Strategies for Streamlining Enterprise Architecture in the Age of Agile

John Mallinger & Paul Newell

22nd Annual NDIA SE Symposium

10/24/2019

Right Sizing Architecture for Agile

- The Case for Architecture on Agile Projects
- Acquisition Strategies – Contracting for Agile Success
- Aggressive Tailoring – Driving Architecture Value
- Changing at the Speed of Agile – Planning for Evolution

The agile challenge…

- The Agile Manifesto challenges software design and architecture to add value
  - Prioritizes collaboration and responding to change
  - Deemphasizes documentation and processes
- Architecture activities need to add clear and recognizable return on effort invested
  - Team strategy may leverage emergent design
- Agile development strives for a balanced approach to documentation
  - Avoid creating documentation shelfware

The Case for Defining Architecture

- Eliminating all design and architecture efforts drives substantial risk into agile development
  - Emergent designs fail to meet undefined quality attributes
  - Poorly aligned teams build incompatible interfaces
  - Incomplete system decomposition fails to satisfy user needs
- Defined architecture drives greater efficiency in agile
  - Architecture supports reuse, commonality, and adoption of common patterns
  - Coordinated approach helps agile scale to larger projects and teams

Agile Done Wrong

- Big Design Up Front – developing 600+ requirement System Spec
  - Agile principles undermined by defining large fixed and immutable MVP
- Applying Mil-Std-1521B to Agile Acquisitions
  - Artificially forces Big Design Up Front
  - Drives accelerated design to satisfy waterfall events
  - Limits tailoring of design and architecture artifacts
- Using out-of-the-box Earned Value to manage Agile projects
  - Constrains ability to re-plan and react to changes and discovers during sprints

Agile in the Enterprise

- Layers of Architecture
  - Big Design Up Front
  - Iteration-Driven Architecture
  - Architecture at the Sprint Level
- Hybrid Approach managing backlog
  - Small Design Up Front
  - Enterprise Architecture within Iterations
  - Component & Service Architecture within sprints
- Make architecture inclusive and drive participation from all stakeholders
  - Agile seeks inputs and buy-in from everyone
  - Team needs to own and leverage architecture to drive value

Agile Acquisition Strategies

Defense Innovation Board Report:

Speed and cycle time are the most important metrics for software.

Faster reduces risk because it demands focus on the critical functionality rather than over-specification or bloated requirements.

Ten Most Important Things to Do (D3): Shift from the use of rigid lists of requirements … to a list of desired features and required interfaces / characteristics to avoid requirements creep, overly ambitious requirements, and program delays.

DIB SWAP Study

TechFAR Handbook:

- Use Product Vision Statement to scope high-level objectives for Agile Software acquisition
- Structure requirements as a Statement of Objectives
- Product Owner defines / revises high level requirements as part of scrum-based agile process

TechFAR Handbook

Digital Services Playbook:

Well-defined contract can facilitate good development practices like:

- Conducting research and prototyping phase
- Refining product requirements as service is built
- Contract gives government delivery team enough flexibility to adjust feature prioritization and delivery schedule as project evolves

USDS Playbook

Driving Architecture Value

- Extreme tailoring to maximize design & architecture value
  - Focus on high value critical design tasks
  - Tailor out low value or redundant tasks and artifacts
- How extreme?
  - Everything is on the table
  - No artifact or task is sacred

Agile Architecture Objectives

- Capture Up Front Design and High Level Approach
- Support tactical decisions within Sprints
- Architecture Change Management and Responding to Update Requests

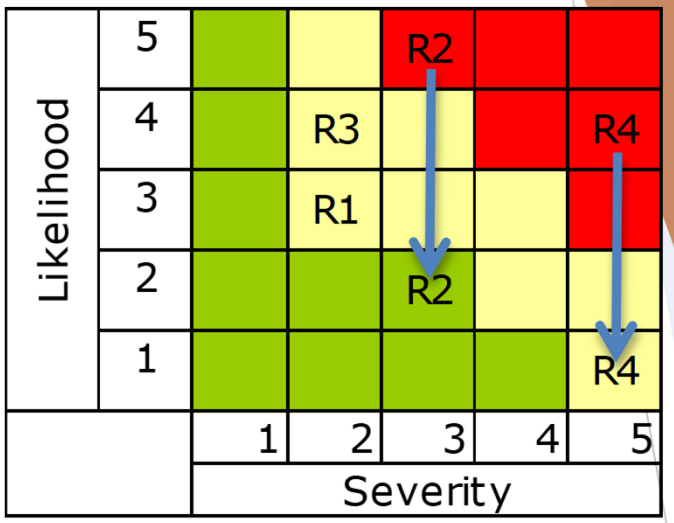
Quantifying Architecture Value

- Use an objective framework to assess value of architecture tasks
- Options
  - Agile-based planning
    - Points based voting or priority poker to rank
  - Risk Driven
    - Design and Architecture mitigate risk of leveraging emergent and undefined development strategies
  - Weighted Shortest Job First (WSJF) strategy
    - Don Reinertsen – The Principles of Product Development Flow
    - Quantify cost of delay from missing design or architecture tasks

Risk-Based Prioritization

Risk assessment and mitigation strategy drives

- Prioritize architecture tasks to maximize risk mitigation
- Target moderate or high design risks
- Can approach with generic architecture tasks or specific design challenges



#	Architecture Task	Risk	Initial Score	Mitigated Score
R1	Develop CONOPS		6	
R2	Define External Interfaces	Late definition drives incompatibility	15	6
R3	Create User Storyboards	Displays do not meet user needs	8	
R4	Identify & Map Quality Attributes	Unable to meet reliability, maintainability, or scalability	20	5

Weighted Shortest Job First

- Method for prioritizing work based on likely impact
- Team assesses Cost of Delay and Job Size for each task
  - Cost of Delay encompasses value of artifact in driving design or implementation and alignment
  - Job size is relative complexity of a task compared to other tasks

#	Architecture Task	Cost of Delay (CoD)	Job Size / Duration	WSJF	Dependencies?
1	Develop CONOPS	8	5	1.6	
2	Define External Interfaces	12	12	1	
3	Create User Storyboards	12	8	1.5	Task #1
4	Identify & Map Quality Attributes	20	1	20	

Donald G Reinersten: The Principles of Product Development Flow: Second Generation Lean Product Development

Implementing Agile Architecture

- Results of architecture value assessment drive implementation
- Small Design Up Front
  - Limited big picture design to communicate high level objectives
  - Define and select quality attributes to emphasize in architecture
- Plan for Change and Evolution of Design
  - Agile sprints and iterations will not follow prescribed order
  - Consider use of TOGAF Transition Architecture to track evolution

https://pubs.opengroup.org/architecture/togaf9-doc/arch/chap13.html

Change at the speed of agile

- Agile Success starts with the contract
- Limit big design up-front
- Minimize architecture artifacts to those absolutely necessary
- Embed most design and architecture within iterations or springs
- Get entire team engaged in architecture