Policy Driven Practices for SOA

Lawrence Wilkes

CBDI Forum

www.cbdiforum.com
Agenda

- Enterprise SOA Challenge
- SOA Policy Areas
- Layered Service Architecture as a basis for Policy
- Service Lifecycle as a basis for Policy
- Compliance Testing
- Service Engineering
Core SOA Characteristics

1. **Loose Coupling**
   - Enabling rapid process integration & optimization

2. **Functional standardization**
   - Reuse to reduce cost and deliver consistency across different solutions

3. **Consumer (solution) flexibility**
   - Use alternative and or specialize services

4. **Resource virtualization**
   - Who, What and Where

---

**Consuming Solutions**

- Service A
  - Y
  - X
  - Z

- Service B
  - B
  - A
  - C

**Usage decisions determined by Policy**

**Functional Capabilities/Resources**
Enterprise Challenge

Lots of Disparate Consuming Solutions Driven by Pressing Needs Of Individual Business Sponsors

Lots of Services delivered with good intention, but failing to deliver the full benefits of SOA

Minimal Sharing, still silo’ed, disparate, and meeting only the requirement of individual business sponsors!

Lots of Duplicated, Silo’ed, Disparate, Distributed Capabilities/Resources
Technology Isn’t the Solution

Lots of Disparate Consuming Solutions Driven by Pressing Needs Of Individual Business Sponsors

Enterprise Service Bus, Web Service Protocols, etc

Technology is an important enabler
But it isn’t just a wiring problem!

Lots of Duplicated, Silo’ed, Disparate, Distributed Capabilities/Resources
Managed Service Portfolio

**Consuming Solutions**

X  Y  Z

**Services Provided and Consumed**

*Services Grouped by Domain*

*Business Domain*

*Services Organized by Purpose and Type*

*Services Selected for Sharing, Aggregation or Differentiation*

A  B  C

**Functional Capabilities/Resources**
## SOA – Three Perspectives

<table>
<thead>
<tr>
<th>SOA is a Management Framework</th>
<th>Focus</th>
<th>Interest</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Business and IT Resource Optimization</td>
<td>Strategy and Roadmap</td>
</tr>
<tr>
<td></td>
<td>Business/IT Convergence</td>
<td>Organization and culture</td>
</tr>
<tr>
<td></td>
<td>IT Process for SOA?</td>
<td>IT Process Governance</td>
</tr>
<tr>
<td></td>
<td>Provider/Consumer Supply Chain?</td>
<td>Provisioning and Sourcing Policies</td>
</tr>
<tr>
<td>SOA is an Architectural Framework</td>
<td>Federated Service Architectures</td>
<td>Enterprise Architecture Context</td>
</tr>
<tr>
<td></td>
<td>Service Identification and Specification</td>
<td>Architectural Constructs for SOA</td>
</tr>
<tr>
<td></td>
<td>Service Lifecycle</td>
<td>Architectural Governance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Architectural and Design Policies</td>
</tr>
<tr>
<td>SOA is a Deployment Framework</td>
<td>Run-time deployment of Services and Resources</td>
<td>Standards</td>
</tr>
<tr>
<td></td>
<td>Operational Infrastructure</td>
<td>Service Technology</td>
</tr>
<tr>
<td></td>
<td>Service Management</td>
<td>Run-time Governance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Operational Policies</td>
</tr>
</tbody>
</table>
### SOA Policy Areas

<table>
<thead>
<tr>
<th>Type</th>
<th>Determines/Governs</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program/Process</td>
<td>SOA Delivery process</td>
<td>RAEW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Funding</td>
</tr>
<tr>
<td>Architecture/Design</td>
<td>Use of architectural constructs in the SOA</td>
<td>Layering</td>
</tr>
<tr>
<td></td>
<td>Flexibility</td>
<td>Mediation</td>
</tr>
<tr>
<td>Asset</td>
<td>Change in state - Service lifecycle</td>
<td>Certification</td>
</tr>
<tr>
<td>Sourcing</td>
<td>How Services and associated resources are sourced</td>
<td>Standardization/Commoditization</td>
</tr>
<tr>
<td>Operational</td>
<td>Run-time policies</td>
<td>Monitoring</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SLA</td>
</tr>
<tr>
<td>Security</td>
<td>Permissions</td>
<td>Authentication</td>
</tr>
<tr>
<td>Commercial</td>
<td>How a Service is paid for</td>
<td>Pricing</td>
</tr>
<tr>
<td>Relationship</td>
<td>Obligations between different parties</td>
<td>Provider/Consumer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IT/Business</td>
</tr>
</tbody>
</table>
Layered Service Architecture

- Reasons for Layering
  - Higher degrees of reuse/sharing
  - Flexibility in assembly of Services at different layers
  - Functional standardization and commoditization in lower levels
  - Customization in higher layers
  - Separation of concerns
  - Determine policies by layer

- Policies Vary by Layer. E.g.
  - Different Sourcing permitted
  - Degree of Standardization/ Differentiation allowed
Service Classification - Layers

Order System

Product Dev System

Stock Control Application

Solution Layer
(presentation and dialog)

Process Services
(orchestration layer)

Core Business Services
(“backbone” layer)

Underlying Services
(that need a facade)

Utility Services
(high reuse layer)

Order Fulfillment Service

Stock Management Service

Orders Service

Products Service

Stock Movements Service

Stock Reordering

Customers Service

Purchasing
(from highly generic component)

Accounts Receivable API
(from legacy Accounting System)

AddressReformatter

CurrencyConversionService

Order System

Order System

Product Dev System

Product Dev System

Stock Control Application

Stock Control Application
Basis for Single and Shared Service Policy

Orchestrate operations from many core business operations
Support process unique processing
Store process level information

**Single Service** provides consistent view of corporate data and business rules
Provides a 360° view of the resource
Stores a record of each instance of each business type
Applies common validation and business rules

Exploit pre-existing functionality for wider reuse
**Aggregate** functionality from pre-existing Services and systems

The most widely reused **Shared Services**
Services that perform widely used sub-routines, operations
Critical policy area
Determines economics, flexibility, competitive differentiation and standardization
Determines sets of standard services based on economics and feasibility
Manage solution usage on basis of competitive differentiation
- Core/Context
- Core/Non Core
Manage sourcing on basis of economics

Increasing Commodity Services
Differentiated Service Behavior
Differentiated Services
Increasing Customization
# Basis for Architecture and Design Rules

<table>
<thead>
<tr>
<th>Example rules</th>
<th>Dependencies allowed</th>
</tr>
</thead>
<tbody>
<tr>
<td>May be called by apps that support other business processes</td>
<td>May call Core Business &amp; Utility services directly</td>
</tr>
<tr>
<td>Cyclic dependencies not permitted, except for callback. May not call Process Services</td>
<td>May call other Core Business, Underlying and Utility Services directly</td>
</tr>
<tr>
<td>May not call Core Business or Process Services</td>
<td>May call Utility Services, but normally would not</td>
</tr>
<tr>
<td>Cyclic dependencies not normally permitted</td>
<td>May call other Utility Services directly</td>
</tr>
</tbody>
</table>

**Process Services**
(orchestration layer)

**Core Business Services**
(“backbone” layer)

**Underlying Services**
(that need a facade)

**Utility Services**
(high reuse layer)
Driving Service Architecture

<table>
<thead>
<tr>
<th>Scope</th>
<th>Focus</th>
<th>Service Identification</th>
<th>Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Solution Driven</strong></td>
<td>Project Business Process</td>
<td>Business Process Value Chains</td>
<td>Process Modelling Use Case</td>
</tr>
<tr>
<td></td>
<td>Business Optimization</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Resource Integration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Process Agility</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Domain Driven</strong></td>
<td>Enterprise Domain</td>
<td>Analysis of Business Domain</td>
<td>Business Type Models</td>
</tr>
<tr>
<td></td>
<td>Resource Optimization</td>
<td>Classification into Layers</td>
<td>Data-centric Reuse analysis</td>
</tr>
<tr>
<td></td>
<td>Sharing and Consistency</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Standardization/Commoditization</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Service Flexibility - Provider Agility</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>System Driven</strong></td>
<td>Existing Systems Application</td>
<td>Existing systems and databases</td>
<td>Current Systems Analysis</td>
</tr>
<tr>
<td></td>
<td>Existing APIs Integration</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Process Driven**

- Solution Model
- Data Centric
- Domain Model
- Current Systems Model
- Bottom Up

**Process Services**
- Order Fulfillment Service
- Orders Service
- Products Service

**Core Business Services**
- Accounts Receivable API

**Underlying Services**
- Address Reformatting Service

**Utility Services**

Architecting for Agility

Specialized Solutions
On Demand Assembly

Pick Exterior Color
Pick Interior Color
Pick Wheel Style

Finished Product

Standard
GT
Special

Base Product

Flexible Sourcing of Components

Commoditized Services

Planning and Design

Run-Time

Service Consumer

Policy Driven

Service Requestor Applications

Alternative Service or Provider

Process Service

Management Service

Business Services

Service Provider Applications

Service Provider

Policy Driven
The Service Life Cycle – Enabling Governance

- **Planned** /include proposed service in portfolio plan
- **Specified** /prepare service specification and WSDL
- **Being Provisioned** demand for operations arises / …
- **Provisioned** /handover tested service
- **Certified** /confirm service offers required quality
- **Published** /publicize service, catalog and subject to change control
- **Operational** /deploy service
- **Retired** /withdraw obsolete service
- **Archived** /archive service artifacts

Lifecycle Governance over state change

- **State (pre)**
- **Activity**
- **Compliance Check**
- **State (post)**

Policy Driven

Policy Driven
How can Policies be applied across different tools?

Policies may be tool specific, with tool specific definitions

How is compliance checked?

Service is defined in many different tools

How is consistency maintained?

How is the compliance with the specification checked?

Changing State may mean
- Moving from tool to tool
- Changing Level of Abstraction

OMG RAS – Reusable Asset Specification

OMG UML 2 – Models used to document service and the SOA

WS-protocols – even if the Service is not a WS

Use of WS-Policy

Standards that may help share Service artefacts or information across the lifecycle
Need for Richer Service Specifications

- Operation signatures do not explain enough
- WSDL is not good at explaining service behavior

CBDI Service Description (primarily used in Planning)
- Lightweight – not a specification
- Described in business, not technical terms

CBDI Rich Service Specification
1. Interface Definition (signatures of all the operations)
2. Behaviour Definition (without pre-empting how implemented) e.g. pre-post condition pairs
3. Service Information model
4. Mandatory Message Sequences
5. Properties and Features
6. Quality of Standards Compliance

Functional Specification
Non-functional Specification
Role of Registry in the Service Lifecycle

Registry becomes “System of Record” for Service Lifecycle
Policy Compliance Points

Service Provision

Asset Management Tools

Developer Tools

Service Consumption

Asset Management Tools

Developer Tools

Service Consumption

Validate Service Consumption

Discover

Validate Service

Consume

Validate Run-time Compliance
Validate SLA

SM/ESB

Validate Provider

SM/ESB

Validate Consumer

Specify

Validate Service

Validate Specification

Registry

Publish

Certify

Operate

Validate Service Design

Validate Run-time Compliance
Validate SLA

Validate Consumer

Validate Provider

Validate Service

Validate Specification

Specify

Discover

Consume
# Sample Governance Compliance Checks

<table>
<thead>
<tr>
<th>Compliance Check</th>
<th>Type of Check and Standards Relevance</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS-I profile</td>
<td>Check compliance with WS-I profiles to ensure interoperability</td>
</tr>
<tr>
<td>WS-Security</td>
<td>Enforce and validate Security policies</td>
</tr>
<tr>
<td>Schema</td>
<td>Validate XML Schemas, validate that Services use the correct schema</td>
</tr>
<tr>
<td>Classification</td>
<td>Validate classification of Services. Registries provide classification mechanisms</td>
</tr>
<tr>
<td>Architecture</td>
<td>Proper assignment to layer, compliance with dependency policies</td>
</tr>
<tr>
<td>Design Policies</td>
<td>User defined methodology conformance to best practices.</td>
</tr>
<tr>
<td>Service Specification</td>
<td>Completeness of specification according to user defined methodology</td>
</tr>
<tr>
<td>Approved Provider</td>
<td>Inspect endpoint references against known and approved providers. For example</td>
</tr>
<tr>
<td>Service Consumption</td>
<td>Ensure that only Services published in catalog are consumed. For example</td>
</tr>
<tr>
<td>SLA</td>
<td>Monitor compliance with SLA policies. SLA definitions and hence compliance checks are likely be proprietary to the WSM/SOAM/ESB product</td>
</tr>
<tr>
<td>Business Policy Compliance</td>
<td>Inspect Service Requests and Responses to ensure business rule compliance, and/or transform Service Requests and Responses based on business rules. Business Rules Engine defines compliance tests. WSM/SOAM/ESB can enforce business-based mediation rules (routing, transformation, etc)</td>
</tr>
<tr>
<td>Regulatory or Auditing Compliance</td>
<td>Inspect Service Requests and Responses to ensure regulatory compliance, and auditing requirements. Use WSM/SOAM/ESB. Typically user defined. Some products may have pre-defined templates.</td>
</tr>
</tbody>
</table>
Relationship Governance

- Use policies as a way of managing relationships
- Compliance works both ways and places obligations on both parties

Use policies as a way of managing relationships
Compliance works both ways and places obligations on both parties
Service Engineering Process Context

SERVICE PORTFOLIO PLANNING

- Define Policies
- Identify Services
- Describe Services
- Publicize Portfolio Plan

SERVICE PROVISIONING

- Specify a Service
- Acquire the Service
- Certify, Deploy Service
- Publish Service in Catalog

BUSINESS MODELING

- Define business capabilities
- Define business relationships
- Define business policy
- Model Business Semantics
- Model Business Capability
- Model Value Chains

BUSINESS PROCESS DESIGN

- Model Business Process

SOLUTION DELIVERY

- Design Software Solution
- Request Services and Operations
- Construct Software Solution
- Test Software Solution
SPP Policies

- Service View policies govern portfolio content Service identification and classification:
  - Service Layering rules
  - Service Dependency Rules
  - Standardization and customization
  - Sourcing
  - Target consumers, QoS
  - . . .

- Implementation View policies govern mapping to automation to automation units:
  - Sourcing
  - Component selection and or design criteria
  - Integration

- Deployment View policies govern allocation of automation units to technical infrastructure:
  - Performance, Security
Conclusions

- SOA Policies fall into many areas
  - Process
  - Architecture
  - Operational
  - Relationships
- Layered Service Architecture drives much policy thinking
- Service lifecycle provides a framework for managing compliance governance

- Policies must be flexible
  - Know when to enforce, and when to allow optionality
  - Many policies must be checked by hand – don’t overburden the organization with bureaucracy
Relevant CBDI Reports

- Practical Service Specification and Design - a five part series commencing with:
  [Link](http://www.cbdiforum.com/secure/interact/2005-03/Practical_Service_Spec.php)

- Service Portfolio Planning Revisited

- Improving SOA Governance with the Systinet Business Services Registry

- Software Development Asset Management with LogicLibrary Logidex
  [Link](http://www.cbdiforum.com/secure/interact/2005-06/Software_Dev_Asset_Man_Logidex.php)

- The Service Lifecycle

- SOA Governance in the Life Cycle
Independent Insight for Service Oriented Practice

- Monthly CBDI Journal
  - Best Practice Series e.g.
    - Practical Service Identification and Specification
  - Enterprise SOA
  - Developing the Architectural Framework for SOA
- Service Oriented Business Series e.g.
  - Telco
  - Insurance
  - Pharmaceutical
  - Automotive
- Market Trends e.g.
  - Service Management
  - ESB
- 15,000+ subscribers worldwide
- Architects, Business Analysts, CIOs, CTOs, Product Managers,

Some *Free* Resources
- SOA and Web Service Roadmap
- SOA Fundamentals

Consulting and Education
- SOA Roadmap Planning
- Service Portfolio Planning
- Business Requirements for SOA
- Technology Infrastructure for SOA

www.cbdiforum.com