SOA Concepts – British Computer Society Advanced Programming Group

James Allerton-Austin
EMEA Fusion Middleware SOA Product Manager
The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Recommended Reading

  Authors Davies, J; Schorow, D; Ray, S; Rieber, D

Agenda

- What is SOA?
- Separating concerns
- What is a Service?
- Beyond the Bus Service
- Summary
- Q&A
Service-Oriented Architecture (SOA)

A Business Lead IT Strategy that enables the creation of loosely coupled, interoperable business services that can be easily shared within and between enterprises to meet and adapt to business need.
What is SOA?

Key areas

- Run-time re-use
- Discrete Business Services
- Reliable Infrastructure
- Service Lifecycle
Service Component Architecture
Composites, not monolithic applications

- Standard framework for “wiring” services
- Networks of services, provide capability
- Combining code and configuration
- Capability orchestrated from:
  - Service Enabled applications
  - Coded services
  - “Wiring” configuration
Agenda

• What is SOA?
• Separating concerns
• What is a Service?
• Beyond the Bus Service
• Summary
• Q&A
Separating concerns
Layering and separating responsibility

- Service Enablement
- Service Virtualisation
- Mediation
  - Translation
  - Transformation
  - Routing
  - Security
- Canonical Data

*That doesn’t make sense to program again.*
Separating concerns
Catching the Bus

- Delivered with a modern Service Bus
- Features delivered:
  - Service Virtualisation
  - Mediation
  - Transformation
  - Service Brokering
  - Service Aggregation
  - Security
  - Monitoring
  - Load balancing
  - Configuration, rather than coding
The Service Bus
Delivering Service Virtualisation

- Service Clients
  - Application Client

- Service Bus
  - Proxy Service
  - Business Service

- Enterprise Services
  - Service

- Inbound interface
- Transformation, etc.
- Routing

- Outbound interface
- Load balancing

transport
Separation of Concerns
Adapter based Integration
Separation of Concerns
Service Brokering

Service Bus
- Message Brokering
  - Multiple Messaging Protocols
- Multiple Data Formats
- Multiple Communications Paradigms

Service Clients
- Application Client
- Application Client
- Application Client
- Application Client
- Application Client

Enterprise Services
- Service
- Service
- Service

Communication Protocols:
- HTTP/SOAP
- RMI
- JMS
- FTP
- SMTP
- File
- Tux
Separation of Concerns
Message transformation
Separation of Concerns
Mediation and Routing
Separation of concerns
Monitoring Service Health
Separation of Concerns
Alerting SLA violations
Separation of Concerns
Error Handling
Separation of Concerns
Load balancing and dynamic service resolution
Separation of Concerns
Securing heterogeneous services
Oracle Service Bus

Key Ingredients for Enterprise Service Bus

Service Management
- Monitoring
- SLA Alerts
- Reporting

Service Virtualization
- Content Based Routing
- Transformation
- Service Chaining

Configuration Framework
- Change Center
- Validation
- Import / Export
- Metadata Store

Unified Security
- Authentication
- Authorization
- Identity
- Sign/Encrypt

Adaptive Messaging
- HTTP/S
- JCA
- JMS
- WSRM
- REST
- MQ
- SMTP
- FTP
- File
- Tux
- EJB
Agenda

- What is SOA?
- Separating concerns
- What is a Service?
- Beyond the Bus Service
- Summary
- Q&A
What is a Service?

- Service Definition
  - Determines the service boundaries.
  - Defines service contract.
  - Identifies service classification and scope for the service.
Benefits of Service Boundaries

Programming best practice

- Isolation
  - Enables services to physically scale or technologically evolve independently
  - Enables parallel development
- Encapsulation
  - Prevents unwanted side-effects since the service is self-contained
  - Enables the service implementation to evolve without impacting users
- Standardised interfaces
  - Enables quick composition of services within a process or service bus
  - Enables standardised service and business activity monitoring
- Managed dependencies
  - Enables effective impact analysis since dependencies can be registered, tracked, and managed
- Reduced system complexity
  - Prevents “hidden” dependencies that would otherwise complicate the delivery of requirements.
Service Design

- Service Design
  - Defines the service interface including message schemas
  - Determines the implementation approach
    - Java, .Net, composition, etc.
  - Determines the underlying resources used by the service
  - Covers both functional and non-functional requirements
What defines a Service?

Three key aspects

**CONTRACT**
- **INTERNAL USER**
  - Purpose
  - Policies
  - Operations/Exceptions
  - Semantics
  - Protocol, QoS
  - Invocation Style
  - Pre/Post Conditions
- **EXTERNAL USER**
  - Security++
- **INTERNAL PROVIDER**
  - Invocation Style

**INTERFACE**
- **WS-Policy**
- **Asynchronous**
- **SOAP**

**IMPLEMENTATION**
- **C++, Java, COBOL, ...**
- **SAP, PeopleSoft, ...**
- **J2EE, .NET**
- **...**

**WS-ReliableMessaging**

**WS-Security**
Shared Service Relationships
For a single service

- Service
  - Describes
  - Bound To

- Contract
  - Fulfills

- Implementation
  - Implements
  - Provides Functional Implementation
  - Provides Service Enablement

- Interface
  - Exposes

- Development Platform
- Service Infrastructure
Service Versioning
Service lifecycle

- Evolving services requires a service versioning strategy
  - Major Release
    - Breaks backward compatibility
    - If common indicates service identification is not working
  - Minor Release
    - New capability with backward compatibility
    - Adding new methods, not changing existing methods
  - Point Release
    - No new features
    - Bug fix or performance enhancement
- Service testing strategy is key to successful service evolution
  - Full service regression test for each release
  - Consumers retest for major and minor releases
Service Versioning Principles

- Architecture should support several versions of a service being in production concurrently
  - Governance should limit the number of concurrent versions
  - Advantage of using a Service Bus to route requests
  - Consider adding a consistent message header for version information
- Consumers should not be forced to upgrade immediately
  - Consumers must migrate eventually
  - Services retired gracefully
- Consumers must be able to determine a suitable version
  - Advantage of using registry/repository for service lookup
- Services should strive for backward compatibility whenever possible
  - Point releases should never break backward compatibility
Versioning & Service Deprecation

The service lifecycle

Service (2.0.0)
Operational/Current

Service (1.1.0)
Operational/Deprecated

Service (1.0.1)
Retired

Contract
V2.0.0

Implementation
V2.0.0

Interface
V2.0.0

Contract
V1.1.0

Implementation
V1.1.0

Interface
V1.0.0

Contract
V1.0.0

Implementation
V1.0.1

Implementation
V1.0.0

Implementation
V1.0.0

Implementation
V1.0.0
Using a Service

Coming back to the Service Bus

Service Registry

Service

Service Bus

Composite Application

Development

Invokes

Orchestrates

Contains

Describes

Fulfills

Implements

Contains

Describes
Agenda

• What is SOA?
• Separating concerns
• What is a Service?
• Beyond the Bus Service…
• Summary
• Q&A
Components of a SOA Infrastructure
There’s more than a Bus!

Build/re-use Services
Web, Legacy or ext. Services

Connect
Adapters & B2B

Route, Transform, Virtualise
ESB

Orchestrate
BPEL/BPM process engine

Externalize business rules
Rules Engine

Involve human beings
Human Workflow

Secure
Global Security Framework

Visualize
Business Activity Monitoring

Manage & Govern
Enterprise Manager,
Repository & Registry

ORACLE
Service Orchestration

Building a service

- *Declarative orchestration to rapidly derive new value from existing services & assets:*
  
  - Systems
  - People
  - Documents

- Separates the business process definition from the implementation
  
  - Built-in asynchronous support
  
  - Long running processes
  
  - Standards based
Tracking assets and services...
Asset visibility
Service Oriented Security
Securing assets in a service network
SOA Composite Editor
How Oracle helps you pull it all together

- Mediator
- Orchestration
- Human Tasks
- Spring / Java
- Business Rules
- Web Services and Adapters
- Palette of Service Components & Adaptors

Oracle
Agenda

- What is SOA?
- Separating concerns
- What is a Service?
- Beyond Services
- Summary
- Q&A
Summary

SOA Challenges

- Integration to business systems
- Building effectively for quicker, timely delivery
- Achieving performance levels
- Providing compliance and regulatory data
- Securing services in SOA
- Admin and Management of orchestrated apps

Proving ROI of design and runtime assets
Why Oracle?
SOA Suite 11g

Complete
Comprehensive SOA Offering

Integrated
Designed to Work Together

Open
Standards-Based Architecture

Best of Breed
Industry Leader

Broad & Deep Offering
Lower Cost, Lower Risk

Less Effort
More Value

More Choice
Maximizes Existing Investments

Robust, Proven Industry leading Software
Agenda

• What is SOA?
• Separating concerns
• What is a Service?
• Beyond Services
• Summary
• Q&A
Useful links

- Mastering SOA Series -
- Service Component Architecture -
- Building Services with BPEL -
- New SOA Standards -
Thank you