An Enterprise Architecture and Model-Driven approach to Service Application Creation

Jon Chard
Principal Consultant
Session 1 – SOA overview
What is ‘Service Oriented Architecture’?

- Architecture built from many individual services
- Evolution of past architectures
- The opposite of a monolithic application
- Common communication platform

Think about a city with many interworking pieces…
SOA – Market Trends

“*The march of SOA into companies is unavoidable at this point. Adoption is accelerating, and SOA is being baked into virtually every commercially available enterprise application. However, SOA adoption is uneven, with some industries and geographies leading and others clearly lagging.*”

Source: Ian Finley, AMR Research 2007
What is a Service?

- Supplies a business need
- loosely-coupled to other services
- Well defined service contract
- Autonomous
- Abstract
- Reusable
- Self-described and discoverable

Legacy assets may be ‘SOA-enabled’ with a SOA compliant wrapper
What makes up the Architecture?

• The complete set of services and their interconnections
  – (With all the supporting information)

• Some common issues
  – What are the interconnections (dependencies)?
  – What will the Architecture look like in the future?
  – How will you control changes to it?
  – How will you find a particular service?
  – How do people get to understand the Architecture?

⇒ The Architecture has to be maintained at an abstract level
  – Simply having a bunch of Web Services which call each other isn’t an Architecture
Why SOA?

• Increase Business Agility
  – Respond to customer needs
  – More flexible IT (plug and play)
  – Bring products/services to market quicker
  – **Flexibility + Efficiency = Innovation**

• Reduce Costs
  – Manage out overlapping or redundant systems
  – Easily scale each application for max efficiency

• Reduce Risk
  – Better Understand IT Capability
  – Assess impact on the business
  – Deliver required capability

• It’s a **BIG** competitive advantage
  – Your users and customers can better meet their requirements and deadlines
Some Common Pitfalls

• Building SOA like a traditional distributed application
• Inability to communicate between teams
• Not creating a transition plan
• Not standardising across your projects
• Not defining the contracts first
• Poor analysis and design
Avoiding the pitfalls of SOA – a lifecycle approach

• Effective SOA is not just about getting the architecture right
• Effective SOA is not just about a correct technical implementation

• Effective SOA requires an integrated, yet flexible approach, including:
  – Accommodating new and evolving business needs
  – Supporting efficiency through effective reuse and optimisation
  – Supporting and enabling effective technology strategies
A Layered Enterprise SOA Approach

Business Motivation
- Business organization
- Business goals

Business Process Layer
- Business processes
- Governance
- BPMN
- Process execution

Services Layer
- Orchestration
- Storyboarding
- Schemas
- UML
- WSDL
- Service registries

Application Layer
- Applications
- Service adapters
- Legacy systems
- Service repositories
- UML

Technology Layer
- Operating systems
- Databases
- Code
- Application platforms
- Technology platforms
- Java
- C#
- .NET
- Tau
- J2EE

Business requirements
User requirements
IT
Enterprise Architecture and Model-Driven SOA
Benefits of Enterprise Architecture-Driven SOA

• A single repository of business SOA information
  – Capture business strategy, processes and information
  – Model the systems and applications they have to support the business needs
  – Capture the relationships between architectural elements

• A basis for communication analysis and understanding
  – Measure and compare the value of different initiatives
  – Report on the service landscape, revealing gaps and overlaps
  – Time-based reporting for ‘what-if?’ and ‘when?’ analysis
  – Evaluate the alignment of services before and after deployment.
Benefits of Model-Driven SOA

• Satisfy business requirements
  – Design applications that directly tie to the business plan
• Tame complexity
  – Visually analyse, design and assemble service architectures
  – Provide a single SOA technical service repository
• Collaboration and understanding
  – Visualize, import and generate WSDL and XSD
• Adapt legacy services
  – Apply round-trip engineering
• Ensure correctness
  – Simulate and test your logic **before** any code is written!!
• Enable reuse
  – Keep designs abstract to extend their life
Enterprise Workflow

- Business requirements
- User requirements
- System requirements

- Develop enterprise architecture
- Analyze & optimize business strategies
- Design services & applications
- Deploy & maintain

Trace and analyze relationships
Add the middleware stuff to this diagram
Implementation Considerations - 1

Implementing a SOA workflow and toolset is a strategic undertaking which will underpin the business over multiple projects and time frames

• Both business architecture and technical service information must be available across projects and across the business

  ⇒ Implementation approach should provide repositories for (1) enterprise architecture and (2) technical services

• Notations will vary across the SOA lifecycle:

  – Business: BPMN, IDEF, BPEL, etc.
  – Technical: UML, WSDL, Code, etc.

  ⇒ Tool support for required notations and definable mappings between the layers is essential
Different Approaches

Model-centric

Business Processes

Requirements

Architecture/Service Model (Platform Independent)

Application Model (Platform Specific)

Application Code

Runtime

development

integration & reuse
Implementation Considerations - 2

• Different workflow approaches require bi-directional modeling interfaces

• Two information flows through the lifecycle:
  – Model elements
  – Traceability – end to end linkage of models and requirements

• Technology independence of solutions:
  – SOA is strategic – technologies change
  – Technology independence is vital for effective reuse
  – Middleware is the interface
Interim Conclusions

• SOA has the potential to deliver business agility with reduced costs and risk
  – If SOA projects are handled correctly

• Effective SOA projects must address the whole landscape from business motivation to technical delivery
  – Enterprise Architecture can deliver benefits in planning and aligning SOA activities to the business
  – Model Driven Development can deliver benefits in collaboration, correctness, consistency and reuse in the technical implementation phase

• Combining Enterprise Architecture and Model Driven Development with flexible workflow support is the key to delivering effective SOA
Thank You!

Questions?
An Enterprise Architecture and Model-Driven approach to Service Application Creation

Jon Chard
Principal Consultant
Session 2
A practical Enterprise Architecture and Model-Driven SOA Workflow
Add the middleware stuff to this diagram
Example tool support
Modeling the Enterprise Architecture

**enterprise blueprint... gain understanding**

- Business Process Diagram
- Application Architecture
- Service Collaboration Diagram
- Network Concept Diagram

**Modeling the Enterprise Architecture**

- Model the key domains and their relationships:
  - Business, Strategy, Applications, Infrastructure, Data
- Relate to the overall service infrastructure and its elements
An example EA for SOA workflow

1. Identify & model business processes
2. Map services to BPMN processes
3. Build Application portfolio
4. Map services to applications
5. Build service model
Service collaboration diagram

- Send Correspondence
  - Bid Notification
  - Place Bid
  - Payment Details
  - Online Credit Card Payment (debit)
  - Close Auction

Auction Bidding (Service Collaboration)
System Architect
Mon Apr 16, 2007 11:56

Comment
Illustrates how services collaborate within a scenario.
SOA Deployment Analysis
optimize implementation environment, prepare for change

• Provide the enterprise view of the SOA project
  – Show how business operates today
  – Explore how new services are going to fit into the organization
  – Perform impact analysis on removal or change of existing services

• Business and IT can see the service plan for any implementation
  – Identify candidates for reuse
  – Perform analysis to determine characteristics of desired future state architectures
SOA Deployment Analysis
visualize the SOA environment, manage change

- Capture and visualize service hierarchies
- Automatically generate service roadmap reports
- Manage delivery
- Assess the impact of changes
- Spot gaps and duplication in plans
Business Analysis & Planning

*business agility ... reduce risk*

- Map business objectives and goals
  - establish scope
- Understand the existing business in terms of:
  - people
  - processes
  - applications
  - information
  - services
- Plan changes in line with objectives and goals
  - roadmaps
- Develop and maintain business service architecture
Linking Enterprise Architecture and Technical Development

Business Processes

UML Use Case Analysis

Modeling information flow

Traceability

UML technical service implementation
EA to UML Modeling Flow
Linking the EA and MDD environments

EA environment

UML2 environment

EA repository
Project Architecture Analysis and Design

*improve accuracy ... increase quality*

- Start from the business needs from System Architect
  - Capture understanding of the overall purpose for development
- Visualize the significant participants and what they need
- Determine what is involved in meeting their needs
- Maintain full traceability
- Simulate and test as a whole
Distributed Application Assembly

*improve functionality ... quality ... reduce complexity*

- Split the Architecture up into manageable components
  - Treat each component as a standalone model
- Simulate and test as a set of connected services
- Check and Test that each component is correct
- Extract documents and reports from the model
  - Can be updated automatically by extracting the information again
- Analyze the model to determine overall impact
Technical Service Development

deliver functionality

- Perform gap analysis to determine missing services
  - Buy or make?
- Extract design data to define service’s architecture
- Add platform details for deployment
- Round-trip engineer deployable software
  - In Java, C# or C++
Encouraging Re-use

*business agility ... fast time to value ... scale as needed*

- Reuse brings major advantages
  - Reduces effort
  - Reduces risk – zero impact on quality
  - Speeds delivery
- Re-use existing information in the IT landscape
  - Import existing Service definitions
  - Import other data (XML/XMI/CSV/Java/C#/*C++/...)
- Re-use within the service Architecture
  - Search for existing services (harvesting)
  - Design new services with re-use in mind
Automating Service Reuse

1 - Locate Web Service

2 – Import and visualize in UML

3 - Auto-generate API and client

4 – Auto-generate executable and orchestrate web-service
Collaboration

manage distributed teams... outsource contractors

• Divide and govern large projects
  – Logically separate project models
  – Develop and deliver specifications or interface
    information (WSDL and/or XSD)
  – Powerful compare/merge

• Speak the same language
  – Common basis for specifying and enforcing compliance
  – Gain acceptance at each stage through shared models

• Keep teams fully accountable
  – Traceability ensures accurate reporting
  – Prove designs against the original requirements
Round Trip Traceability - The Audit Trail

Reduce Risk … Improve/Achieve compliance

business processes

business requirements

service design

user requirements

system requirements
Summary

• Correctly implemented, SOA provides the potential to improve agility whilst lowering risk / costs

• An EA and MDD SOA approach assures that the whole lifecycle is covered, ensuring that business requirements are driving the IT landscape

• Enterprise architecture brings many benefits to SOA planning:
  – Alignment of technical development with business needs
  – Management of the evolving landscape

• Model Driven Development brings many benefits to the technical implementation of SOA:
  – Platform independence of design assets improves agility
  – Early, model-based testing reduces risk
  – Model-based services repository and services discovery improves reuse
Thank You!

Questions?

Come and see the demo at the Telelogic stand!