This Presentation

- **SOA (today)**
  - UPMS from OMG, and Model Driven Solutions for SOA
  - SCA from OSOA
  - Wrapping and Adapting Legacy Systems

- **SOI (today)**
  - JEE and JBI from JCP
  - Sun's SGF

- **SOE (ten years out ;-)**
  - Agency API’s and the Federal Target Architecture
  - Model Based Acquisition and Service Based Procurement
  - Semantic Interoperability and Software Factories
  - Orchestrating and Choreographing with the FTA Test Harness
  - Market Driven, Dynamic Col’s
This Presentation

• SOA (today)
  – UPMS from OMG, and Model Driven Solutions for SOA
  – SCA from OSOA
  – Wrapping and Adapting Legacy Systems

• SOI (today)
  – JEE and JBI from JCP
  – Sun’s SGF

• SOE (ten years out ;-)
  – Agency API’s and the Federal Target Architecture
  – Model Based Acquisition and Service Based Procurement
  – Semantic Interoperability and Software Factories
  – Orchestrating and Choreographing with the FTA Test Harness
  – Market Driven, Dynamic Col’s

UPMS Fundamental Concepts

• Service Oriented Business Architecture (SOBA) Model
  – SOBA is modeled as a Collaboration of enterprise level Participants using Service Contracts

• Participant
  – A specification of the responsibility to perform specific functions in the context of a business process

• Collaboration
  – A set of two or more participants interacting to carry out a business process to achieve some joint purpose
    • An interaction between participants conforming to a Service Contract is modeled as a UML Collaboration Use

• Service Contract
  – A collaboration that defines a conversation in which services are consumed and provided
    • A Service Contract is modeled as a UML Collaboration
    • Long running conversations may be specified using an Owned Behavior
SOBA Participants and Service Contracts

First the submitter submits a bill to the receiver...

While one Participant requests the service and the other responds, information may flow both ways during the interaction.

First the submitter submits a bill to the receiver...

...then either the bill is successfully delivered or it is returned.

Service Contract Owned Behavior
Financial Management is responsible for providing a number of Acquisition Accounting services.

The Acquisition Accounting Services are delegated to a number of different Participants within Financial Management.

Acquisition is an External Consumer relative to Financial Management.
Participants at this level could roughly be individual people or system functions.

Internal interactions needed to carry out the required business services.

Business service provided and consumed by Receivables Accounting.

Receivables Management Activities

Activity

Sent event

Establish Unfulfilled Customer Order

Customer Order Established

Customer Order Rejected

Generate Recurring Receivable

Receivable Accepted

Receivable Rejected

Information flow
Establish Unfilled Customer Order Subactivities

- Establish Unfilled Customer Order
  - Input parameters
    - customer order established
    - customer order rejected
    - customer order established
  - Subactivity
    - Record Unfilled Customer Order
    - Output parameter
      - general ledger transaction
      - invoice

Record Unfilled Customer Order Behavior

- Can be specified using basic *UML Activity Diagrams*
  - Or use BPMN (not part of UPMS) for BPMS (from left to right!)
From SOBA to Reference Architecture

- The Service Components that implement the SOBA are modeled using UML Components
  - Presentation Components provide user access to application services
  - Service Components provide transactional implementation of application services
  - Data Components persist data between application transactions

Provided and Required Interfaces

- Usage dependency
- Realization
- The operations defined on Service Interfaces provide the basis for bidirectional service interactions
- Ports are typed by Port Types that realize and use service interfaces
- Required interface
- Provided interface
- Ports with "conjugate" interfaces may be interconnected

13 14
George Thomas, GSA EA Group 7-Sep-07
Port Types from Service Contracts

The Participant Types act as the Service Interfaces

Port Types for Composite Service Contracts

Participant Types get ports corresponding to the roles they play in each of the nested services.

The types of the ports are the Port Types defined for the nested services.
The plays role dependency specifies the interfaces of a service component by declaring how it is to participate in the business architecture.

The service component gets a port corresponding to each role played by the indicated participant.
Delegation of a provided service to an internal part

A composite service can be delegated piecewise to multiple internal parts

Internal service connection

Delegation of a consumed service by an internal part
Receivables Accounting Business Architecture

Receivables Accounting Reference Architecture

User of a consumed service by multiple internal parts
From Service Models to Web Service Implementations

Logical View

A single bidirectional connection between consumer and provider service ports

Physical View

The consumer requests services via the provider's port

The provider "calls back" via the consumer's port

For synchronous services, an alternative is to map "call back" operations into return types

Example Web Services Generation

Example of web service implementation:

```xml
<wSDL:port type name="BillSubmission.BillSubmissionReceiverInterface">
  <wSDL:operation name="submitBill">
    <wSDL:input message="tns:BillSubmissionCluster" name="billSubmission">
    </wSDL:input>
  </wSDL:operation>
</wSDL:portType>

<wSDL:port type name="BillSubmission.BillSubmissionSubmitterInterface">
  <wSDL:operation name="notifyBillDelivered">
    <wSDL:input message="tns:BillDeliveredCluster" name="billDelivered">
    </wSDL:input>
  </wSDL:operation>
  <wSDL:operation name="notifyBillReturned">
    <wSDL:input message="tns:BillReturnedCluster" name="billReturned">
    </wSDL:input>
  </wSDL:operation>
</wSDL:portType>
```
From Reference Architecture to Reference Implementation

<table>
<thead>
<tr>
<th>Logical System Architecture (ZIFA)</th>
<th>Physical Technology Architecture (ZIFA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Platform Independent Model (MDA)</td>
<td>Platform Specific Model (MDA)</td>
</tr>
<tr>
<td>Segment Architecture (FEA)</td>
<td>Solution Architecture (FEA)</td>
</tr>
</tbody>
</table>

Reference Architecture       Reference Implementation

Financial System Specification       Financial System Implementation
Service Interfaces          Web Services
Service Components          Service Façades
Service Manager Components        Platform (JEE) Components
Behavioral Specifications        Language Functions
Message Specifications          XML Schemas
Data Manager Components          Data Bases
Persistent Data Specifications       Data Base Schemas

Solution Architecture, Target J2EE Technology Platform

Adapter Tier
- J2EE Application Server
  - File Adapters
  - Web Service Adapters
- File
- FTP
- SOAP/HTTP

Integration Tier (ESB)
- JMS Broker
  - Request Queue
  - Result Topics
- JMS

Transactional Services Tier
- JMS Application Server
  - EJB
  - Data Manager
  - JDBC

Meta Data Repository
- Management Console
- Directory
  - LDAP
  - Directory
  - HTTP/HTTPS

Server-Side Presentation
- GL System
Software Factories

- Markup
  - Tags used to generate the code for the target platform
- UML Profile
  - Standard extension mechanism
    - <<stereotypes>>
    - (tagged="values")
  - <<Provision>>
    - To tiers:
      - Integration
      - Services
      - Data
    - J2EE platform
      - MDB, SSB, etc.

Same Source (PIM) For New JEE/JBI SA Target (PSM)

Integration Tier
- ESB Centralized Management Server
- JMS Broker
- JBI Runtimes
- HTTP BC

Transactional Service Tier
- EJB Container
  - Transaction Manager
  - RMI
  - Data Manager

Adapter Tier
- Transform Script
- Adapter Program
- JMS
- SOAP/HTTP

Now Targeting Service Units and Service Assemblies

SA SU SU SU
This Presentation

- SOA (today)
  - UPMS from OMG, and Model Driven Solutions for SOA
  - SCA from OSOA
  - Wrapping and Adapting Legacy Systems

- SOI (today)
  - JEE and JBI from JCP
  - Sun’s SGF

- SOE (ten years out ;-)
  - Agency API’s and the Federal Target Architecture
  - Model Based Acquisition and Service Based Procurement
  - Semantic Interoperability and Software Factories
  - Orchestrating and Choreographing with the FTA Test Harness
  - Market Driven, Dynamic CoI’s

SCA Business System

- Services, References, and Wires
  - Composites implemented by components
  - Components implemented by composites

![SCA Business System Diagram]

System

- Composite X
  - Service
  - Wire

- Composite Y
  - Component A
  - Wire
  - Implementation
  - Composite A

- Component B
  - Wire

- Composite Z
  - Reference
  - Wire
  - Implementation
  - Composite B
SCA Composite Graphic Notation

Service
Java interface
WSDL PortType

Composite A

Component A

Property setting

Component B

Binding
Web Service
SCA, JCA, JMS

Reference
Java interface
WSDL PortType

Binding
Web Service
SCA, JCA, JMS

Wire

SCA Service Assembly Model

- Tight coupling
  - Composites (a deployment unit)
    - Programming ‘in the small’
- Loose coupling
  - Systems (a composite applications)
    - Programming ‘in the large’
SCA Policies, Profiles and QoS

- Policy Intents are attached to SCA model elements
- Policy Sets declare what QoS intents they provide and which Bindings they are for
- Intents index into a PolicySet for each Binding

---

SCA Policies, Profiles and QoS

- Framework consists of
  - SCA policy intent
    - Each represent a single abstract QoS intent
  - SCA Profile
    - Aggregates a set of abstract, cross-domain, QoS intents to represent an overall QoS
  - SCA Policy Sets
    - Represents a collection of concrete policies to realize an abstract QoS intent
  - WS-Policy
    - A syntax for concrete policies in policy sets (others possible)

- Interaction policies affect the contract between a service requestor and a service provider
- Implementation policies affect the contract between a component and its container
This Presentation

• SOA (today)
  – UPMS from OMG, and Model Driven Solutions for SOA
  – SCA from OSA
  – Wrapping and Adapting Legacy Systems

• SOI (today)
  – JEE and JBI from JCP
  – Sun’s SGF

• SOE (ten years out ;-)
  – Agency API’s and the Federal Target Architecture
  – Model Based Acquisition and Service Based Procurement
  – Semantic Interoperability and Software Factories
  – Orchestrating and Choreographing with the FTA Test Harness
  – Market Driven, Dynamic Co’s

Wrapping Legacy - Adapter and Façade Patterns

• A façade provides a simplified interface to a large body of software to;
  – Reduce dependencies of software consumers
    • by wrapping a collection of APIs with a single API

--------------------------------------------------

• An adapter is used when we must respect an existing Service Interface that is;
  – Incompatible with another existing Service Component Interface
    • by wrapping the new Adaptor interface around the existing Service Component interface
Modernizing Legacy - MDA Top Down, ADM Bottom Up

Discovery of System Details and generation of Technology Specifications is largely automated

<table>
<thead>
<tr>
<th>As-Is</th>
<th>To-Be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation Independent</td>
<td>One-GSA Business Model</td>
</tr>
<tr>
<td>Platform Independent</td>
<td>Knowledge Discovery Model</td>
</tr>
<tr>
<td>Platform Specific</td>
<td>System Specification</td>
</tr>
</tbody>
</table>

This Presentation

• SOA (today)
  – UPMS from OMG, and Model Driven Solutions for SOA
  – SCA from OSOA
  – Wrapping and Adapting Legacy Systems

• SOI (today)
  – JEE and JBI from JCP
  – Sun’s SGF

• SOE (ten years out ;-)
  – Agency API’s and the Federal Target Architecture
  – Model Based Acquisition and Service Based Procurement
  – Semantic Interoperability and Software Factories
  – Orchestrating and Choreographing with the FTA Test Harness
  – Market Driven, Dynamic CoI’s
JEE5 – now with JBI!

- **Java EE service units**
  - Using Java EE service units, end users can deploy and manage a composite application as a single entity; a JBI service assembly

- **Transaction support**
  - Enables Java EE web services and JBI services to participate in a single transaction
  - For example, a BPEL process can call an EJB that updates a database, all in the same transaction

- **Security support**
  - This feature enables separate JBI components to make use of a single authentication mechanism
  - In practice, a user signs on once, and the JBI system propagates the security credentials to the various JBI components as needed

- **Cluster support**
  - Deploy a Java EE service unit to an application server cluster
  - When the JBI service assembly is deployed to the cluster, the Java EE service units are also deployed to each server instance in the cluster

---

Aspect Oriented Policy - Intercepting Filter Pattern
A Governed Service

- Includes Non-Functional Requirements (NFR) as Policy Attachments
  - QoS, 'ilities', Security, Privacy
  - Generic, or Industry specific
- Published in UDDI, pointed to by WSDL
  - Service Façade should abstract the FBL
- Contract governance independent of Service governance!
  - Aspect decorations change but the Int/Impl doesn’t

Layered Governance Contracts

- Service gateway model
  - Relegate governance to edge devices
- Service governance intermediaries
  - Vary the granularity of governance contracts
  - Apply governance at multiple enforcement points in the architecture
Dynamic Policy Model

- Aspects, Aspect Groups
  - Aspect Engine (a JBI SE) applies the appropriate ‘Policy Pipeline’

This Presentation

- SOA (today)
  - UPMS from OMG, and Model Driven Solutions for SOA
  - SCA from OSOA
  - Wrapping and Adapting Legacy

- SOI (today)
  - JEE and JBI from JCP
  - Sun’s SGF

- SOE (ten years out ;-)
  - Agency API’s and the Federal Target Architecture
  - Model Based Acquisition and Service Based Procurement
  - Semantic Interoperability and Software Factories
  - Orchestrating and Choreographing with the FTA Test Harness
  - Market Driven, Dynamic CoI’s
Agency API's -> FTA -> Emergent G2G/G2B/G2C Mashups

- www.ProgrammableAgency.gov
  - Ala http://www.programmableweb.com
  - NFR and QoS Policy Attachments and Pipelines
  - Result is the Federal Target Architecture (FTA) and Sequence Plan
    - Apply Performance Based Contracting to Service Choreographies
    - Apply Competitive Sourcing to Service Orchestrations
- Changes IT Procurement, a public/private ‘design by contract with America’
  - Public owns and manages Service Specifications, an Authoritative Reference Architecture (ARA) of Public Services
    - Fed LoB and eGov SSP’s/CoE’s manage ARA’s in external Standards Organizations (ala FEA-PMO FTF and RMS in OMG!)
  - Private product or service applied, results in (configured instance of) Service implementations, a Reference Implementation (RI)
    - Open Source ARA/RI ‘test compatibility kits’ as FTA test harness, hosted by IOI LoB SSP’s
  - Business and Technical Policies
    - Service contracts and governance policies move from machine parseable (XML syntax based) to machine interpretable (Semantic technology based) expressions

Model Based Acquisition, Service Based Procurement

Each RFP will specify physical interfaces to ensure interoperability

The appropriate parts of this model will be included in an RFP as a logical specification

Closed and Open Source COTS vendors or SI’s are expected to supply these (whether we build, buy, or adapt)
### Semantic Interoperability and Software Factories

- **Design time semantics**
  - Languages, Specifications and Policies grounded in logical formalisms
  - Examples;
    - Concepts and Synonyms
      - SCA Policy Intents/Sets - SGF Policy Aspects/Groups
    - Conflated concepts, and Polysemy
      - ‘Service Component’ - FEA SRM, UPMS, SCA, ...

- **Runtime semantics**
  - Recommenders
    - Automating Capability and Requirements matching, suggesting Service Contracts between Providers and Consumers
  - Ratings
    - Monitoring and reporting on Service Contract SLA’s (Policies)

- **Software Factories leverage Semantics**
  - Further enhancing the utility of today’s MDA (MDD/MDE) and BPMS techniques

### Orchestrating and Choreographing w/the FTA Test Harness
Market Driven, Dynamic CoI's

- The strategy pattern is useful for;
  - Situations where it is necessary to dynamically swap application algorithms
    - For dynamic Service Compositions
    - Lets algorithms (orchestrations, choreographies) vary independently from Consumers (end users)
- New Service Provider?
  - Chosen based on Recommender or Rating
    - Continuous Improvement = Competitive Differentiation
- Enabled by semantically grounded governed services
  - Ala Sun’s SGF for example…

References

- SOA
  - UPMS from OMG, and Model Driven Solutions for SOA
    - http://soa.omg.org/
  - SCA from OSOA
    - http://osoa.org
  - Wrapping and Adapting Legacy
    - Strategy and Façade patterns (Wikipedia)
    - Architecture Driven Modernization (ADM) at OMG
- SOI
  - Java Business Integration
    - http://java.sun.com/developer/technicalArticles/J2EE/sunjsaas_ejuge/  
  - Service Governance Framework
- SOE
  - Agency API’s and the Federal Target Architecture
    - See http://www.programmableweb.com/
Thank You!

- **Contact me:**
  - George Thomas
    - General Services Administration
      - Office of the Chief Information Officer
      - Enterprise Chief Architect
    - Federal CIO Council
      - Architecture and Infrastructure Committee
      - Services Subcommittee co-chair
    - Object Management Group
      - Government Domain Task Force
      - Steering committee
  
  - g.thomas@gsa.gov
  - 202.219.1979