Service-Oriented Architecture
A View From the Field

Paul C. Brown, Ph.D.
Principal Software Architect
What is a Service?

- **A coherent package of commonly used functionality**
  - e.g. Sales Order Management
    - Place Order
    - Modify Order
    - Cancel Order
    - Get Order Status

- **Packaged for consistent re-use**
  - Readily accessible from many places

- **A de-facto standard in the enterprise**
  - The preferred access mechanism for the functionality

- **Most functionality already exists**
  - In one system, now accessed in many ways
  - Duplicated in multiple systems

- **The goal is to save/make money!**
  - Standardize the functionality so that what the next project needs is already there
    - Reduce IT costs (the “small” ROI)
    - Provide competitive advantage (the “big” ROI)
What’s in the Service Package?

- **Pure functions – computations**
  - Client supplies all input data and receives results
  - Totally stateless
  - Limited value
    - Very few pure functions in the enterprise

- **Data management – an information repository**
  - Manages data related to a particular concept – including persistence
    - E.g. sales order information
  - CRUD operations mediate access - (create, read, update, delete)
  - Increased value
    - Provides a managed home for a category of information

- **Managed business functionality**
  - Data + Business Rules for its management
  - Operations become business relevant
    - Place order, cancel order, query order status
  - Greatest value
    - Encapsulates the complexity of business rules
Service Data Ownership

Where's the Service Boundary?
• States are business process milestones
  • May be composite states for reporting

• Business rules govern transitions
  • Can’t cancel an order that has been shipped!

Service Business Rules

- placeOrder
- Unshipped Order
- Partially Shipped Order
- Shipped Order
- Order Cancelled
- orderItemsShipped[unshipped items remain]
- orderItemsShipped[all shipped]
- cancelOrder
Technical Challenges

- **Data ownership and management**
  - Which concepts are owned by the service, which are external?
  - Which relationships are owned by the service, which are external?
  - How do we manage relationships that cross the service boundary
    - E.g. what happens to existing orders if a catalog item is deleted?

- **Data representations at service boundaries**
  - Common data models

- **Access technology**
  - Mechanics of how will the service be accessed
    - E.g. SOAP over HTTP or JMS

- **Supporting infrastructure services**
  - Communications and messaging
  - Access control: authentication, authorization, and encryption
Typical Service Operation Architecture

**Traditional Object/Component Approach**

- Native semantics for operation and data
- Native technology for operation and data

**Service Approach**

- Native semantics for operation and data
- Native technology for operation and data

- Some level of standardization
  - Technology of access
  - Data semantics
  - Operation semantics

**Using Component**

- Native Interface
- Provider of Functionality

**Using Component**

- Service Interface
- Native Interface
- Provider of Functionality
Where Do Services Make Sense?

- When there is functionality that is either used in more than one place or is provided in more than one place, particularly when those “places” are different applications.
Data Normalization

- Decision must be made whether to use system-neutral data format in communications

  - Direct transformation (no neutral data format)
    - One component usually sends, the other listens
      - Requires $n-1$ transformation definitions
    - Requires $n-1$ transformation run-time executions per message

  - Neutral data format
    - Requires $n$ transformation definitions where $n$ is the number of different types of end-points using the message
    - Requires $n$ transformation runtime executions per message
Understanding Data Normalization Tradeoffs

- Direct transformation always requires fewer runtime transformations
  - N-1 transformations

- So why would you want to use a neutral data format?
  - Replace source system without changing the mappings
  - Makes mapping easier
  - Makes it more accessible (using XML for example)
Determining Data Normalization Policies

- Selecting a policy is a tradeoff between:
  - Implementation cost
    - Number of transforms required
    - Complexity of transforms
  - Run-time processing power
    - Number of transforms executed
    - Complexity of transformation
  - Network bandwidth
    - Number of messages appearing on the network
    - Size of messages
  - Cost of evolving data structures
    - Development cost
    - Deployment complexity
    - Maintenance Costs
3 Styles of Service Coordination

- **On-Demand**
  - Service waits for requestor to invoke an interface and then initiates the requested action

- **Event-Driven**
  - Upon receipt of an event, the local service performs its required function
  - Service proactively notifies subscribers when specific events occur

- **Continuous**
  - Service that runs on its own without formal invocation either periodically or continuously
Services can exist at many levels of abstraction, but generally they can be broken into four broad categories:

- Infrastructure Services
- Point-to-Point Services
- Business Services
- Composite Business Services
Infrastructure Services

- Services for use by technologists!
- Building blocks that provide commonly required infrastructure in a standardized way
- Exposing infrastructure as services significantly reduces the level of effort required to build higher level services
- Common infrastructure services include:
  - Messaging Services
  - Event Services
  - Audit and Logging Services
  - Error Notification Services
  - Security Services
  - Portal Services
Complete Uniformity is Not Always Possible

- Sometimes a re-usable component (library) needs to be provided in the user’s technology and embedded

- Examples:
  - Local interface for error logging
  - Security access
**Point-to-Point Services**

- Point-to-Point Services standardize the *technology* used to access operations and represent data.

- Point-to-Point services do not standardize the *semantics* of the operation or the data.

```
Using Component

Service Interface

Data or Application Service

Native Interface

Provider of Functionality

- Native semantics for operation and data
- Standard technology for operation and data

- Native semantics for operation and data
- Native technology for operation and data
```
Business Services

- Business services standardize both the semantics and the access technology
- The standardization greatly simplifies the reuse of the functionality in many contexts
- This standardization also makes it easier to construct or modify composite business services
Composite Business Service

- Composite business services orchestrate the use of other services to create a new higher-level service
- A composite business service may be a complete business process, giving us Business Process Management
  - BPM and Business Works can be viewed as tools for creating composite business services
- Composite business services make possible the overall management of the encapsulated business process including monitoring and error reporting
ATM Example Services

- Withdraw Cash
- Obtain Disbursal Authorization Service
- Transfer Funds
- Report Funds Delivered Service
- Make Deposit
- Deposit Funds Service

Bank Customer

© 2006 TIBCO Software Inc. All Rights Reserved.
- The disadvantage of this approach is that you lose the big picture of how all the components interact to carry out the function.
Scenario Showing Service Design in Context

Customer: Person
- insert card and enter PIN
  (card data, PIN)
- validate PIN
- prompt for transaction
  (prompt)
- prompt for amount
  (prompt for amount)
- invoke obtain disbursal authorization
  (amount)
- dispense cash
  (cash)
- remove cash
  (removal notice)
- remove card and receipt
  (card, receipt)
- determine bank and forward
  (forwarded request)
- grant disbursal authorization
  (disbursal authorization)
- record withdrawal transaction
  (notification acknowledgement)
- forward request
  (forwarded notification)
- notification acknowledgment

ATM Machine
- select "Withdraw Cash"
  (selected transaction)
- enter amount
  (amount)
- invoke obtain disbursal authorization
- dispense cash
- remove cash
- remove card and receipt

ATM Server
- determine bank and forward
  (disbursal authorization)

Bank
Mindset Issues

- **Services are not about technology**
  - Services are about cost-effectiveness
  - Focus should be on what reusable functionality is needed
  - Technology issues are secondary

- **Every interface isn’t a service!**
  - Services involve overhead, both at design and run-time
    - Adapters, mappings, authentication, authorization, encryption, etc.
    - Granularity of work must outweigh the overhead
  - Must demonstrate potential for reusability (common need)
    - Identify the multiple users of the service
    - Make sure that the functionality is, indeed, the same!

- **There’s more to using services than just orchestration**
  - BPEL assumes all functionality is encapsulated as a web service
    - Exposing functionality as a service forces access control for critical functions
  - Real business processes require non-service functionality as well
    - Data structure transformation, complex condition evaluation
The Reusability Challenge

- **How do we design for future usages?**
  - Today we enter orders in person, via paper, by phone, on-line, …
    - What’s next – via Blackberry? Automatic re-order?
  - Your CPG firm decides to sell branded clothing as a promotion!
    - Orders now need sizes, colors, etc.

- **Insight is required when conceptualizing a service**
  - What might change in the future?
    - Evolutionary changes – organic growth
    - Revolutionary changes – buying your biggest competitor, new markets
  - How do these changes challenge existing functionality?
  - Which alternatives are worth investing in?

*Who can provide this insight in your organization?*
What’s Driving SOA?

Source: InfoWorld Research Report: Service Oriented Architecture (SOA), March 2006

Q: What are the business problems your company hopes to address using SOA?
Base: 521 (Among qualified respondents)
These Pressures Require Multi-Silo Responses

- Lack of Overall Responsibility
- Services and Integrations that Span Silos
- Shrinking Time Frames

Diagram showing various application silos, service interfaces, and data center connections.
No Cross-Silo Ownership in Current Organizational Structures

Who owns projects that span silos?

Business Executive Sponsor

IT Executive Sponsor

Business Manager

IT System Owner

Business Manager

IT System Owner

Business Manager

IT System Owner

Business Manager

IT System Owner

Application Silo

Application Silo

Services, Integration, and Process Management Silo

Application Silo

External Applications

Data Center

Communications and Services Infrastructure

Front-Office Applications
Services Span Both Silos AND Projects!

Project 1

Project 2

Project 3
Other Potential SOA Risks

- **Services will not be re-used**
  - Technical or business design not suitable
  - Potential users unaware of existing services
  - Lack of governance to ensure reuse

- **Service development will be difficult**
  - Lack of support for accessing services in different end-point system technologies
  - Lack of support for events as well as request-reply

- **Services will be hard to manage**
  - Inconsistent implementation technologies – too many variations create complexity
  - Inconsistent design and utilization patterns
    - FT, HA, load distribution
    - Security
Typical Client-Server Development

Gatekeeper

Requirements → Development → QA → Production
Distributed Systems Development

- Quantified business benefits
- Cost and schedule constraints
- Business risks

Missing Steps (and Governance!)
Multi-Silo Project Organization

- Business Executive Sponsor
- Project Manager
  - Business Process Architect
  - Systems Architect
  - IT Executive Sponsor

Silo 1
- Business Manager
  - Project Sub-Team: Team Member, Team Member, Team Member
- IT Manager

Silo 2
- Business Manager
  - Project Sub-Team: Team Member, Team Member, Team Member
- IT Manager

Silo n
- Business Manager
  - Project Sub-Team: Team Member, Team Member, Team Member
- IT Manager
Multiple Projects Require Time for Oversight!

- **Business Executive Sponsor**
  - Project Manager
  - Business Process Architect
  - Systems Architect

- **IT Executive Sponsor**

- **Silo 1**
  - Business Manager
  - IT Manager
  - Project Sub-Team
    - Team Member
    - Team Member
    - Team Member

- **Silo 2**
  - Business Manager
  - IT Manager
  - Project Sub-Team
    - Team Member
    - Team Member
    - Team Member

- **Silo n**
  - Business Manager
  - IT Manager
  - Project Sub-Team
    - Team Member
    - Team Member
    - Team Member
Enterprise Projects Organization

- Enterprise Projects Manager
  - Manages silo-spanning projects
    - Provides day-to-day oversight
  - Reports to the Business Executive Sponsor (directly or indirectly)

Business Executive Sponsor

Enterprise Projects

- Project Manager
  - Systems Architect
    - Business Process Architect
- Project Manager
  - Business Process Architect
  - Systems Architect
- Project Manager
  - Business Process Architect
  - Systems Architect
But Services Span Projects – and Involve Business!
Enterprise Architecture Organization

- **Enterprise Architecture Responsibilities**
  - Business Process Architecture
  - Architecture for infrastructure and applications
    - Design patterns
    - Best practices
  - Data Architecture
  - Service validation and specification
  - Architecture to support operations
    - Component monitoring
    - Process monitoring
Completed Organizational Picture
Questions?