Systems Engineering in Large-scale Agile Software Development

Will Menner
Johns Hopkins University
Will.Menner@jhuapl.edu
240.228.5354
My experience: attempts to find a home for agility within systems engineering constructs have largely come up empty.

Instead of starting with a systems engineering model and attempting to find room for agility, start with an agile model and figure out where systems engineering can be of greatest utility.
Motivation

- 2014: NDIA Acquisition Reform Letter - reduced regulatory burden, better-educated acquisition workforce, improved oversight
- 2014: USD(AT&L), Performance of Defense Acquisition System - cycle time is increasing
- 2013: NDIA/ADAPT report: Better Buying Power in Software Acquisition
- 2011: UK Institute for Gov’t report on “fixing the flaws” in gov’t IT
- 2010: HASC report: “the acquisition system is particularly poorly designed for the acquisition of information technology”
- 2009: DSB report: AoA-to-IOC averaging 91 months
- 2008: GAO reports: DoD acquisition programs averaging 21-month delay

The underlying cause of acquisition problems may be largely due to our attempts to control uncertainty. Acquisition reform may need to embrace uncertainty and explore how we can develop methods and systems to better manage it.
The Product Owner Team

- Coordinate with external projects and systems
- Help write and prioritize user stories
- Manage Product Backlog(s)
- Identify Iteration Backlogs
- Devise alignment mechanisms
- Allocate user stories to teams
- Help define and execute acceptance tests
- Develop and maintain metrics
- Coordinate between implementation, architecture, and research teams
- Ensure teams have the technical resources necessary to be successful
- Conduct or facilitate retrospectives to ensure continual improvement
- Create and conduct education and training sessions for stakeholders
- Facilitate the collection of feedback from end-users
- Synthesize feedback into new user stories

Activities for Systems Engineers on Large-scale Agile Teams
Activity-based Decision Framework

An Example based on Data Analysis Workflow for a Generic “Mission”

1. Need Discovery (Reqmt Mining, too)
   - Morph Needs Into Reqmts
   - “Granularize”
   - Prioritize
   - Estimate
   - Allocate

2. Ongoing Operations
   - Mission Sources & Sensors
   - Data Collection
   - Data Preparation
   - Analysis / Exploitation
   - Provide I&W, Alerts, Tipping, Cuing, SA, ...
   - Mission Planning
   - Mission Execution

3. Validation
   - Mission Outcomes / Effects
   - Adaptations
   - Deployment
   - Validation

4. Mission Capabilities
   - Applications, Tools, Analytics, ...

5. Infrastructure Architecture
   - Comm’s, IA, Security, ...

6. Mission Application Architecture
   - Capability Delivery
   - Reuse

7. Tooling
   - Leverage:
     - App Store
     - Service Catalog
     - Architectural Models
     - External Providers

8. Plan & Execute
   - “Spins” & Iterations
   - Team Set A
   - Team Set B
   - Team Set C
   - Team Set D
   - Team Set E
   - Team Set F
   - Team Set G
   - Team Set H
   - Type C Scrum
   - Innovation Cell

9. Agile Development
   - Stakeholder Involvement
   - “Spins” (Strategic)
   - Iterations (Tactical)
Functional Description of “Mission” used to Organize and Allocate Requirements to Teams

An Example based on Data Analysis Workflow

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Databases / Repositories</td>
<td>Geo-Registration</td>
<td>Signature Dev &amp; Recognition</td>
<td>Communication with external stakeholders Planning</td>
<td>Measures of Performance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Metadata Harmonization</td>
<td>Feature Extraction</td>
<td>Multi-source, Multi-sensor Data Fusion or Correlation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Planning &amp; Tasking</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Specific Projects, Components, and Applications

... Tiger Teams, Special Interest Groups, Communities of Practice, …

Scenarios, End-to-end Test Cases, Threads

<table>
<thead>
<tr>
<th>Team Set A</th>
<th>Team Set B</th>
<th>Team Set C</th>
<th>Team Set D</th>
<th>Team Set E</th>
<th>Team Set F</th>
<th>Team Set G</th>
<th>Team Set H</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integration Team 1</td>
<td>Integration Team 2</td>
<td>Integration Team 3</td>
<td>Integration Team 4</td>
<td>Integration Team 5</td>
<td>Integration Team 6</td>
<td>Integration Team 7</td>
<td></td>
</tr>
</tbody>
</table>

Systems Engineering, Coordination, Collaboration, Deployment, Validation, …
“Any organization that designs a system (defined more broadly here than just information systems) will inevitably produce a design whose structure is a copy of the organization's communication structure.” - Melvin E. Conway, 1968, Datamation

- Why let your existing organization structure drive the way you tackle problems? (Especially if it’s not a good match.)
- Understand the problem space; then reshape your organization to efficiently attack it.
Summary

- Agile principles and practices extend well to large-scale development
- Product Backlogs become multi-dimensional
  - Architectural issues get extra attention
  - Research and technology development get extra attention
- Product Owners have a team
  - Systems engineers may help in many ways, including ...
  - ... spreading the vision, validating the product, and promoting reuse strategies
- Organizations need to flex their structures to match what is best for operations (not the other way around)
  - Allows better alignment and allocation of work to teams
  - Allows flexibility through the use of Type C Scrum
The End
Scaled Agile Framework (SAFe)
DevOps

- DevOps involves the efficient integration of organizational functions to streamline the path between concept and cash.
- Sustained innovation is one of the few ways left to gain competitive business advantage.
- Organizations must quickly transform ideas into marketable and revenue-producing products.
- Each functional unit must strive to no longer be a rate-limiting impediment on the product’s path from development to delivery.

Agility needs to permeate the entire organization!
Disciplined Agile Delivery