Metadata Management: New Requirements

Jason Bloomberg & Ron Schmelzer
ZapThink, LLC

Take Credit Code: NOMETA

zapthink

zapthink

The Best Technology…

- Is complex on the inside yet simple on the outside

- The secret is the *abstraction layer*

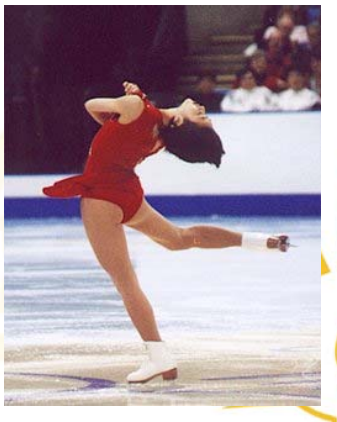
SOA Abstraction Layer

- Coarse-grained business Services
- Location independence
- Underlying systems loosely coupled from Service consumers
- Dynamic discovery and invocation

Sounds good, but how does it work?

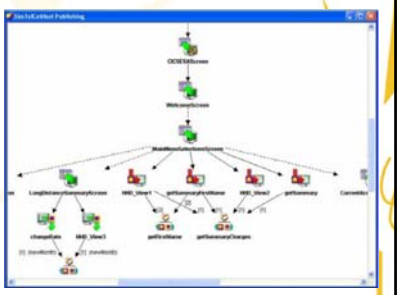
The Secret of the Best Ice Skaters

- Making ice skating look difficult is easy
- Making it look easy is difficult
- A good abstraction makes it look easy, but actually adds complexity



The Secret Sauce: Metadata

- Service contracts enable Services abstraction
- *Business* logic in composite applications represented in metadata
- Composite applications built *declaratively* (through configuration), not programmatically (with code)
- Puts business logic in the hands of business users!



What are Metadata?

- Literally, *data about data*
- More broadly, *data about the workings of a system*, as opposed to the data the system works with

This is a purchase order
It has an address field
It has an item field
It has a dollar field
...

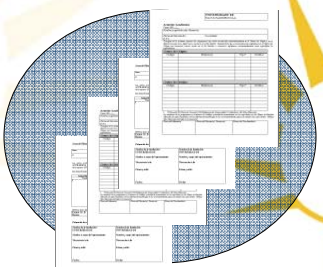
Metadata

ABC Co.
32 Widgets
$47.00

Data

Metadata for SOA

- Contract metadata
  - What a Service should do (functional)
  - How a Service should behave (non-functional)
- Process metadata
  - How processes are configured
  - How processes are composed
- Policy metadata
  - What rules apply in various situations
- Schemas
- …



Metadata Management Requirements

- In an SOA, business logic is in *metadata*, not code
- Types of metadata
  - Service interfaces
  - Business process flows
  - Policy definitions
  - Event messages and types
  - Schema and semantics
- Need: development tools and methodologies

Goals of Metadata Management

- Design time storage and discovery of all kinds of metadata
- Service versioning/lifecycle management
- Runtime Service discovery
- Governance
- Composite application/business process management

SOA is not about the *code*…

it's about the ***metadata***

Metadata Management & SOA

- Metadata at the core of any SOA implementation
  - Contracts, policies, SOBA configurations, schemas…
- SOA Governance depends upon metadata as well
- "Registry/Repository" at the center of the SOA metadata management puzzle

What is a Registry?

- ***Metadata System of Record*** for SOA
  - Interfaces
    - UDDI – *Web Services*
    - LDAP
    - Microsoft Word
    - Word of Mouth

- What do you want from your registry?
  - Design time capabilities to discover Services
  - Interoperability across runtime infrastructure
  - Ability to use at runtime for binding….

What is a Repository?

- ***Asset Store*** for SOA
  - Artifacts of design
    - Conformance Documentation
    - Contract versions
    - Sample Code
    - Profiles
    - Schema
  - Runtime operational store
    - Messages
    - Policies
    - Logs, Security Certificates, Keys, SAML artifacts
    - Transformations
    - Contracts
- Interfaces:
  - ebXML
  - CVS / Source Code systems
  - WebDAV
  - File Systems
  - SQL
  - XQuery

Standards vs. Products

- STANDARDS provide *interoperability* while PRODUCTS provide *functionality*
- UDDI: A standard for Registry Interoperability; ebXML & JAXR other registry-related standards (may be more?)
- UDDI and the other standards are NOT products!
- There are many other ways to interact with registries
- Registries do more than just UDDI
- Repositories have a UDDI role, too!

The Business Benefits

- *Provide Governance*
  - Single point of control
  - Enable distributed development, but centralized management
  - Simplify policy design and management
- *Facilitate Reuse*
  - The more you know, the more you can reuse
  - Reduces redundancy
  - Enables discovery
- *Increase Agility*
  - Runtime usage of registries and repositories enables dynamic change
  - Metadata controls it all
  - Contract negotiation?

Service Registry/Repository: Roles

Enablement	Publishing	Discovery	Management
How do I quickly • design • develop • deploy my Web Services SOA software?	How do I certify and approve my Services for: • company • IT • SOA standards and deployment readiness?	How do I: • share • find • use my Services anytime, anyplace, anywhere?	How do I make Services scale for: • availability • reliability • integrity • measure overall quality?

Source: Systinet

Service Registry/Repository: Roles

Enablement	Publishing	Discovery	Management
• Mapping Business Processes to SOA Model • Business, IT governance development • SOA foundation development • Web Services development	• Registering Business Services and SOA information • Certification Process • Change Management • Approval Process & Management	• Locating and using Services • Design time usage • Run-time usage • Configuration and Change • Coordination Management	• Operating, monitoring, and changing Services •Change, Impact Analysis •SLA & Policy Management • Integration - Security & Identity Mgmt •Log, Reports

Source: Systinet

Lifecycle/Governance/Metadata

- Flashline
  - Flashline for SOA
- Infravio
  - X-Registry
- LogicLibrary
  - Logidex
- Mindreef
  - Coral
- SOA Software
  - Registry
- Systinet (now Mercury)
  - Registry, Blizzard
- WebLayers
  - Center

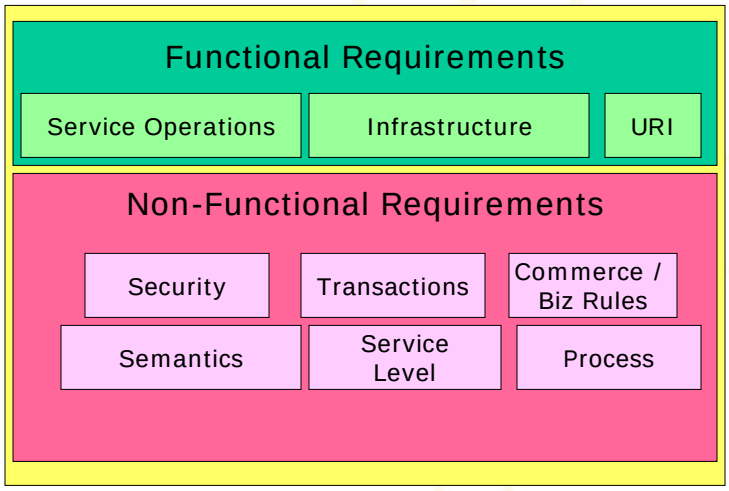
Key Management/Lifecycle Standards

- WSDM (Web Services Distributed Management)
  - Seeks to unify management infrastructures by providing a vendor, platform, network, and protocol neutral framework for enabling management technologies to access and receive notifications of management-enabled resources
- WS-Management
  - A general SOAP-based protocol for managing systems such as PCs, servers, devices, Web services and other applications, and other manageable entities
- UDDI (Universal Description, Discovery & Integration)
  - A standard interoperable platform that enables companies and applications to quickly, easily, and dynamically find and use Web services over the Internet, and also allows operational registries to be maintained for different purposes in different contexts

Service Model Drives Contract-First Development

- Service contracts specify required functionality to IT and provided functionality to the business
- Service model represents the clearinghouse for information about IT environment
- Contracts go beyond WSDL:
  - Usage policies
  - Security policies
  - Consumer delivery contracts
  - Service-level agreements, etc.

What's in a Contract?



Aspects of a contract

- **Service Description**
  - What does the Service provide for prospective consumers?
  - Description of what is accomplished by the Service
  - Quality of Service or applicability constraints
  - Constraints on types of consumers or other consumer requirements
- **Service Usage Description**
  - How is Service used?
  - Message and semantic formats for requests
  - Conditions for particular outcomes / behaviors
  - Process leading to outcomes

Aspects of a contract (cont.)

Service Interaction Description

- How can we communicate with Service?
- Description of acceptable communication protocols
- Invocation style (request/response, one-way, event-driven)
- Expectations of latency

What's NOT in the Contract

A "Contract" is an expression of ***visible*** aspects of Service behavior.

It does NOT specify:

- Service Implementation Details
  - Programming model
  - Object references
  - In-memory representations
- Examples
  - Exposing Java Classes
  - Serialization of Java Objects

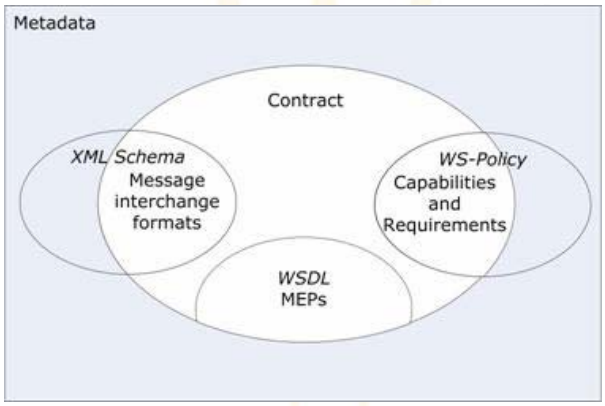
WSDL: Service Contract Starting Point

```
<?xml version="1.0" encoding="UTF-8"?>

<definitions  name ="DayOfWeek"

targetNamespace="http://www.roguewave.com/soapworx/examples/DayOfWeek.wsdl"

xmlns:tns="http://www.roguewave.com/soapworx/examples/DayOfWeek.wsdl"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns="http://schemas.xmlsoap.org/wsdl/">

 <message name="DayOfWeekInput">
  <part name="date" type="xsd:date"/>
 </message>
 <message name="DayOfWeekResponse">
  <part name="dayOfWeek" type="xsd:string"/>
 </message>

 <portType name="DayOfWeekPortType">
  <operation name="GetDayOfWeek">
   <input message="tns:DayOfWeekInput"/>
   <output message="tns:DayOfWeekResponse"/>
  </operation>
 </portType>

 <binding name="DayOfWeekBinding" type="tns:DayOfWeekPortType">
  <soap:binding style="document"
   transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="GetDayOfWeek">
   <soap:operation soapAction="getdayofweek"/>
   <input>
    <soap:body use="encoded"
     namespace="http://www.roguewave.com/soapworx/examples"
     encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
   </input>
   <output>
     <soap:body use="encoded"

namespace="http://www.roguewave.com/soapworx/examples"

encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </output>
   </operation>
  </binding>

  <service name="DayOfWeekService" >
   <documentation>
    Returns the day-of-week name for a given date
   </documentation>
   <port name="DayOfWeekPort"
binding="tns:DayOfWeekBinding">
    <soap:address
location="http://localhost:8090/dayofweek/DayOfWeek"/>
   </port>
  </service>

</definitions>
```

- Sample WSDL file
- Describes port type, binding, input, output, Service name and URL

Source: Rogue Wave

Contract Metadata Beyond WSDL



State of the Art!

Source: W3 - http://www.w3.org/2004/08/ws-cc/jsklms-20040903

Thank You!

ZapThink is an advisory, analysis, & influence firm focused exclusively on Service-Oriented Architecture, Web Services, & Enterprise Web 2.0.

Read our new book, *Service Orient or Be Doomed! How Service Orientation Will Change Your Business.*

Ronald Schmelzer
rschmelzer@zapthink.com

Jason Bloomberg
jbloomberg@zapthink.com