Agile Product Line Architecture

SATURN 2011
May 2011

Paul Clarke
Chief Architect

Agile Product Line Architecture

Airborne ISR Product Line
Context
- Leader in design, development, and integration of advanced AISR system architectures
- Complete lifecycle through field support and upgrades

Our Achievements from Agile Architecture
- Lower costs
- Faster time-to-market
- More capability
- More flexibility and adaptability
- Higher performance
- Smaller size, weight, and power (SWAP)

Agile Product Line Architecture
- Adapt to change
  - Customer needs
  - Market opportunities
  - Competition
  - Technology evolution
- Use a systematic process
- Create architecture roadmap with Business Case
- Follow through

Agile Development in a Product Line
- Scrum sprints for iterative development
  - Focus is on managing implementation details
  - Flexibly adapt to needs of multiple product teams
- Retain Architecture Practices
Life Before the Product Line

- Matrix Organization
  - Engineering supplied talent to projects
- A project delivered a system to a customer’s specification

Product Line: A Strategy of Planned Reuse Across Products

Core Asset Identification

Core assets are software or hardware components developed for systematic reuse across the product line. Core assets share a common architecture.
Managing Variation in a Product Line

- **Requirements**
  - Additional requirements or differences in performance levels

- **Design**
  - New features are “configurable” so the software can run in either a “basic” mode or in extended modes with the new features available

- **Change Management**
  - Requirements
  - Software code and unit test
  - Design documents
  - Test documents
  - Re-test

Airborne ISR Product Line Features

- A variety of different types of hardware components (circuit cards or larger)
  - Including custom hardware components

- Most products require some customization

- Products are highly customizable and support a broad variety of requirements across customers
  - This can run the gamut from software configuration changes to custom hardware
  - Scale: Size, Weight, and Power (SWAP)
  - Aircraft variations
  - Capabilities and features
Agile Product Line Architecture

Agile Architecture: Strategic Architecture Roadmap

- Objective: An Action-Based Plan for Moving Forward
- Guide Product Line Improvements
  - What improvements and features significantly generate revenue
  - When do improvements and features need to be available
  - How will they be implemented
- Validate the Business Case
- Selectively Target R&D Investments
- Align Products and Services with the Business Long-Range Strategic Plan

The Strategic Technology Roadmap combines multiple views into a Time-phased Plan to effectively fill capability gaps
Strategic Architecture Roadmap Methodology

1. Track Business Environment Changes: Gap Analysis
2. Develop Business Case
3. Evaluate Technology Trends
4. Re-evaluate Entire Architecture
5. Define End-State Architecture
6. Create Architecture Roadmap to Realize the Architecture
7. Periodically Reassess the Roadmap and Adjust it as Conditions Change

Business Case: Gap Analysis

**Market Evolution**
- Faster response
- More customized
- More integrated functions

**Payload Size**
- Large
- Medium
- Small

Size corresponds to size of opportunity
Business Case: Gap Analysis (2)

- New Capabilities
  - Require greater processing and I/O performance

- Processing Performance
  - Low-power processor technology transition

- I/O Interconnects
  - High-speed interconnects
Re-Evaluate Entire Architecture

- Re-Evaluate Customer Needs
  - Identify gaps in “Missions” and “Threats” in our domain
- Re-Evaluate Market Opportunities and Competitive landscape
  - Link the gaps to market opportunities
- Re-Evaluate Technology
  - Incorporate new technologies that address a gap

- Re-evaluate portions of the architecture
  - e.g. I/O fabrics, middleware, available interface standards, processors, operating systems, chassis form factors, bottlenecks, ...
- Re-evaluate overall architecture

Define End-State Architecture

What to change

- Refactor Architectural Boundaries
  - For greater flexibility and adaptability
    - Adaptable to aircraft
    - Adaptable to missions & threats
    - Adaptable to new hardware
    - Adaptable to new algorithms
  - Scalable
  - Easy to integrate new capabilities
  - Create “standard” products
- Upgrade Technology
  - Processing
  - I/O Fabrics
- Reduce Size, Weight, and Power (SWAP)
  - Compress hardware features

Simplified End-State Architecture
Strategies Used to Change the Architecture

• **Simplify**
  - Remove features rarely needed
  - Exploit newer standards
  - Replace proprietary implementations with purchased technologies

• **Isolate**
  - Reduce internal interfaces
    - A simple interface with a complex implementation was more cost-effective than a complex interface with a simple implementation
  - Add abstraction layers to hide complexity
    - Hide hardware complexity and variation

• **Downsize**
  - Re-package for smaller-scale problems
  - Compress hardware functions opportunistically

• **Unbundle**
  - Split monolithic functions into smaller modules
  - Can independently tailor performance of each module
  - Can choose best of breed for each module

• **Bundle**
  - Create larger-grained modules as standard reusable components

• **Find New Modularity**
  - Create modular hardware components for reuse and flexibility

• **Retain backwards compatibility**

Create the Strategic Architecture Roadmap to Realize the Architecture

**Define Transitions in the Roadmap**

- Identify time sequence
- Identify funding sources

**Align Roadmap with Business Pursuits**

- Place Roadmap on a schedule
- Align Roadmap timeline with key opportunity milestones
Changing Missions, Threats, Markets, and Technologies Drive Dramatic Improvements in Capability, Adaptability, Cost, and Time to Market

Agile Development in a Product Line

A Short Story…
We Stumble into Agile…

• We were entering system I&T for a project…

• We had an extreme problem:
  – A major component was not going to meet performance and availability requirements
  – We were late in the game and we need a software rewrite

• We decided to use **extreme** programming to re-architect and re-implement the component
  – We paired a domain expert with a OO development expert

• The pair finished in time!
  – We met the deadline and passed factory acceptance test

We Try Scrum for the Product Line…

• Based on that experience, we decided to explore Scrum for the product line

• We obtained training from Rally Software and read books

• We took over a conference room and made it our Scrum room

• The team was excited and motivated

• We had great planning sessions
  – Retrospectives (start/stop/continue)
  – Engineer availability
  – Story development/sizing using poker
  – Task creation/development
  – Rush the wall – self organizing

• Agile and Architecture
  – For new core assets we created design stories
  – Once the architecture was complete we followed the process to develop the detailed design and implementation
We Make Adjustments Along the Way…

- We adjusted sprint duration to the type of software
  - Longer sprints were more effective for embedded development

- We added more products
  - We created a scrum for each product
  - Established repeatable sprint activities

- Added Problem Tracking Reports (PTRs) to the sprints
  - Create a story for PTR’s and a task for each PTR

- When we couldn’t finish a story in a sprint:
  - Create a new story for the next sprint with whatever remaining tasks are left
  - Estimate hours to complete

We Surrmount Challenges…

- Buy-in from engineers
- Buy-in from project managers
- Transitioning leadership and management roles
  - Follow through/action by management
  - Instilling discipline
- Culture change and fear of change
- Scaling for large team implementation – scrum of scrums
- Meeting CMMI requirements – how to do this in an agile environment
  - Architectural design and agile
  - PTRs and agile

- Integrating with traditional schedules
  - We added stories to traditional program manager (PM)schedules via unique IDs managed in the PM and sprint tools
We Achieve Agile Nirvana (Benefits)…

- In addition to frequent tangible milestones, we have improved our:
  - Teamwork
  - Communication
  - Technology transfer
  - Planning and re-planning
    - Release planning
    - Sprint planning
    - Mid-sprint reviews
  - Development estimates by engineering staff
  - Ability to accommodate change

Scrum!

Summary

- Architecture changes are driven by
  - Gap analysis and business case

- Agile Architecture Evolution is achieved by
  - Following a strategic roadmap process
  - Using agile development to facilitate execution

Benefits Achieved

- New capabilities
- Flexibility and adaptability
- Reduced costs
- Reduced time to market
- Reduced SWAP

Acknowledgements:
Portions of this presentation based in part on material provided by Ragan Wilkinson and John Dydiw of Northrop Grumman