ADAPTing to Agile: A Guide to Transitioning

Mike Cohn
9 November 2009

Mike Cohn - background

Agile coach and trainer
- Founding member and director of Agile Alliance and Scrum Alliance
- Founder of Mountain Goat Software
- Ran my first Scrum project back in 1995
- Typical programmer to manager etc. progression

© Mountain Goat Software, LLC
Topics today...

1. Why transitioning to agile is hard
2. ADAPTing to agile development
3. Iterating toward agility
4. The role of leadership
5. Overcoming resistance

Why Transitioning to Agile Is Hard
1. Change is not top-down or bottom-up; it’s both

- Two simplistic views of change:
  - Top down
    - Powerful leader shares a vision
  - Bottom-up
    - A team starts and everyone else sees the benefits of the new approach
- But, transitioning to agile is neither top-down nor bottom-up
  - It’s everywhere, all together, all-at-once

2. Best practices are tempting

- It is tempting to codify things that work in a given context into best practices
  - This leads to inflexible processes†
- Once we know what’s “best” we stop adapting
  - Or even thinking about what we’re doing
- Once we’ve stopped inspecting and adapting we’re not agile, or won’t be for long

†Source: Anderson, P. “Seven Layers for Guiding the Evolving Enterprise” in The Biology of Business.
The transition process must be congruent with the development process.

Part of the move to agile is a move to self-organizing teams.

Moving to self-organization requires self-organization.

4. Organizations are unpredictable, living systems

- Traditional view of the organization is as a machine
- We can disassemble it, study its parts, put it back together
- Once we set it in motion, it will continue in motion
We need a different mental model

- The organization as a Complex Adaptive System (CAS)

- A dynamic network of many agents
  - acting in parallel
  - acting and reacting to what other agents are doing
- Control is highly dispersed and decentralized
- Overall system behavior is the result of a huge number of decisions made constantly by many agents

John Holland in *Complexity: The Emerging Science at the Edge of Order and Chaos* by Mitchell Waldrop
Local goals and gaps

- Local agents (individuals, project teams, discipline coworkers) identify local gaps based on their local goals.

Differing views of success

Newtonian view
Success = closing the gap with the desired state

CAS view
Success = achieving a good fit with the environment
Each paired statement below and on the next slide describes either the traditional or CAS view of how to change an organization.

Put an X in the appropriate column to indicate which describes the traditional view and which the CAS view.

<table>
<thead>
<tr>
<th></th>
<th>Traditional view</th>
<th>CAS view</th>
</tr>
</thead>
<tbody>
<tr>
<td>Behavior is predictable and controllable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Behavior is unpredictable and uncontrollable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Direction is determined through emergence and by many people</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Direction is determined by a few leaders.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Every effect is also a cause</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Every effect has a cause</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Traditional view</th>
<th>CAS view</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relationships are directive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Relationships are empowering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Responsiveness to the environment is the measure of value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Efficiency and reliability are measures of value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Decisions are based on facts and data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Decisions are based on patterns and tensions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Leaders are experts and authorities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Leaders are facilitators and supporters</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ADAPTing to Agile Development

- Awareness that the current approach isn’t working
- Desire to change
- Ability to work in an agile manner
- Promote early successes to build momentum and get others to follow
- Transfer the impact of agile throughout the organization so that it sticks
...the developers are not meeting expectations for code quality. One of our challenges is that we’re still hacking our way through lots of legacy code that isn’t unit-testable or automated yet, but is mission critical and the person who has been working mostly on that area of code consistently leaves holes in the design and implementation of new pieces of that code. We also have the issue with at least one other developer as well.

1. Is this a problem of Awareness, Desire or Ability?
   • Why?
2. Thinking about ADAPT, what might you try?

Individual and group change

• All individuals will need to move through the Awareness, Desire, and Ability stages
  • But will do so at different rates
• Early adopters and leaders:
  • Use the Promote stage to build Awareness and Desire in later adopters
  • Need to Transfer the impact of agile to groups like Human Resources or the transition will fail
Tools for building...

**Awareness**
- Communicate that there’s a problem
- Use metrics
- Provide exposure to new people and experiences
- Focus attention on the most important reason or two for changing

On the following slides, identify at least five ways to build desire, create ability, promote the transition, and transfer its implications.
Stop!

- My thoughts on desire, ability, transfer and promote are on the following pages.
- You don’t want to see them before you think about your own.

Desire

- Communicate that there’s a better way
- Create a sense of urgency
- Build momentum
- Get the team to take agile for a test drive
- Align incentives (or, at least, remove disincentives)
- Focus on addressing any fears
- Help people let go
- Don’t discredit the past
- Engage everyone in the transition
Ability
- Provide coaching and training
- Hold individuals accountable
- Share information
- Set reasonable targets
- Just do it

Promotion
- Publicize success stories
- Host an agile safari
- Attract attention

Transfer
- Transfer the effects of agile beyond the current group
  - A team transfers to its department
  - A department transfers to its division
  - etc.
- If you don’t transfer, the transition will eventually and inevitably fail
  - Too much organizational gravity pulling us back toward the status quo
- Example:
  - If you don’t align promotions, raises, annual reviews, those will work against you
Iterating toward agility

Enterprise Transition Community (ETC)

Improvement backlog

Energy, support, resources, guidance, & direction (occasionally)

Impediments

Improvement Communities

Improvement backlog

Improvement backlog

Improvement backlog

© Mountain Goat Software, LLC
Enterprise Transition Community

- Creates a culture in which passion and desire to improve thrive
- Does not direct the transition effort
  - Provides energy, resources, support and guidance
  - Removes organizational impediments to agility
- Encourages Improvement Communities to form

ETC members

- Sponsor
  - From highest level at which change is supported
  - Not a checkbook-only commitment
- Others
  - From any level but driven by desire to improve
- Disbands when the “transition” part of adopting agile is over
ETC responsibilities

- Articulate the reasons for adopting agile
- Stimulate conversation
- Provide resources
- Engage everyone
- Set appropriate aspirations
- Anticipate and address people issues and other impediments
- Encourage simultaneous focus on practices and principles

An ETC’s improvement backlog

<table>
<thead>
<tr>
<th>Item</th>
<th>Who</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an “Agile Office” where teams can get help.</td>
<td>Jim (CTO)</td>
<td>Jim to talk this up at monthly development meeting. Let’s see if there’s any interest.</td>
</tr>
<tr>
<td>Establish an internal program for developing ScrumMasters.</td>
<td></td>
<td>How do we identify good internal candidates? How do we develop them?</td>
</tr>
<tr>
<td>Collect and disseminate Scrum success stories in our company.</td>
<td>SC</td>
<td>Savannah has expressed interest in this.</td>
</tr>
<tr>
<td>Resolve dispute with facilities over rearranging second floor cubicles.</td>
<td>JS</td>
<td>Jim to talk to Ursula in facilities</td>
</tr>
<tr>
<td>Get more teams to do continuous integration.</td>
<td>AR</td>
<td>Arie will summarize metrics from his project and see how many teams he can motivate.</td>
</tr>
</tbody>
</table>
Improvement communities (ICs)

- Form around the passion of a small number of people
- Expand from there
- Do the real work of improving how the organization implements agile
- Focus on goals with practical relevance
- Examples:
  - ScrumMaster, Testing, Product Owner, Continuous Integration

Working with an IC

- An IC works with a project team
- Work is not done in an ivory tower
- Most ICs work in 2–4-week iterations
- Disband or refocus when goal has been achieved
ETC Improvement Backlog

... 

Establish an internal program for developing ScrumMasters.

...

An IC Improvement Backlog

Figure out how to identify good candidates to become ScrumMasters (in addition to those who ask to participate in this program).

Establish an internal mentoring program.

Develop some internal classroom training. Which courses? Who can teach them? Can we license courses?

Get budget for next year for external coaching. How many days? At what expected daily rate?

Not everything on an IC’s improvement backlog needs to tie back to the ETC’s backlog

Improvement backlogs

1. Write some items that you would like an on your organization’s initial improvement backlogs.
2. What improvement communities would you like to see form that could help with some of these improvements?
3. What obstacles are in the way of making these improvements?
4. What obstacles are in the way of forming an ETC or improvement community to get started?
The Role of Leadership

Leading an agile transition

- Transition team and other formal leaders must lead the transition
  - but cannot do so in the usual ways
- Self-organizing groups still require leadership
- Lead through example, questions, and focus
  - “Nudge” the organization; Poke and prod;
  - See how the organization responds
Pre-requisites of self-organization

- **Container**
  - A boundary within which self-organization occurs
  - Company, project, team, city, role, nationality

- **Differences**
  - There must be differences among the agents acting in our system
    - Technical knowledge, domain knowledge, education, experience, power, gender

- **Transforming Exchanges**
  - Agents in the system interact and exchange resources
    - Information, money, energy (vision)

Using the CDE model

- When stuck thinking about how to nudge the organization think of the:
  - **Containers**
    - formal teams, informal teams, clarify (or not) expectations
  - **Differences**
    - Dampen or amplify them within or between containers
  - **Exchanges**
    - Insert new exchanges, new people, new techniques or tools
Containers

- Enlarge or shrink teams
- Enlarge or shrink the responsibility boundary of teams
- Change team membership
- Create new teams or groups

Differences

- Don’t require consensus
  - Creativity comes from tension
  - Quiet disagreement is not as good as fierce debate that leads to behavior change
- Ask hard questions
  - Then expect teams to find solutions
Transforming exchanges

- Encourage communication between teams and groups
  - Who isn’t talking that should?
- Add or remove people from exchanges
  - Change reporting relationships
  - Relocate people
- Encourage learning

You are the ScrumMaster or project manager...

- The next set of slides describes some teams with some trouble spots. Think about how you might help them by changing their Containers, amplifying or dampening Differences, or changing their Exchanges.
- For each case, identify at least one thing you’d do.
- Note whether you are tweaking their Container, Differences, or Exchanges. (You might be affecting more than one.)
The team consists of four developers, two testers, a database engineer and you. The developers and testers are not working well together. Developers work in isolation until two days are left in the iteration. They then throw the code “over the wall” to the testers.

The team is failing to deliver potentially shippable software at the end of each iteration. None of the items they start are 100% finished. They’re close but work is always left to be done in the next iteration.

The team seems to be consistently undercommitting during iteration planning. They finish the work they commit but it doesn’t seem like much. The product owner hasn’t complained yet but you’re worried she will soon.

Your organization has 20 different agile teams. Each team has its own testers who are starting to go in different directions in terms of preferred tools and approaches.
Jeff, a senior developer, is very domineering. During iteration planning the team defers to him on every decision even though he is a horrible estimator. You notice the glances that other team members exchange when he suggests very low estimates on some tasks.

You are responsible for two teams. Team members on one discuss all sides of various issues before making a decision. This has been working well. On the other team, discussions drag on endlessly because they pursue absolute consensus in all cases.

The self-organizing path

- Self-organization is not something that happens one time
  - A team is never done doing it
  - The team continually re-organizes in a sense-and-respond manner to its environment
- As you see the team self-organize you can influence, but not control or direct, its path
- We can view this as the evolution of a team
Self-organization proceeds from the premise that effective organization is evolved, not designed. It aims to create an environment in which successful divisions of labor and routines not only emerge but also self-adjust in response to environmental changes. This happens because management sets up an environment and encourages rapid evolution toward higher fitness, not because management has mastered the art of planning and monitoring workflows.

“Seven Levers for Guiding the Evolving Enterprise,” in The Biology of Business edited by John Henry Clippinger III.

~Philip Anderson

Variation, selection & retention

- Evolution is the result of three elements:
  - Variation, selection and retention
- Consider a giraffe:
  - **Variation:** A random mutation that leads to a longer neck
  - **Selection:** The long neck helps it reach food others can’t; so it is more likely to survive and breed
  - **Retention:** The mutation is passed to its descendants
Three ways to influence team evolution

1. Define performance
2. Manage meaning
3. Evolve vicarious selection systems

Define performance

- The principle of selection tells us that the traits that help us survive will be the ones retained
- Managers and leaders send messages about which traits should survive
- What message is your organization sending about the relative importance of short vs. long-term performance?
  - What messages are sent if the organization:
    - Provides training
    - Supports working at a sustainable pace
    - Allows employees time to explore wild ideas
    - Doesn’t exchange meeting a deadline for unmaintainable
2 Manage meaning

- Individuals in a CAS respond to the messages they receive
  - bees responding to a “danger” message
  - ants responding to a “food found over here” message
- Leaders can push messages into the system
  - e.g., putting the team in touch with customers
- Or keep messages out
- Meaning often comes from the stories, myths and rituals that are repeated
  - “We will become profitable this quarter.”
  - “Our GM counts the cars in the lot every day at 5 PM”

3 Evolve vicarious selection systems

- Selection determines which variations will be retained
  - Can take a long time
- So we often use vicarious selection systems
  - Example: an animal that can smell that a food is poisonous, rather than eating it
- Using only the marketplace as our selection mechanism takes too long
- Organizations also evolve vicarious selection systems
  - Retrospectives, Google’s 20% policy, improvement communities
Overcoming Resistance

Why They Resist

Like status quo
Dislike agile

How They Resist

Active

Passive

Diehards
Saboteurs

Followers
Skeptics

Why They Resist

© Mountain Goat Software, LLC
Handling resistance

- Pick one type of resistor.
- Identify some underlying reasons for the behavior and some things you could try to overcome it.

![Resistance types diagram]

Skeptics
- Let time run its course
- Provide training
- Solicit peer anecdotes
- Appoint a champion skeptic
- Push the issue
- Build awareness
**Saboteurs**

- Success
- Reiterate and reinforce the commitment
- Move them
- Fire them
- Encourage a thriving set of communities

**Diehards**

- Align incentives
- Create dissatisfaction with the status quo
- Acknowledge and confront fear
Followers

- Change team composition
- Praise the right behavior
- Model the right behavior
- Involve them
- Identify the true barrier (awareness, desire, ability)

Upcoming public classes

<table>
<thead>
<tr>
<th>Date</th>
<th>What</th>
<th>Where</th>
</tr>
</thead>
</table>
| Feb 1–2    | Certified ScrumMaster  
             Certified Scrum Product Owner | Dallas |
| Feb 3–4    |                                                                      |        |
| March 1    | User Stories for Agile Requirements  
             Certified ScrumMaster  
             Agile Estimating & Planning | Boulder|
| March 2–3  |                                                                      |        |
| March 4    |                                                                      |        |
| April 12   | User Stories for Agile Requirements  
             Certified ScrumMaster  
             Succeeding with Scrum (new!) | San Diego|
| April 13–14|                                                                      |        |
| April 15   |                                                                      |        |

See [mountaingoatsoftware.com](http://mountaingoatsoftware.com) for details