Agile Adoption & Transformation Strategies

Mike Cottmeyer, Agile Coach
LeadingAgile, LLC
PART ONE: Exploring the Mechanisms of Process Adoption

PART TWO: Understanding the Scaled Agile Delivery Framework

PART THREE: Defining our Adoption & Transformation Strategy
PART ONE: Exploring the Mechanisms of Process Adoption
Agile Adoption vs. Agile Transformation
Adoption vs. Transformation

• Agile Adoption is more about what you do... practices, tools, techniques, and habits

• Agile Transformation is more about who you are... reflected in both the structure of the organization and who you are as people
Adoption/Transformation Cycle

Introducing Agile is Iterative and Incremental

- Changing some of the physical structures in our organization
- Teaching people new ways of working together
- Helping people internalize how and why agile really works... living the value system
Adoption/Transformation Cycle

Organizational Transformation

- Establish a top-down organizational design pattern and roadmap
- Make incremental changes to the structure of the organization, build Agile teams
- Establish policies and working agreements

Personal Transformation

Adopt Practices
Adoption/Transformation Cycle

Organizational Transformation

Personal Transformation

Adopt Practices

- Instantiate sprint planning, daily standups, reviews & retrospectives
- Identify a ScrumMaster and Product Owner
- Start doing TDD and Continuous Integration
- Story Mapping and Agile Requirements decomposition
Adoption/Transformation Cycle

Organizational Transformation

Personal Transformation

Adopt Practices

**Personal Transformation**

- Develop a greater ability to deal with ambiguity and inspect and adapt
- Help people to work together toward common goals
- Demonstrate greater understanding and empathy amongst individuals
- Teamwork
The goal is never to adopt agile...
...The goal is to improve business outcomes!
Delivery Goals

- Faster Time to Market... frequent, smaller releases
- High quality products... low number of escaped defects
- Efficient delivery... minimize waste
- Predictability... deliver what we say we are going to deliver
- Happy Customers
Product Definition

- Extremely clear product vision
- Well articulated product roadmap
- Decompose product themes into fine grained product features
- Define detailed acceptance criteria and make trade-offs
- Accurately estimate effort and duration
Planning & Coordination

- Establish a regular planning cadence
- Break down features into tasks and define acceptance tests
- Establish a cadence of delivering working tested features
- Limit the amount of work in process
- Regularly make and meet commitments
- Continually adapt the plan
Delivery Practices

- Emergent Architecture and Design
- Rapid delivery of small features into the product
- Continuous testing of the product
- Clean code and fixing defects as we go
- Deploy the solution either internally or externally on demand
Continuous Improvement

- Metrics and Reporting
- Teams that have a stable velocity
- Retrospectives at all levels of the organization
- Manage stakeholder expectations
- Enable process improvement
- Lead and manage change
Organizational Enablement

- Team based delivery
- Regular checkpoints to make sure everyone is on the same page
- People work together to solve problems
- People are empowered to make decisions
- High levels of trust between coworkers
Focus less on implementing specific agile practices...
...and more on developing situationally specific strategies to solve business problems.
Agile Competency Model
Product Definition

- **Establish the Product Vision**
  - The ability to determine and clearly communicate the product’s primary customer base, its competitive differentiators, and competitive alternatives. At the release level, it’s the ability to determine why we are building this product, whom it is for, and why the release is important.

- **Define the Product Roadmap**
  - The product roadmap is the strategic plan for how the Product Vision will be executed. In an agile organization, this roadmap should be at the Epic level and show when various Epics need to be in market. The Product Roadmap should be supported with either Epic size or budget, and be validated against proven capacity to deliver. Epics are generally 1-3 months in size.

- **Decompose Features**
  - The ability to decompose features means to break Epics into high-level feature functions that can be communicated and/or committed to customers. Features follow the same format as user stories, but are at a higher level of abstraction, much like use cases, or use case scenarios. Features are generally take 2-4 weeks to deliver.

- **Estimate Size and Effort**
  - Does the organization have the ability to accurately estimate the size and effort of a given Epic, Feature, or User Story? Are these estimates generated using team based, collaborative techniques? Are these estimates validated with empirical evidence gathered from actual delivery of working software?

- **Define Acceptance Criteria**
  - Does the team have clear guidance on what is the definition of done? Do they know what it will take to meet the business requirements defined by the product owner?
Planning & Coordination

- **Establish a Planning Cadence**
  - Is there a release train in place? Is there a regular release-planning cadence? Do the teams meet regularly with their Product Owner to plan sprints? Does the organization do strategic planning and roadmap planning?

- **Perform Activity Breakdown**
  - Do the teams break user stories into sufficiently small increments that they can be incrementally delivered and tracked through the sprint? If not, do the teams break larger user stories down into tasks and tests that can be tackled by more than one team member at a time

- **Establish Delivery Cadence**
  - Does the team have a pattern of delivering working, tested software every sprint? Do multi-team projects show a pattern of coming together to deliver an integrated increment of software on a regular, periodic basis? Does the organization show a pattern of early delivery of whole Epics in a release cycle?

- **Limit Work in Process**
  - Does the organization, release architecture, or team have the ability to limit the number of Epics, Features, or User Stories they are working on concurrently? Does the organization value completing work rather than getting new work started? Are teams allowed to focus on one thing until delivery, or are they constantly pulled onto other, higher priority initiatives? Do priorities change often?

- **Make and Meet Commitments**
  - Does the organization, release, or team regularly do what it says it will do? Do they have the ability to make and meet commitments on short time-boxed intervals?
Delivery Practices

- **Define the Solution**
  - Does the team have the capability to allow architectures and designs to change as we learn more about the emerging product? Do the team use agile modeling techniques? Is there a desire to plan everything before we start building any working product? Does the team use a value and risk driven approach to working out the systems architecture and design.

- **Build the Solution**
  - Do the developers have the tools necessary to build an increment of working software? Can software be checked in and validated on a continuous basis. Are the teams doing unit testing? Are the tests run in a test harness at check in? Are the developers confident working in the code base? Is the code safe to change?

- **Test the Solution**
  - Is there a mechanism in place to incrementally test and validate the software as it is being built? Is all software tested before it is accepted? Is all software tested before it is put into production? Are the number of ‘hardening sprints’ equal to or greater than the number of sprints it took to build the product? Are the teams using continuous integration and TDD?

- **Technical Debt and Defects**
  - Are defects routinely carried over from sprint to sprint and handled toward the end of the release? Do teams have difficulty estimating work due to unexpected defects and code that is difficult to understand and maintain?

- **Deploy the Solution**
  - Is there an ability to incrementally deliver the solution, either to an internal customer for review, or to an external customer that will actually use the product?
Continuous Improvement

• **Metrics and Reporting**
  – Does the organization have a package of agile metrics that support team level up to executive level decision-making? Are there processes in place for gathering these metrics and reporting them to the appropriate stakeholders?

• **Establish Stable Velocity**
  – Can the organization, at the enterprise, release, or team levels; reliably and predictably deliver a known quantity of working software at every iteration or release boundary?

• **Conduct Retrospectives**
  – Does the organization regularly conduct reviews and retrospectives at the end of every iteration or release boundary? Is there a mechanism for acting on lessons learned and new opportunities discovered in these sessions?

• **Update the Release Backlog**
  – Is there a mechanism in place for quickly updating the release backlog when new information is learned about the emerging product or when business priorities change?

• **Enable Process Improvement**
  – Is there a mechanism in place for quickly updating organizational processes in the face of impediments that might impact product delivery?
Organizational Enablement

• **Team Based Delivery**
  – Is the organization formed around agile teams? Do the teams have everything they need to successfully deliver an increment of working tested software? Are members constantly pulled away from teams and assigned to other initiatives. Is there team level accountability for sprint outcomes? Is there team level accountability for release level outcomes?

• **Communication**
  – How well do people talk to each other and communicate the right level of information? Do teams openly and honestly share information that could help the organization be more successful?

• **Collaboration**
  – Do cross-functional teams regularly work together to define requirements, architectures, designs, test plans, etc.? Do team members often work in silos with limited communication amongst team members?

• **Empowerment**
  – Are people and teams authorized to make decisions within their established constraints or within pre-defined guidelines? Are decisions routinely overturned? Are the right stakeholders present when decisions are made?

• **Trust**
  – Is there open and honest communication between team members? Is it safe to share bad news? Does management ‘shoot the messenger’ when bad news is delivered? Do people feel they can openly and honestly give negative feedback?
Applying the Agile Competency Model at Different Frequencies
Continuous
Daily

Continuous
Iteration

- Continuous
- Daily
- Iteration
Strategic

Release

Iteration

Daily

Continuous

Strategic

Itera(ion

Daily

Continuous
Applying the Agile Competency Model at Different Levels of Scale
Team
Multi-Team
Multi-Team
Multi-Team
Program Management
Portfolio Management
Enterprise Agility
How do the three dimensions Fit Together?
Create Situationally Specific Strategies for Each Competency, Frequency & Scale Combination
125 Possible Combinations

- Competency: *Continuous Integration*
- Frequency: Daily Build
- Scale: Across Multiple Teams
125 Possible Combinations

- Competency: **Continuous Integration**
- Frequency: Daily
- Scale: Across Multiple Teams

- Competency: **Continuous Integration**
- Frequency: Release
- Scale: At the Portfolio Level
125 Possible Combinations

- Competency: Define the Product
- Frequency: Iteration Planning
- Scale: Single Team
125 Possible Combinations

- Competency: Define the Product
- Frequency: Iteration Planning
- Scale: Single Team

- Competency: Define the Product
- Frequency: Strategic Planning
- Scale: Entire Enterprise
125 Possible Combinations

• Competency: Explore Improvement Options
• Frequency: Iteration Planning
• Scale: Single Team
125 Possible Combinations

- Competency: **Explore Improvement Options**
- Frequency: Iteration Planning
- Scale: Single Team

- Competency: **Explore Improvement Options**
- Frequency: Release Planning
- Scale: Portfolio Management
Assessing the Organization and Getting Better Over Time
Assessment Results – at Start
Assessment Results – at Month One
Leading Change by Incrementing and Iterating Through the Organization
Incremental and Iterative Delivery

Incremental

Various parts of the system are developed at different times or rates, and integrated as they are completed. You can do this in a waterfall project or an iterative project.

Iterative

Go back over parts of the system to revise and improve the system. In iterative development testing and/or user feedback is used to revise the targets of the successive deliverables. The practice of iterations arises from a desire to coordinate feedback from increments to revise a future deliverable.

Courtesy of Jeff Patton
Organizational Transformation

Personal Transformation

Adopt Practices

Value Delivery

Phase I

Product Definition
Planning Coordination
Delivery Practices
Continuous Improvement

Organizational Enablement
Organizational Transformation

Personal Transformation

Adopt Practices

Phase I

Value Delivery

Product Definition
Planning Coordination
Delivery Practices
Continuous Improvement

Organizational Enablement
Phase I

Organizational Transformation

Personal Transformation

Adopt Practices

Product Definition

Planning Coordination

Delivery Practices

Continuous Improvement

Organizational Enablement

Value Delivery
Organizational Transformation

Personal Transformation

Adopt Practices

Phase II

Value Delivery

Product Definition

Planning Coordination

Delivery Practices

Continuous Improvement

Organizational Enablement
Value Delivery

Phase II

Organizational Transformation
Personal Transformation
Adopt Practices

Product Definition
Planning Coordination
Delivery Practices
Continuous Improvement

Organizational Enablement
Organizational Transformation

Personal Transformation

Adopt Practices

Phase III

Value Delivery

Product Definition
Planning Coordination
Delivery Practices
Continuous Improvement

Organizational Enablement