Agile Acquisition

Pete Modigliani
pmodigliani@mitre.org
(703) 983-9131

8 May 18

See more at aida.mitre.org
Purpose / Outline

Discuss adoption of Agile development practices in federal acquisition

- Agile Overview
- Program Structure
- Requirements
- Contracting
- Cost Estimation
- Testing
- Barriers and Enablers
Agile Guiding Principles

Small, Frequent Releases

Responsive to Changes in Ops, Technologies, Budgets, Threats

Review Working Software Not Extensive Docs

Active User Involvement

Agile Acquisition: How IT Acquisition programs can leverage Agile Software Development practices

Tailored from Agile Manifesto (see backup slide)
Agile Is. . .

- A mindset and cultural change, not simply a process to follow
- A journey of continuous improvement
- Tailored for your organization’s vision, culture, and needs

https://enterprise-knowledge.com/ek_agileorganizations/
Large Software Projects Rarely Succeed

Two Simple Rules
1. Smaller is better
2. See #1

<table>
<thead>
<tr>
<th>Project Size</th>
<th>Successful*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grand</td>
<td>6%</td>
</tr>
<tr>
<td>Large</td>
<td>11%</td>
</tr>
<tr>
<td>Medium</td>
<td>12%</td>
</tr>
<tr>
<td>Moderate</td>
<td>24%</td>
</tr>
<tr>
<td>Small</td>
<td>61%</td>
</tr>
</tbody>
</table>

*Success*: On Time, On Budget, Satisfactory Result

Source: Standish Group 2015 CHAOS Report
Benefits of Agile

- Responsive to Changes
- Faster Deliveries
- Higher Satisfaction
- Reduced Risk
- Improved Productivity
- Improved Quality
- Improved Visibility

Related report at VersionOne
# Programs Well-Suited for Agile Adoption

<table>
<thead>
<tr>
<th>Software Intensive Systems</th>
<th>Accessible Stakeholders</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Incrementally deliverable</td>
<td>• Users, relevant representatives can engage throughout</td>
</tr>
<tr>
<td>• Rapid delivery</td>
<td>• Able to share ops insights, feedback</td>
</tr>
<tr>
<td>• Short lifespan</td>
<td></td>
</tr>
<tr>
<td>• Loosely coupled architecture</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Uncertain Solution Space</th>
<th>Follow-on Increments</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Rapidly changing demand, technology</td>
<td>• First increment may be infrastructure via traditional methods</td>
</tr>
<tr>
<td>• Evolving requirements</td>
<td>• Iteratively develop capabilities</td>
</tr>
</tbody>
</table>
# Acquisition via Traditional vs Agile Methods

<table>
<thead>
<tr>
<th></th>
<th>Traditional</th>
<th>Agile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mindset</td>
<td>Define rigid requirements, design, develop, produce</td>
<td>Collaborative culture to iteratively deliver priority capabilities to users</td>
</tr>
<tr>
<td>Size/Scope</td>
<td>5-Year Increments</td>
<td>&lt;6 month releases</td>
</tr>
<tr>
<td>Requirements</td>
<td>Defined upfront via large requirement documents and contracts</td>
<td>Iteratively defined and prioritized via dynamic backlogs</td>
</tr>
<tr>
<td>Contracts</td>
<td>Rigid, product based, long timelines, limited changes</td>
<td>SW Development-as-a-Service via iterative task orders</td>
</tr>
<tr>
<td>Cost Estimate</td>
<td>Exhaustive upfront analysis, rigid baselines</td>
<td>Iterative, integrated, collaborative</td>
</tr>
<tr>
<td>Testing</td>
<td>Long timelines following system development</td>
<td>Automated, daily, integrated throughout development</td>
</tr>
</tbody>
</table>
Notional Agile Terms and Timelines

**Release**

- **≤6 Months**
  - Sprint
  - Sprint
  - Sprint
  - Sprint
  - Sprint

**Sprint**

- **≤1 Month**
  - Prioritized capabilities developed, integrated, tested
  - Demonstrated to users – Potential to deliver capability

How to go from user story to code
- Design
  - How to go from user story to code
- Develop
  - Develop code and track tasks
- Integrate
  - Continuously, at least daily
- Test
  - Automated and integrated testing

**Demo**
- Demonstrate functionality to users and stakeholders

Agile terms and timelines will vary and often shorten over time

© 2017 The MITRE Corporation. All rights reserved.
Structuring an Agile Release

- Develop structured time-box and tailor processes to support
  - Hold schedule, while flexing scope – Continual improvement

- Gov’t testers, certifiers, and users involved EARLY and often
  - Minimizes rework and surprises at the end of the release
  - Maximize use of automation – integration, build, regression testing

Lengths Based on Operational, Acquirer, Contractor Agreement

12-month alternative in backup
Tailored Agile Model on Agile Fundamentals Overview

MATERIEL SOLUTION ANALYSIS (MSA) PHASE
- Materiel Development Decision (MDD)
- Analyze Requirements
- Analysis of Alternatives (AoA)
- Develop Acquisition Strategy
- Market Research
- Cost Estimation
- Risk Management

TECHNOLOGY MATURITY AND RISK REDUCTION (TMRR) PHASE
- Milestone A
- Mature Requirements
- Competitive Prototyping
- Systems Engineering
- Mature Acquisition Strategy
- Contract Preparation
- Risk Management
- Request for Proposal

ENGINEERING & MANUFACTURING DEVELOPMENT (EMD) PHASE
- Milestone B
- Manage Program Backlogs
- Release Execution
- Manage Contracts
- Metrics
- Scaling Agile

Detailed how-to guidance for adopting Agile by functional area and across the lifecycle

See more at aida.mitre.org
Agile Requirements Backlog

- An evolving, prioritized inventory of requirements
  - Near term requirements are more detailed
- Integrates operational and technical requirements
  - Requirements captured in user stories
  - Architecture, cybersecurity, and SCRM integrated
- Product owner (representing users) manages backlog
  - Collaborates with end users on priorities, issues, operations

Iteratively Define, Prioritize, and Tackle Requirements
Agile Requirements Can Be in User Stories

- Concise, written descriptions of a capability valuable to a user
- High-level description of features
- Written in user language, not technical jargon
- Provides information to estimate level of effort
- Small and succinct
- Worded to provide a testable result
- Traceable to overarching mission threads
Potential Agile Team Construct

Multi-Team Alternatives in Backup
# Traditional vs. Agile Contracts

<table>
<thead>
<tr>
<th>Contracting Area</th>
<th>Traditional</th>
<th>Agile</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Timelines</strong></td>
<td>Rigid contracting processes with long contracting timelines</td>
<td>Streamlined contracting processes</td>
</tr>
<tr>
<td><strong>Requirements</strong></td>
<td>Defined in detail up front</td>
<td>Ability to <strong>re prioritize requirements</strong></td>
</tr>
<tr>
<td><strong>Scope</strong></td>
<td><strong>Locked in</strong> at contract award</td>
<td><strong>Flexible contracts</strong> with ability to “learn” from development processes</td>
</tr>
<tr>
<td><strong>Contractors</strong></td>
<td>Traditional contractors with no or limited Agile experience</td>
<td><strong>Qualified contractors</strong> with experience in Agile</td>
</tr>
<tr>
<td><strong>Contract Management</strong></td>
<td>Inconsistent contract management</td>
<td>Close management of <strong>Government contractor relationships</strong></td>
</tr>
<tr>
<td><strong>Incentives</strong></td>
<td>Incentivized to deliver against fixed requirements</td>
<td>Incentivized to be <strong>efficient and collaborative</strong></td>
</tr>
<tr>
<td><strong>Technical Evaluation</strong></td>
<td>Award made based on strength of the technical solution</td>
<td>Award based on <strong>strength of the team</strong> and experience with Agile</td>
</tr>
</tbody>
</table>
## Product vs Services Based Contracts

<table>
<thead>
<tr>
<th></th>
<th>Product</th>
<th>Services</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Core Strategy</strong></td>
<td>Acquire a defined software product</td>
<td>Acquire time of an Agile developer</td>
</tr>
<tr>
<td><strong>Selection</strong></td>
<td>Strongest technical solution</td>
<td>Strongest development team</td>
</tr>
<tr>
<td><strong>Requirements Changes</strong></td>
<td>Costly and Timely</td>
<td>Flexibility</td>
</tr>
<tr>
<td><strong>Gov’t / Contractor Relationship</strong></td>
<td>Separate – Less visibility</td>
<td>Enables close teaming</td>
</tr>
<tr>
<td><strong>System Integrator</strong></td>
<td>Contractor</td>
<td>Government</td>
</tr>
<tr>
<td><strong>Development Strategy</strong></td>
<td>Contractor Driven</td>
<td>Government Driven</td>
</tr>
<tr>
<td><strong>Contract Types</strong></td>
<td>FFP, Cost Reimbursement Completion</td>
<td>FFP, T&amp;M, Cost Reimbursement Term</td>
</tr>
<tr>
<td><strong>Summary</strong></td>
<td>More difficult for Agile</td>
<td>Best option for Agile</td>
</tr>
</tbody>
</table>
Elements of a Potential Agile Contract Strategy

- **Portfolio-level (capability-based) agile development contract**
  - Manage development via quick execution of orders for each release (e.g., 6 months) within each phase
  - T&M for max flexibility (transition to FFP or CR after initial period)
  - Scope and requirements can adjust over time

- **Services-based contract: SW Development As A Service (SDAAS)**
  - Contract for the services of the development team
  - Cost-boxed and time-boxed releases and sprints
  - Requirements in product backlog are flexible, prioritized by Gov’t
  - Structure releases (e.g. 6 months) via separate task orders

Devise Strategy to Embrace Change and Rapid Deliveries
Cost Considerations

- Cost estimating techniques for an Agile program are not very different, but do require an iterative, integrated, and collaborative approach.

- Early-on, cost estimates will be required for the entire project.

- Detailed cost estimates will be developed prior to each release.

- Agile promises some real cost savings however, many key benefits of Agile may not be realized as cost savings.
## Life Cycle Cost Impact

**Largest Area of Potential Decreased Cost Lies in Sustainment**

<table>
<thead>
<tr>
<th>Life Cycle Cost Element</th>
<th>Cost Impact Range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Best Case</td>
</tr>
<tr>
<td>Program Management/ System Engineering</td>
<td>=</td>
</tr>
<tr>
<td>Software Development</td>
<td>-</td>
</tr>
<tr>
<td>Integration and Test</td>
<td>=</td>
</tr>
<tr>
<td>Fielding/Deployment</td>
<td>=</td>
</tr>
<tr>
<td>Training</td>
<td>+</td>
</tr>
<tr>
<td>Sustainment</td>
<td>--</td>
</tr>
</tbody>
</table>

++ significant increase, + increase, = no impact, – decrease, – – significant decrease

Investment Costs

Sustainment Costs
Scaling Agile for Large Programs

- Requires sound engineering discipline
  - Integration of multiple smaller efforts increases complexity
- Robust Enterprise Architecture required
  - Provides framework for individual efforts to map to
- System Performance
  - Designs and testing on system-of-systems and individual component

- Strategies, Backlogs, Roadmaps
  - Define clear program structure that defines mission and business environments
- Cross-Team Integration
  - Frequent collaboration across development teams to track progress, issues, and solutions

See the multiple scaling techniques at https://aida.mitre.org/agile/scaling-agile/
Testing in an Agile Environment

- Testing is continuous and heavily automated
  - As developers check-in code
  - Nightly integration testing
  - Testers are embedded in development team

- Test documentation key to a minimum

- Test Driven Development
  - Define test cases upfront
  - Develop software until passes tests

- Active collaboration with product owner

Agile Shortens Testing Timelines – Early ID of SW Issues
### Barriers and Enablers for Agile Adoption

<table>
<thead>
<tr>
<th>Category</th>
<th>Barriers</th>
<th>Enablers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Culture</strong></td>
<td>Change = Risk</td>
<td>Small empowered teams</td>
</tr>
<tr>
<td></td>
<td>Heavy Oversight</td>
<td>Delegated decisions</td>
</tr>
<tr>
<td></td>
<td>Resistance to Tailoring</td>
<td>Review SW, not docs</td>
</tr>
<tr>
<td><strong>Processes/Policies</strong></td>
<td>Long Timelines</td>
<td>Capstone contracts</td>
</tr>
<tr>
<td></td>
<td>Define requirements upfront</td>
<td>Requirements backlogs</td>
</tr>
<tr>
<td></td>
<td>Costly contract mods</td>
<td>SW Dev as a Service</td>
</tr>
<tr>
<td><strong>User Involvement</strong></td>
<td>Few end-users available</td>
<td>High bandwidth comms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Demo interim sprints</td>
</tr>
<tr>
<td><strong>Program Structure</strong></td>
<td>Fixed Scope/Requirements</td>
<td>Small iterative releases</td>
</tr>
<tr>
<td></td>
<td>APB/EVM Management</td>
<td>Tailored processes</td>
</tr>
<tr>
<td></td>
<td>Limited ent arch, standards</td>
<td>Ent arch guides dev</td>
</tr>
<tr>
<td><strong>Experience</strong></td>
<td>Limited experience in government and industry</td>
<td>Train gov’t/ctr teams</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use Agile coaches</td>
</tr>
</tbody>
</table>
Enable Agile Adoption for Enterprise/Program

- Clearly define WHY go to Agile
  - What operational, acquisition, development factors are driving Agile?

- Provide visible leadership for Agile
  - Shape culture, champion new methods, risk tolerance to adopt Agile

- Tailor environment to enable Agile success
  - Redesign core processes, reviews, docs for small frequent releases
  - Get stakeholders from each functional area to shape new environment
  - Identify key metrics to track, reward early success and failures

- Identify target software intensive programs/areas to pilot Agile
  - Provide pilots a quality team and maximum flexibility

- Gov’t and Contractors go through Agile training together
  - Bring in Agile SMEs as coaches to guide new roles and processes
BACKUP SLIDES
**Agile Manifesto – Values and Principles**

<table>
<thead>
<tr>
<th>Value and Principle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individuals and interactions over processes and tools</td>
</tr>
<tr>
<td>Working software over comprehensive documentation</td>
</tr>
<tr>
<td>Customer collaboration over contract negotiation</td>
</tr>
<tr>
<td>Responding to change over following a plan</td>
</tr>
</tbody>
</table>

1. Continuous delivery of valuable software
2. Welcome changing requirements
3. Deliver working software in weeks/months
4. Work together daily
5. Build projects around motivated individuals
6. Face-to-face conversation
7. Working software is the measure of progress
8. Promote sustainable development
9. Good design enhances agility
10. Simplicity is essential
11. Self-organizing teams
12. Reflect on how to become more effective

[http://agilemanifesto.org](http://agilemanifesto.org)
1. Start with Agile guidance and an Agile adoption strategy
2. Enhance migration to Agile concepts using Agile terms, such as user stories (used to convey requirements), and Agile examples, such as demonstrating how to write a user story
3. Continuously improve Agile adoption at both the project level and organization level
4. Identify, address impediments at the organization and project levels
5. Obtain stakeholder/customer feedback frequently
6. Empower small, cross-functional teams
7. Include requirements related to security and progress monitoring in your queue of unfinished work (the backlog)
8. Gain trust by demonstrating value at the end of each iteration
9. Track progress using tools and metrics
10. Track progress daily and visibly
GAO: 14 Challenges to Agile Adoption

1. Teams had difficulty collaborating closely
2. Procurement practices may not support Agile projects
3. Teams had difficulty transitioning to self-directed work
4. Customers did not trust iterative solutions
5. Staff had difficulty committing to more timely and frequent input
6. Teams had difficulty managing iterative requirements
7. Agencies had trouble committing staff
8. Compliance reviews difficult to execute within an iteration timeframe
9. Timely adoption of new tools was difficult
10. Federal reporting practices do not align with Agile
11. Technical environments were difficult to establish and maintain
12. Traditional artifact reviews do not align with Agile
13. Agile guidance was not clear
14. Traditional status tracking does not align with Agile
## Alternate Agile Structure: 12 Month Release

<table>
<thead>
<tr>
<th>JAN</th>
<th>FEB</th>
<th>MAR</th>
<th>APR</th>
<th>MAY</th>
<th>JUN</th>
<th>JUL</th>
<th>AUG</th>
<th>SEP</th>
<th>OCT</th>
<th>NOV</th>
<th>DEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sprint 1</td>
<td>Sprint 2</td>
<td>Sprint 3</td>
<td>Sprint 4</td>
<td>Sprint 5</td>
<td>Sprint 6</td>
<td>Sprint 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **6-Week Sprints**
  - Continual development, integration, and testing
  - Regular demonstration of capabilities to users

- **Gov’t Dev Test, Op Test, C&A, Users involved early and often**
  - Minimizes work at the end of the release

### Table: Sprint Details

<table>
<thead>
<tr>
<th>Jan</th>
<th>Feb</th>
<th>Mar</th>
<th>Apr</th>
<th>May</th>
<th>Jun</th>
<th>Jul</th>
<th>Aug</th>
<th>Sep</th>
<th>Oct</th>
<th>Nov</th>
<th>Dec</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sprint 1</td>
<td>Sprint 2</td>
<td>Sprint 3</td>
<td>Sprint 4</td>
<td>Sprint 5</td>
<td>Sprint 6</td>
<td>Sprint 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Government Testing, Operational Assessments**
- **Certification and Accreditation / Cybersecurity / ATO Activities**

### Release Length Based on Program, Ops, and Technical Risk
Scaled Agile Framework (SAFe)  

http://www.scaledagileframework.com/
Storyboards and Mockups Help Team Visualize the System and Features

**Storyboards**

Narrative visual depictions set in time to describe system use

**Mockups**

Visual depictions of the feature of the system
Commercial firms use in-house developers
- Government requires contracted support

Contracting Challenges
- Complex laws and regulations
- Long contracting timelines
- Costly change requests
- Defined requirements to select contractor

Design Contract Strategies to Support Short Delivery Timelines
How is Contracting For Agile Different?

- Traditional IT acquisition programs contract for completion-based end-product capability based on defined requirements
  - Requirements locked; changes handled by contract modifications

- Agile development achieved by acquiring a development team delivering labor hours vs. a defined end-product
  - Services contract provides flexibility to change release requirements continuously
  - Designed to support short development and delivery timelines and changes
  - Gov can issue orders for each release based on requirements captured in product backlog

Design Contract Strategies to Support Short Delivery Timelines
Contracting Officer = Agile Business Partner

PMs must partner with COs early to develop strategies

- Streamline processes
- Incentivize contractor
- Responsive to changes

CO is key linchpin to a successful Gov’t – contractor partnership

- Active collaboration
- Dedicated support ideal
- Manage performance

"This is all about asking the program manager and the contracting officer to take a higher workload and more risk in exchange for the greater good."

- John Inman, Contracting Officer for US CIS
Potential Agile Team Construct 2
Potential Team Construct 3
Requirements in Context

**Contract Requirements**
Tasks and activities that describe what the contractor needs to provide under a contract ("6 FTE software development support")

**Agile Requirements**
Functionality or capabilities often expressed in user stories and managed in a product backlog ("Content search capability")

Understand the Difference To Avoid Miscommunication
Consider a PEO, portfolio, or enterprise-level contract vehicle

- Streamlined contracting processes result in faster awards, deliveries
- Standardized, effective, and efficient contract management