Agile and Enterprise Architecture – Synergy in Practice

June 19, 2014

Copyright © 2014 by cPrime and David Frankel Consulting
David Frankel – Enterprise Architect & Technical Strategist

• **David S. Frankel** has over 30 years of experience as a programmer, architect, and technical strategist. He is recognized as a pioneer and international authority on the subject of model-driven systems and semantic information modeling.

• He has published two books and dozens of trade press articles, and has co-authored a number of industry standards including UML, ISO 20022, BIAN.org, and the XBRL Abstract Model.

• The IT domains in which he has recent experience are: Enterprise architecture, service-oriented architecture, model-driven systems, semantic information modeling, UML, XBRL, software product lines, business vocabularies, and domain-specific languages.

• The business domains in which he has recent experience are: ERP financials, banking, electronic payments, and financial reporting.
At presentation completion you will:

- appreciate the need for enterprise-wide coordination to enable Agile development practices to scale
- understand how enterprise architecture can complement and support agile development practices
- have benefited from some concrete examples
Part 1 – SAFe Approach to Enterprise Architecture

• SAFe basics and principles of Agile architecture

Part 2 – Concrete Examples

• Illustrating the use of Enterprise Architecture in a manner that respects and empowers Agile development teams
Part 1 – SAFe and Enterprise Architecture

See the Scaled Agile Framework (SAFe) “big picture” at:

http://www.scaledagileframework.com
How Does System Architecture Evolve?

Architecture can emerge but some of it needs to be intentional…

• Emergent Architecture
  • “The best architecture, requirements and designs emerge from self-organizing teams” – Agile Manifesto
  • Fast collaboration between Teams, PO, Users produces robust reliable systems that meet their needs
  • Don’t need someone outside to do the design

• Intentional Architecture
  • Individual teams, products and programs may not have visibility to see how enterprise system needs to evolve
  • Systems can grow fragmented over time
  • Architectural vision, some amount of architectural governance and planning is necessary

The challenge is to scale without breaking the spirit of Agile
Roles

• **Enterprise Architect**
  • Operates *across* Programs
  • Helps ensure development strategies and technologies are aligned with business needs as they evolve
  • Maintains ongoing ties with development teams

• **System Architect**
  • Operates at the Program level
  • Maintains the vision of what users need and an understanding of the enterprise architecture as they both evolve
  • Works in daily collaboration with the teams focusing on design decisions
• Enterprises need infrastructure, development tools and platforms (runway metaphor), to support feature implementation (plane metaphor) by the teams without excessive redesign/refactoring.
  • Teams “land” features on the runway.

• Triggers: M&A, technology change, performance issues, costs of duplicate effort
SAFe Principles of Agile Architecture

1. Design emerges. Architecture is a collaboration.
2. The bigger the system, the longer the runway.
3. Build the simplest architecture that can possibly work.
4. When in doubt, code it or model it out.
5. They build it, they test it.
6. There is no monopoly on innovation.
7. Implement Architectural flow.
Part 2 – Concrete Examples – Agenda

• **New Technology: Investment Themes & Architecture**
  - In-Depth Example: Mobile Computing
  - Quick Look at Additional New Technologies

• **Architecting for Agility, At Scale**
  - Separation of Concerns
  - SOA
  - Integration
  - Security
  - End-to-End Traceability
Part 2 – Concrete Examples – Agenda

• New Technology: Investment Themes & Architecture
  • In-Depth Example: Mobile Computing
  • Quick Look at Additional New Technologies

• Architecting for Agility, At Scale
  • Separation of Concerns
  • SOA
  • Integration
  • Security
  • End-to-End Traceability
General Principles for Analyzing New Technologies

• **Investment theme perspective**
  - Examines business opportunities, drives Business Epics
  - Avoids a purely technology-driven approach
  - **The Technology 1.0 Syndrome**¹
    - First wave adoption of new technology
      - Enable businesses to do the same old things more efficiently or in a different form
    - Later waves
      - More fundamental re-thinking

• **Architecture perspective**
  - Policies: In collaboration with development Teams
  - Architectural Runway: Infrastructure needed to support project Teams in using the new technology
Part 2 – Concrete Examples – Agenda

- **New Technology: Investment Themes & Architecture**
  - In-Depth Example: Mobile Computing
  - Quick Look at Additional New Technologies

- **Architecting for Agility, At Scale**
  - Separation of Concerns
  - SOA
  - Integration
  - Security
  - End-to-End Traceability
Mobile Computing
Investment Themes

• First wave
  • Focus on how to enable existing applications to run on mobile devices

• Second wave
  • Focus on new business opportunities
    ▫ Extending the reach of enterprise applications
    ▫ Re-thinking business practices
Mobile: Second-Wave Investment Themes

Extending the reach of enterprise applications:

- Apps for the manufacturing shop floor and field operations
- New apps connect to traditional back-office applications
- Added value from increased efficiencies
- More personnel touch the applications: increases per customer wallet share
Mobile: Second-Wave Investment Themes
Re-Thinking Auto Rental

- Customer app unlocks car, registers pick-up with back office
- Obviates need for attended pick-up and drop-off stations
- Much lower overhead
Mobile: Risks

• **Dependency Management**
  - Mobile apps connected to back office applications → greater complexity

• **Multiple Operating Systems**
  - Complicates decisions as to how to proceed
Dependancy Management Complexity

On-Device Mobile Components

Traditional Back-Office Components

On-Server Mobile Components

Cloud-Based Back-Office Components
Multiple Operating Systems & Devices

- iOS
  - iPad
  - iPhone
- Android
  - Samsung
  - Motorola
  - Etc.
- Windows Mobile
  - Surface Tablet
  - Phones
Mobile: Architecture
Dependency Management

• **Must be addressed at the Portfolio level**
  • Can’t manage successfully if there is no coordination across Teams and Programs
    ▫ Coordination can support variation, subject to constraints

• **Proper architecture support empowers Teams**
  • Relieves worry about hidden dependencies
  • Lifts burden of manual bookkeeping
  • Frees developers to concentrate on what they do best
Dependency Management: Policies

- One approach: Disallow certain kinds of dependencies
  - Such as
    - Back-office component to mobile component
    - Cyclic dependencies
  - Or allow, but flags such dependencies

- Develop policies collaboratively with development Teams
Mobile: Architecture

Problematic Dependencies

On-Device Mobile Components

On-Server Mobile Components

Traditional Back-Office Components

Cloud-Based Back-Office Components

Copyright © 2014 by cPrime and David Frankel Consulting
Mobile: Architecture
Dependency Management Architectural Runway

• **Tools and/or API-level services**
  • For managing dependencies and repositories for development artifacts
    ▫ Should handle versioning, and rewinding or forwarding versions from/to any node of the version tree
    ▫ A dependency system should be configurable via policies
      – such as lock-down on a version or track the latest version
    ▫ Should detect or prevent cyclic dependencies
    ▫ Should support federating repositories
      – in repository hierarchies, as needed
      – Teams have control over their local repositories

• **Collaborate with development Teams!**
Dependency Management Tooling
Federated Repositories

Team Configuration
- Mobile
- Loan Sales
- Team Repository

Program Configuration
- Loan Sales
- Program Repository

Team Configuration
- Back Office
- Loan Sales
- Team Repository
Also requires Portfolio-level coordination
  • Regarding tools and techniques for separating platform-independent
    and platform-specific artifacts
    ▫ Unconstrained → serious difficulties in integration

Policy options
  • Separate code lines for each platform
  • One code line with cross-platform compilers
  • Hybrid

Architecture Runway
  • Development environments
  • API-level services for common application requirements
    ▫ Functional and non-functional requirements
Multiple Operating Systems: Example 1

iOS Configuration
- iOS (Xcode)
  - Mobile Team iOS Repository
Neutral Configuration
- Neutral
  - Mobile Team Neutral Repository
Android Configuration
- Android
  - Mobile Team Android Repository

Dependency/Build Tools
- Central Configuration
Mobile: Architecture

Multiple Operating Systems: Example 2

- iOS (Xcode)
- Neutral
- Android

Dependency/Build Tools

Mobile Team Repository

- Mobile Team iOS Artifacts
- Mobile Team Neutral Artifacts
- Mobile Team iOS Android

Mobile Team Configuration
Part 2 – Concrete Examples – Agenda

• New Technology: Investment Themes & Architecture
  • In-Depth Example: Mobile Computing
  • Quick Look at Additional New Technologies

• Architecting for Agility, At Scale
  • Separation of Concerns
  • SOA
  • Integration
  • Security
  • End-to-End Traceability
Other New Technologies Requiring a Portfolio-Level Perspective

• **In-Memory Computing**
  - Can radically simplify database architecture
  - Fast analytics opens opportunities to fine tune liquidity management to unprecedented degree
  - From enterprise-scale to internet-scale

• **Cloud Computing and Virtualization**
  - Potential to simplify enterprise IT…
  - …along with risks to factor in

• **Sensors**
  - Increase supply chain transparency and efficiency
  - New levels of customer service for appliances

• **Social Networking & Collaborative Computing**
  - Closed-loop collaboration facilities in enterprise applications
  - Monitor and influence customer base via public social networks

• **Big Data**
  - New technology is causing massive growth of data volume
  - Much of the new data is beyond the realm of relational database
Part 2 – Concrete Examples – Agenda

- New Technology: Investment Themes & Architecture
  - In-Depth Example: Mobile Computing
  - Quick Look at Additional New Technologies

- Architecting for Agility, At Scale
  - Separation of Concerns
  - SOA
  - Integration
  - Security
  - End-to-End Traceability
More Architecting for Agility

• **Separation of Concerns**
  • An overriding theme: Avoid hardwiring things together
  • Separate functional behavior from technology specifics
    - E.g. isolate code that is hardwired to a particular service bus technology

• **“Second-Generation” SOA**
  • Agility from proper use of SOA to manage business processes
  • Requires right level of granularity for services

• **Integration**
  • Data integration is a major enterprise IT bottleneck
  • By definition, dealing with this only at Team level is problematic

• **Security**
  • Inconsistency of security mechanisms across the enterprise is a known security vulnerability

• **End-to-End Traceability and Visibility**
  • Programmers produce less procedural code and more declarative metadata
  • Traceability of metadata across the landscape is powerful
References


• Questions?