossible to have a high-quality product without a motivated, inspired and skilled team.

The Principles

At regular intervals, the team reflects on how to become more effective and then tunes and adjusts its behavior accordingly

- The commitment to continuous improvement is a foundation of most Agile practices.
- This can only occur if there is an environment of trust and honesty.
The Toughest Principle?

What is an Agile Maturity Model?

- A model that is designed to enhance and improve Agile practices by assessing the current state of your organization
- A way to determine how closely you adhere to Agile principles
- A model which shows your organization on an Agile maturity continuum from an initial or ad-hoc level to a continuously improving, self-sustaining level
# Agile Maturity Matrix

<table>
<thead>
<tr>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
<th>Level 4</th>
<th>Level 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ad Hoc Agile</td>
<td>Doing Agile</td>
<td>Being Agile</td>
<td>Thinking Agile</td>
<td>Culturally Agile</td>
</tr>
<tr>
<td>• Agile is either not used or used inconsistently across organization</td>
<td>• Teams start to exhibit some consistent Agile habits</td>
<td>• Most of the project portfolio is Agile</td>
<td>• Agile habits are at a high maturity across the organization</td>
<td>• Lean and Agile are part of the organizational culture</td>
</tr>
<tr>
<td>• Variable quality</td>
<td>• Knowledge sharing begins to occur across teams</td>
<td>• Role and responsibilities are consistent across teams</td>
<td>• Successful use of Agile at Scale</td>
<td>• Perfecting waste reduction, increased efficiency and a smooth flow of delivery</td>
</tr>
<tr>
<td>• Predominantly manual testing</td>
<td>• Agile tools and practices are common</td>
<td>• Disciplined, repeatable processes are in place with high quality results</td>
<td>• Success across multiple geographies</td>
<td>• Sustainable pace of innovation</td>
</tr>
<tr>
<td>• Success achieved through heroic individual efforts</td>
<td>• Quality improves</td>
<td>• Respect for people and continuous improvement is occurring</td>
<td>• Measurement systems in place to track business value realization</td>
<td>• Continuous organizational learning and optimization of work process</td>
</tr>
</tbody>
</table>

| Score | 0-80 | 81-160 | 161-240 | 240-320 | +++ |

---

**Online Activity**

Agile Assessment
DISCUSSION

What can you do to advance your Agile practice?

Measuring Success in Agile

Business Outcomes (Value)
- Revenue per Employee
- Employee Satisfaction
- Customer Satisfaction

Time to Market (Lead Time)
- Release Frequency
- Release Stabilization
- Cycle Time

Ability to Innovate (Quality)
- Product Cost Ratio
- Installed Version Index
- Usage Index
- Innovation Rate
- Defects

## Current Value Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revenue per Employee</td>
<td>Gross Revenue / #employees</td>
</tr>
<tr>
<td>Product Cost Ratio</td>
<td>All expenses in the organization that develops, sustains, provides services, markets, sells, and administers the product or system.</td>
</tr>
<tr>
<td>Employee Satisfaction</td>
<td>Engaged employees that know how to maintain, sustain and enhance the software systems and products are one of the most significant assets of an organization.</td>
</tr>
<tr>
<td>Customer Satisfaction</td>
<td>Sound management, solid software, and creative, fulfilled employees.</td>
</tr>
</tbody>
</table>

## Time to Market Metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release Frequency</td>
<td>The time needed to satisfy the customer with new, competitive products.</td>
</tr>
<tr>
<td>Release Stabilization</td>
<td>The impact of poor development practices and underlying design and code base. Stabilization is a drag on competition that grows with time.</td>
</tr>
<tr>
<td>Cycle Time</td>
<td>The time (including stabilization) to satisfy a key set of customers or to respond to a market opportunity competitively.</td>
</tr>
</tbody>
</table>
### Ability to Innovate Metrics

<table>
<thead>
<tr>
<th>Installed Version Index</th>
<th>The difficulty customers face installing a new release. The relatively low value of new releases, or even the # of customers that are evaluating alternatives.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usage Index</td>
<td>Determines a product that is burdensome and difficult to use and excess software that must be sustained even though it is rarely used.</td>
</tr>
<tr>
<td>Innovation Rate</td>
<td>Growth of technical debt caused by poorly designed and developed software. Budget is progressively consumed keeping the old software alive.</td>
</tr>
<tr>
<td>Defects</td>
<td>Measures increasingly poor quality software, leading to greater resource and budget to maintain it and potential loss of customers.</td>
</tr>
</tbody>
</table>

### Business Drivers for Agile

- Is speed to market the primary motivator for change? Does the company need to deliver products more frequently?
- Is the organization inefficient? Does it need to be “leaner” and more efficient with existing resources?
- Is quality the issue? Does the company need to improve customer satisfaction with the software it delivers?
- Is the organization new, or is it one that’s trying to reinvent itself and have a more empowered culture?
Creating a High Maturity Agile Implementation

What is your definition of Agile?

<table>
<thead>
<tr>
<th>Goal</th>
<th>Scrum/XP</th>
<th>Kanban</th>
<th>Enterprise Framework</th>
</tr>
</thead>
<tbody>
<tr>
<td>Improve project success</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Be more efficient with resources</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Increase IT quality</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Have a more empowered culture</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minimize Change</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Types of Change

- **Developmental**
  - Improvements on processes, methods or performance standards
  - These are done in order to stay competitive
  - Causes little stress to employees

- **Transitional**
  - More intrusive because it introduces something completely new
  - Examples are re-organization, mergers, acquisition, new technology
  - May cause instability and insecurity

- **Transformational**
  - Occurs after the transition period
  - Transformation may be necessary when there are radical changes within or outside of the company
Creating a High Maturity Agile Implementation

Barriers to Implementing Agile

- ... an “inability to change organizational culture” and the “general resistance to change” are the most commonly cited barriers

Reasons Why Change Efforts Fail

- Inability to identify all the urgent reasons for change
- Failure to point out the one crucial reason for change
- Lack of commitment
- Lack of determination
- Unsuccessful strategizing, execution and comprehension
- Lack of follow through and control
- Impatience to see immediate results
- Inability to adapt and be flexible
- Resistance overpowers the need to change
- Fear
Kotter – 8 Steps to Successful Change

- Establish a sense of urgency
- Create a guiding coalition
- Develop a clear shared vision
- Communicate the vision
- Empower people to act on the vision
- Create short term wins
- Consolidate & build on the gains
- Institutionalize the change

“We are what we repeatedly do. Excellence then, is not an act, but a habit.”

~ Aristotle
Forces of Change

DRIVING Forces

RESISTING Forces

Force Field Analysis

Define the target of change
The center box represents the situation to be moved or changed
Identify which are driving and restraining forces
Analyze the forces to identify which can be changed
Create an action plan to make the changes to the forces
Creating a High Maturity Agile Implementation

Activity

Force Field Analysis

Course Summary

- It is important to Align your Agile method with your organizations needs and tolerance for change.
- Agile implementations progress through a maturity model
- An effective Agile implementation results in an effective organization
- Fear of change is the biggest obstacle to implementing Agile
Force Field Analysis

Directions

1. Use the worksheet on the next page.

2. On the center box, the change you are anticipating.

3. List all the forces FOR CHANGE in one column, and all the forces AGAINST CHANGE in another column.

4. Rate the strength of these forces and assign a numerical weight, 1 being the weakest, and 5 being the strongest.

5. When you add the “strength points” of the forces, you'll see the viability of the proposed change.

The tool can be used to help ensure the success of the proposed change by identifying the strength of the forces against the change.
Implementing Agile

Forces for Change

Score

Score

Forces against Change

Total Score:

Total Score: