# Introduction to Service-Oriented Architecture

Michael Stiefel
www.reliablesoftware.com
development@reliablesoftware.com

## Goal

Understand Service Oriented Architecture (SOA)

# Why?

Number Months To Revamp:

Sales

Marketing

Change

Buildings

=

Number Years To Revamp IT

## Fundamental Principle

Service Oriented Architecture driven by business, not technical needs.

## Observation #1

***A SOA is architected, not bought…***

Contrary to the view of many vendors

## Observation #2

***It is a journey, not a destination…***

About approaches and principles, not fixed technical approaches or specific implementation technologies

## Observation #3

***Focus on reuse, agility, integration, standards interoperability …***

Leads to fuzziness that makes people confused, wary and unhappy.

# Architecture

Study of the principles of design, construction, and esthetics of buildings

The profession of designing, constructing, and ornamenting buildings

The structure and organization of a particular building or class of buildings

***The confusion is that SOA is about the <u>first</u> definition not the last.***

## SOA is an Architectural Style

Principles vary over historical periods

Principles vary over the artifacts constructed

## Modern Business Environment

Cannot rebuild every application from scratch

Customers’ demands change quickly over time

Depend on vendors and suppliers

***SOA <u>is</u> <u>not</u> about building an application that meets specific business functionality or goals such as ROI or cost effectiveness.***

***SOA is about the principles of constructing loosely coupled, reusable, application-agnostic business services.***

# Sandwich Shop Parable

Carnegie DELICATESSEN
RESTAURANT
WE MAKE OUR OWN PASTRIES, MEATS, PICKLES PROMPT DELIVERIES 757-22
854

Place Order

Pay For Order

Make Sandwich

Deliver Sandwich

## Parable Implication #1

The Sandwich shop has capabilities.

Customers have needs.

A ***service*** allows ***needs*** and ***capabilities*** to interact.

The ***service provider*** provides the ***service***.

***Service consumer*** uses the ***service***.

## Parable Implication #2

For capabilities and needs to be met you need:

***Visibility***

***Interaction***

***Real World Effect***

## Parable Implication #3

***Real World Effect*** is about business behavior, not programming constructs or objects

Behaviors cross <u>trust boundaries</u>

<u>Marketplace</u> of interactions

SOA is different from other distributed architectural paradigms

## Parable Implication #4

In order for the participants to interact, they have to agree on common terminology, or ***semantics***, for the ***interaction***

## Parable Implication #5

This ***interaction*** occurs within an ***Execution Context***

- Contract
- Policy
- Behavior Model
- Information Model

## Parable Implication #6

Focusing on behaviors leads to more scalable systems

Focusing on behaviors leads to a flexible IT portfolio

## Benefits

Scalable Paradigm

Encourages Agility

Encourages Interoperability, Standards

Makes Explicit Ownership Boundaries

## OASIS Reference Model

Parable implications are defined as a vocabulary in a *Reference Model for Service Oriented Architectures*

Developed under the auspices of the *OASIS* standards organization

Current Committee Draft is open for general comment

## SOA

SOA is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains

SOA is a framework for meeting needs with some combination of capabilities

## Sandwich Services

Sandwich Shop is the Service Provider

Customers are the Service Consumers

Service interface defines

- service interaction
- explicit boundary for code and data

Separate independent, reusable capabilities

## Service Description

***Service Interface***

***Policy***

***Information Model***

***Behavior Model***

Provides ***Visibility***

Defines ***Real World Effect***

Defines ***Reachability***

## Service Policy

- Expression of service requirements
- Constraints for message processing
  - Guides analysis of business activities
  - Establishes parameters of activities
- ***Service consumers*** must understand policy independent of making a request

## Semantic Engagement

***Service Consumer*** and ***Provider*** have to understand each other

Clear Semantics

Information Model

Behavior Model

Understand ***Real World Effects***

## Execution Context

- Policy Decision Point
- Contract Enforcement
- Common Infrastructure
  - Messaging
- Different Instances, Different Contexts
- Third Parties
  - Government

## Discovery

***Service Consumer*** must be made ***aware*** of the existence of the ***Service Provider***

- "Ask Alice"
- Advertising
- Registry

## Sandwich Contract

- Ordered sequence of versioned messages that represent business documents
- Series of asynchronous declarative messages
  - Request sandwich, no immediate response
  - Sandwich ready message arrives with no request

## Document Centered

- Exchange Documents/Messages
  - No access of remote objects
  - Contract: content, order of messages
  - Can contain their own state
- No instructions to sandwich maker in request
  - No programmatic types, just business description

## Sandwich Service Interactions

Payment service independent of sandwich making

Long lived interactions / transactions

No sharing of dynamic data such as bread inventory

Can query static data, sandwich composition

## Loose Coupling

Several independent services complete "order"

Dynamic substitution possible

More fault tolerant with multiple revisions

Easier to change business process

Engineering Tradeoff

## *Services are bound at execution, not compilation*

## Sandwich Policy

Credit Cards accepted; No personal checks
Store hours are from 7 AM to 6:30 PM
Take phone orders except during lunch time
2 minutes from order to sandwich ready
30 minute delivery time or its free

***Can query policy independently of going to the store or placing an order***

## Service Transactions

- Real world is asynchronous, occasionally connected
  - Two phase commit is hard enough
- Compensating transactions
  - Credit card charge backs
  - Removing orders from queue
- Better scalability

## Versioning

- Change a contract, policy, or schema of a document
- No recompilation
  - Existing contracts still work, unlike execution environment interfaces
  - Just reject unknown documents, or unsupported versions

***Service reuse is likely to succeed where object reuse failed, because services are loosely coupled, and based on actual business activities, not on programmatic abstractions.***

## No UI in SOA

UI is application development

Composite applications use services

- Services will use other services

Allows services to be used from web, mobile device, fax, phone call, or in-person visit

## Policy and Applications

- Application must be able to query policy to validate message before sending it
  - Design or Runtime
- Service will still need to check against policy

## Agility

Reusing services allows:

- quicker response to new requirements
- reduces the time needed to compose new applications making customers happier
- decrease the cost of composing new business applications
- reduces vendor lock-in and /or switching costs if services can be easily replaced

## Interoperability / Integration

Interoperate with third party services

Integrate legacy applications as reusable services

## Object Oriented Evolution

- Reuse through Inheritance
  - Fragile Base Class Problem
  - “Inheritance Breaks Encapsulation”
- Reuse with Interface and Composition
- Reuse through Templates
- Dynamic Interface, Behavior Reuse with Contracts

# Service vs. Object

| Services | Objects |
|---|---|
| Not new, but not familiar | Well-understood |
| Heterogeneous Environment | Single Execution Environment |
| Schema | Type |
| Addressing | Object References / Addresses |
| Latency, Partial Failure, Concurrency | Assume fast, transparent network |
| Loose Coupling / Messaging | Linkers and Loaders |
| Execution Time Binding | Compile / Link Time Binding |
| Security Throughout | Usually at the Boundaries |
| Model Real World Processes | Model Programming Constructs |

See Waldo, et. al. *A Note on Distributed Computing*

## Business Process

Several services may be "orchestrated" together

- Human intervention part of process
- Activities can transcend business unit boundaries

## Home Mortgage Process

To obtain a home mortgage:

- Process Loan Application
- Do Credit Check
- Decide Credit Worthiness
- Do Home Appraisal
- Decide Loan Viability

## How to Find the Services

- Focus on Stable Business Capabilities
  - What will your business continue to do?
- Example: Mortgage Loan Service
  - Loan Origination for Various Products
  - Credit Check
  - Loan Scoring, Rating and Approval
  - Loan Servicing

## Implementation Technologies

- WS-Lite
  - XML, SOAP, WSDL
- WS-Heavy
  - WS-Lite + WS-Security, WS-Addressing, etc.
- Representational State Transfer (REST)
- XML over HTTP (POX)

# Transportation Parable

- Implications of large scale
- Driven by Economics and Social Factors
- Evolves over time
- Business, people => independent, loosely coupled services
- Complex interconnection of business, vendors, suppliers =>composite applications

## Enterprise Services

Manufacturers, service providers, people => services

Business is done via “contracts”, explicit or implicit

## Business Processes

- “Composite applications” as various services work together
  - Accounting, billing, shipping, receiving, sales, production
  - These services may be internal or outsourced
- Loose coupling => use various suppliers, vendors, employees
- Business is “orchestrated” together

## Open Standards Messaging

- Travel by cars, trucks, buses, trains, planes => messages
- Traffic Laws
- Legal Regulations
- Car, Plane standards
- Mixture of proprietary and “open” standards

# Enterprise Policy

- Policy set by businesses and individuals
  - Credit Terms, etc.
- “Public Policy”
  - Speed Limits
  - Signage
  - Enforcement

## Implications of the Parable…

## Continued Investment

Who decides:

- Where do the new roads, track, or airports go?
- Which are maintained at what level?
- Which are upgraded to new standards?
- How is this paid for?

## Infrastructure

Police for roads, maintenance crews, fire departments, traffic departments

Standard road signs,

Determining speed limits

Road design standards (curve angles for speeds), merge lanes

Accident control, emergency services

Financing of improvements

Toll collection (if any)

***Governance is necessary to make this work…***

## IT Governance

Who makes maintenance and infrastructure decisions?

Who pays for the common infrastructure?

Who makes sure it stays shared?

How do the monies get distributed?

Business units are judged by ROI

Sarbanes-Oxley, HIPAA Compliance

## SOA Pioneers

Credit Suisse First Boston

Deutsche Post

Dell Supply Chain

Disney

EBay

Amazon

## Rearden Commerce

- Employee Business Services (EBS)
  - B2B non PO services Marketplace
    - Travel Services, Conferencing, Small Package Delivery
- Corporate Customers:
  - Motorola, Whirlpool, Warner Home Video
  - HP, American Express will resell EBS
- Brower-based applications can run on desktop or intelligent cell phone

## Employee Business Services

- Customers and Suppliers can build applications based on their own business activities and identity management rules
- Reardon has defined flexible schemas for policy, service orchestration, and identity management
- Tied together with web services
- Graphical application for end users to create composite applications

## SOA vs. CORBA, DCOM, EDI

Designed to simplify construction of distributed systems.

Not interoperable

Based on static programming calls with fixed application signatures

Often based on objects that were not a clear match to business semantics

SOA can support business documentation standards ANSI X12, ebXML, EDI, HL7, ICE9.

# Web Services

- Web Services
  - Independent of Execution Environment
  - Loosely Coupled integration
  - Open Standards (WS*)
- Contract first development
  - XML and XML Schema for defining service messages
  - SOAP for message transport
  - WSDL for contract definition
- Orchestrate services into business processes

# Wonder City Metropolitan Area Medical System Case Study

## Current Problems

- Patient information is incorrect, missing, or at another location
- Cannot find specialists quickly for difficult diagnoses
- Inefficient allocation of resources
  - laboratories, nurses, operating rooms, blood, supplies, surgical equipment, rooms, etc.
- Patient information is not secure

## Consequences

Patients are misdiagnosed

Critical tasks are not correctly scheduled

Wastes time and money leading to higher medical costs

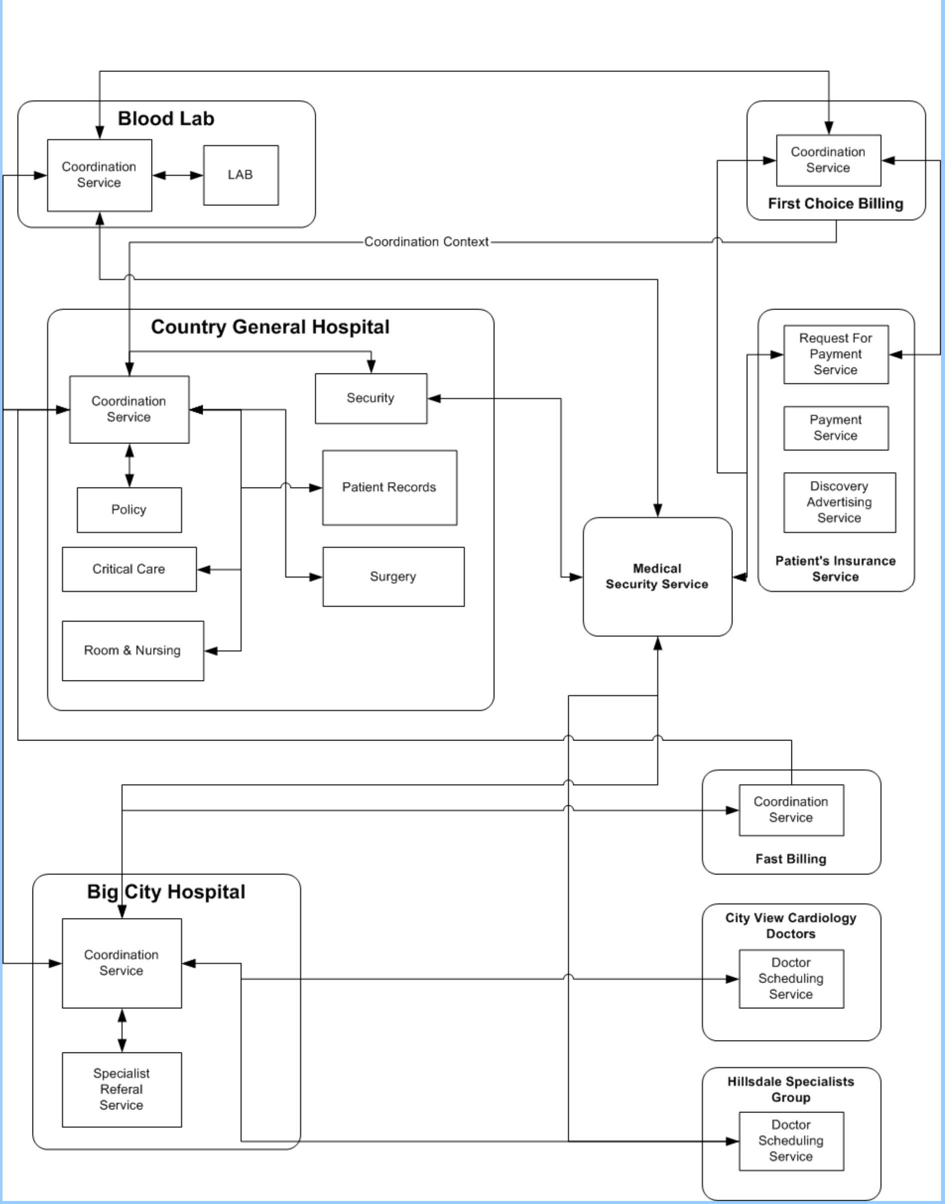
Lower quality of life for staff and patients

HIPAA compliance is difficult

## Use Case

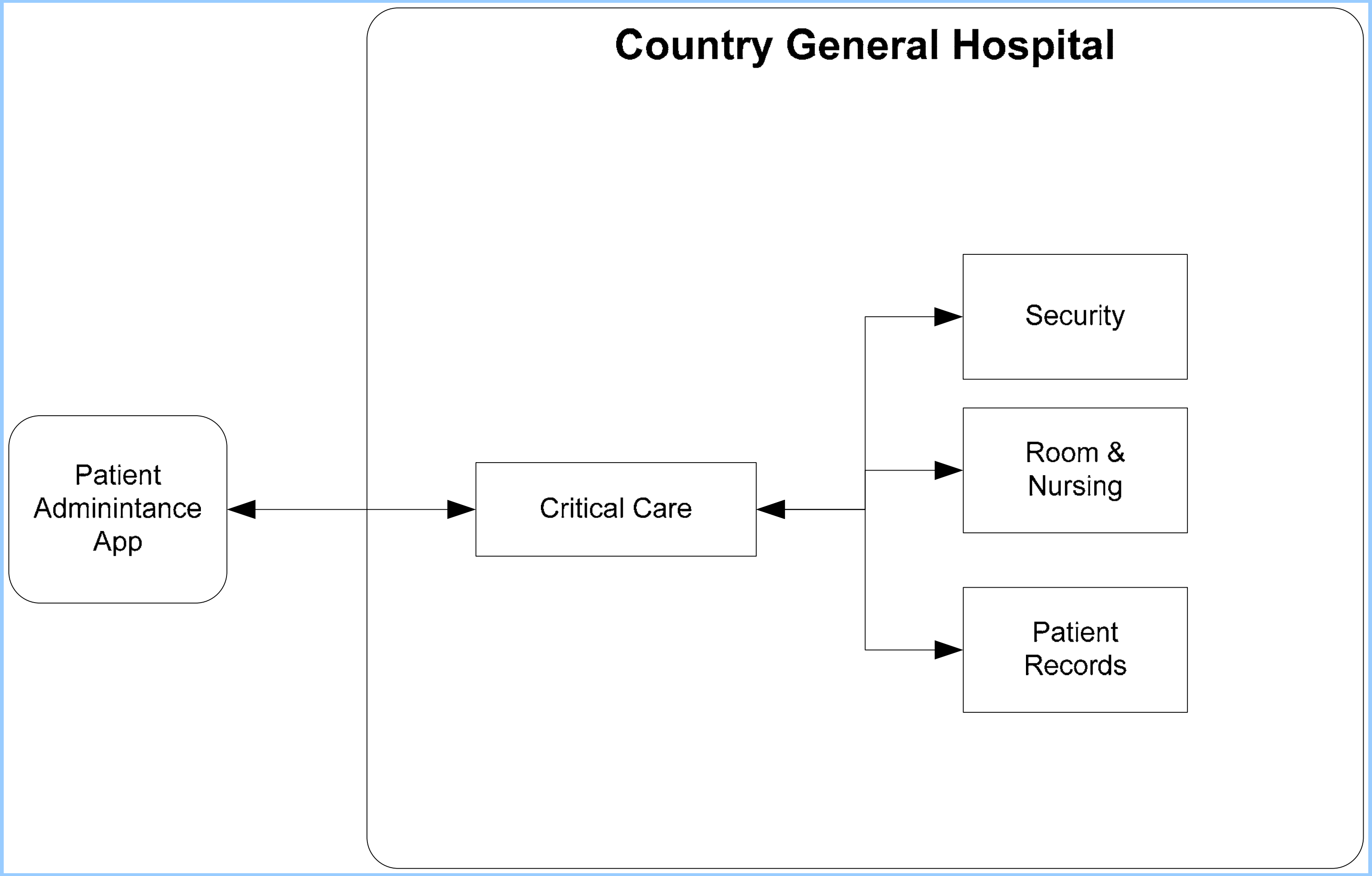
- Patient is critical, but stable
- Can patient be admitted to the hospital?
    - Does hospital have a spare room?
- Does patient need surgery?
    - Run tests
    - Consult with other doctors

## Critical Care Scenario

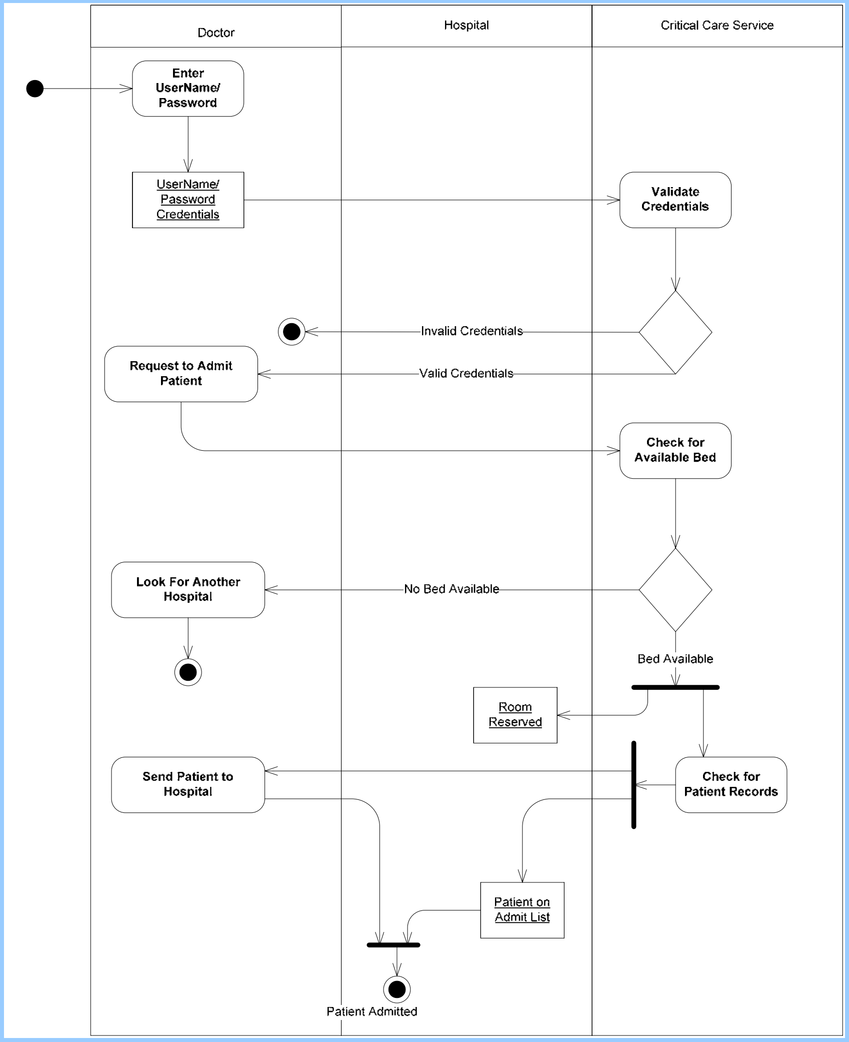


## What are the Business Services?

# Critical Care Service



# Flow Diagram



# Simple SOAP Message

```
<PatientAdmission xmlns="urn:CriticalCarePatient" >
    <Patient>
            <Last>Piper</Last>
            <First>Peter</First>
    </Patient>
    <Address>
            <Street>123 Hampshire</Street>
            <City>Cambridge</City>
            <State>MA</State>
            <Zip>02139</Zip>
    </Address>
    <SSN>111-22-3333</SSN>
    <Diagnosis>massive heart palpitations</Diagnosis>
</PatientAdmission>
```

# X-Ray Attachment

```
MIME-Version: 1.0
Content-type:Multipart/Related;boundary=MIME_boundary;
  type="application/xop+xml"
  start="<doctorjones@bighospital.com>";
  startinfo="application/soap+xml; action=\"ProcessXRay\""
Content-description: SOAP response to x-ray request
--MIME_boundary
Content-type:application/xop+xml;
   charset="UTF-8";
   type="application/soap+xml; action=\"ProcessXRay\""
Content-Transfer-Encoding: 8bit
Content-ID: <doctorjones@bighospital.com>
<env:Envelope
           xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
           xmlns:xmlmime="http://www.w3.org/2004/06/xmlmime"
           xmlns:xop="http://www.w3.org/2004/08/xop/include">
   <env:Header>
      <wsa:To>http://www.bighospital.com/PatientAdmissionXRay</wsa:To>
   </env:Header>
  <env:Body>
      <xray:xraytransmit xmlns:xray="bighospital/xray.xsd">
         <xray:content xmlmime:contentType="application/octet-stream">
            <xop:Include href="cid: doctorjones /patient123xray1.zip"/>
         </xray:content>
      </xray: xraytransmit >
   </env:Body>
```

## Web Service Definition

Types in XML Schema

Services in WSDL

# SOAP Addressing

```
<env:Envelope xmlns:env=”http://w3.org/2003/05/soap-envelope”
                          xmlns:wsa=”http://schemas.xmlsoap.org/ws/2004/03/addressing”>
   <env:Header>
       <wsa:To:>
                  <wsa:Address>http://countygeneralhospital/criticalcare </wsa:Address>
</wsa:To>
<wsa:From>
<wsa:Address>http://doctorjones</wsa:Address>
</wsa:From>
<wsa:ReplyTo>
<wsa:Address>http://doctorjones</wsa:Address>
</wsa:ReplyTo>
<wsa:FaultTo>
<wsa:Address>http://doctorjones</wsa:Address>
</wsa:FaultTo>
       <wsa:Action>urn:PatientAdmittanceRequest</wsa:Action>
         <wsa:MessageID>
            uuid:12345678-1234-5678-123456789012
         </wsa:MessageID>
         <wsa:RelatesTo RelationshipType=”Reply”>
            uuid:12345678-1234-5678-123456789012
        </wsa:RelatesTo>
   </env:Header>
   <env:Body>
<PatientAdmission xmlns="urn:CriticalCarePatient" >
       …
   </env:Body>
</env:Envelope>
```

## Message Exchange Patterns

Request / No Response
Request / Response
Request with Optional Response
Notification
Notification with Acknowledgement
Notification with Optional Acknowledgement
Broadcast

## Reliable Messaging

Messages are delivered at most once without duplication, it is possible that some messages may not be delivered.

Messages are delivered at least once, some messages may be delivered more than once.

Messages are delivered without duplication. This is the logical “and” of the first two assurances.

Messages  are delivered in the same order they were transmitted. This assurance can be combined with any of the previous three assurances.

## Policy

WSDL does not express constraints on a Web service

WS-Policy provides such a framework

Policy Assertions

Policy Alternatives

Collections of Policy Alternatives

## Metadata

WS-MetadataExchange defines how to query a service to find out its metadata

Service semantics are not expressed in either WSDL or WS-Policy statements.

## Sample Policies

Doctor has to have admitting privileges to hospital

Patient information has to be encrypted when transmitted

## Message Security

SSL only secures point-to-point

With more than one recipient or transport layer, you need end-to-end security

## Message Routing

Authentication service

Bed availability service

    information about patient condition

Patient record check service

    should not know about patient condition

## Securing SOAP Messages

- Message signing
  - integrity
  - non-repudiation
- Message encryption
- Message authentication
- WS-Security
- WS-Security Policy

## Trust

Security tokens have to come from a trusted source.

WS-Trust defines protocols

- issuing
- requesting
- renewing
- validating
- transmitting
- how to establish trust between two parties

## Federated Identity

- Identities are valid only within a trust domain
- WS-Federation
    - how trust works between two domains
    - based on WS-Trust
    - identity, authentication, authorization shared
- Single sign on and sign off possible.
- Avoid creating identities in both domains

## Transactions

- WS-Atomic Transaction
  - classic ACID Transactions
- WS-Coordination
  - compensation model

## Summary

SOA is independent of technology

Covered basic principles of SOA

Defined Basic SOA Vocabulary

First look at implementing SOA with Web services