



Deploying Axis in Mission-Critical Environments

Eugene Ciurana

eugenex@walmart.com

<http://eugeneciurana.com>



When should I suggest web services?

- **When an application involves two or more systems:**
 - Different geographical locations
 - Different programming languages
 - Different machine architectures and data formats
- **Interactions between those systems are stateless**
 - Better scalability by implementing zero intermediate steps
 - No system behaviour is prescribed by the messages
- **Messages are constrained by an agreed schema**
- **Interfaces are universally available to all consumers and servers**



Common web services architectures

- **Two general types of web services**
 - REST
 - SOAP
- **REST (Representational State Transfer)**
 - Based on the concept of web resources
 - A resource is anything that has zero or more representations addressable through a URI
 - Simple messages can be encoded in the URI itself
 - Other messages are in XML, normalized through the use of XML Schema or any other schema
 - All interfaces use HTTP (GET, POST, PUT, DELETE)



Common web services architectures

● SOAP (Simple Object Access Protocol)

- RPC
- Web services

● SOAP RPC

- Used for tunneling application semantics and system behaviour through a generic interface
- RPC applications are not interoperable -- tend to become "custom cases"

● SOAP web services

- Document, not behaviour, specific
- Provide an envelope that contains the service semantics in an interoperable manner
- Use WSDL (web service definition language)
- All messages should be encoded by SOAP at all levels of communication



Selecting a web services architecture

REST	SOAP
Defined as a solution to a problem - it works, it's elegant	Standards-based (W3C) technology - it's elegant in a "big framework" way
Uses HTTP verbs	Uses HTTP, FTP, SMTP, etc.
Philosophy: everything is a URI	Philosophy: formal definition of the envelope, message, protocols
Tools: simple! Can be debugged with a web browser	Tools: vendors and consultants love it. Nice and complicated, dedicated tools
Security through HTTP authentication or HTTPS	Security through HTTPS and/or W3C web service additions (WS-*) and/or app server setup
Good choice if lots of platforms are involved, basic data exchange applications	Good choice if the systems involved must use their advanced capabilities between homogeneous systems
Programmers work with XML directly	Programmers work with native objects that are mapped by the system to XML for data exchange (may require tools)



Heterogeneous systems services

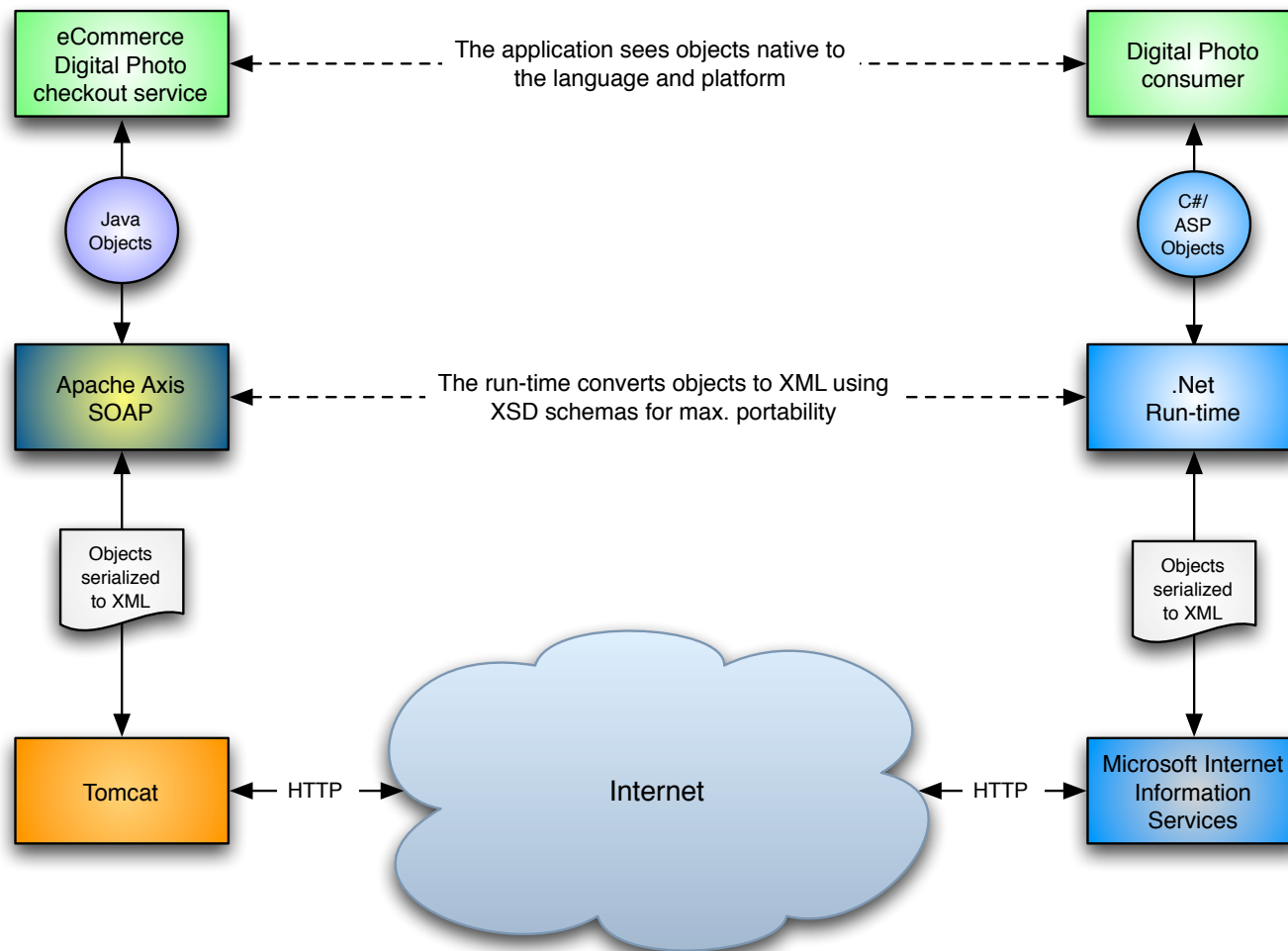
- **Best use of web services? Stateless data exchanges between heterogeneous systems**
 - Remote procedure calls (stateful)? CORBA or RMI are better solutions
 - Systems at both ends have the same architecture? Use “native” serialization
- **Examples:**
 - Java servers talking to Perl clients over REST
 - Java clients talking to .Net servers using SOAP
 - Java clients, Java servers - use RMI or native serialization instead
- **Don't make things more complicated than they need to be!**



SOAP with Axis

- **Axis is the Apache solution for SOAP web services**
- **Its biggest advantage is that the developers don't need to bother with SOAP or XML -- Axis handles that transparently**
 - Java
 - C++
- **Use the Apache version or vendor-supported**
 - IBM
 - BEA
 - others

SOAP with Axis



Deploying Apache Axis in Mission-Critical Environments



Axis pre-requisites

- **You must know all of the following:**

- **Java: servlets, applications, classes, .jars, and where they are deployed on an application server**
- **How to start and deploy applications on your application server**
- **Core HTTP and error codes, including basic authentication**
- **Basic XML**

- **Things not to do:**

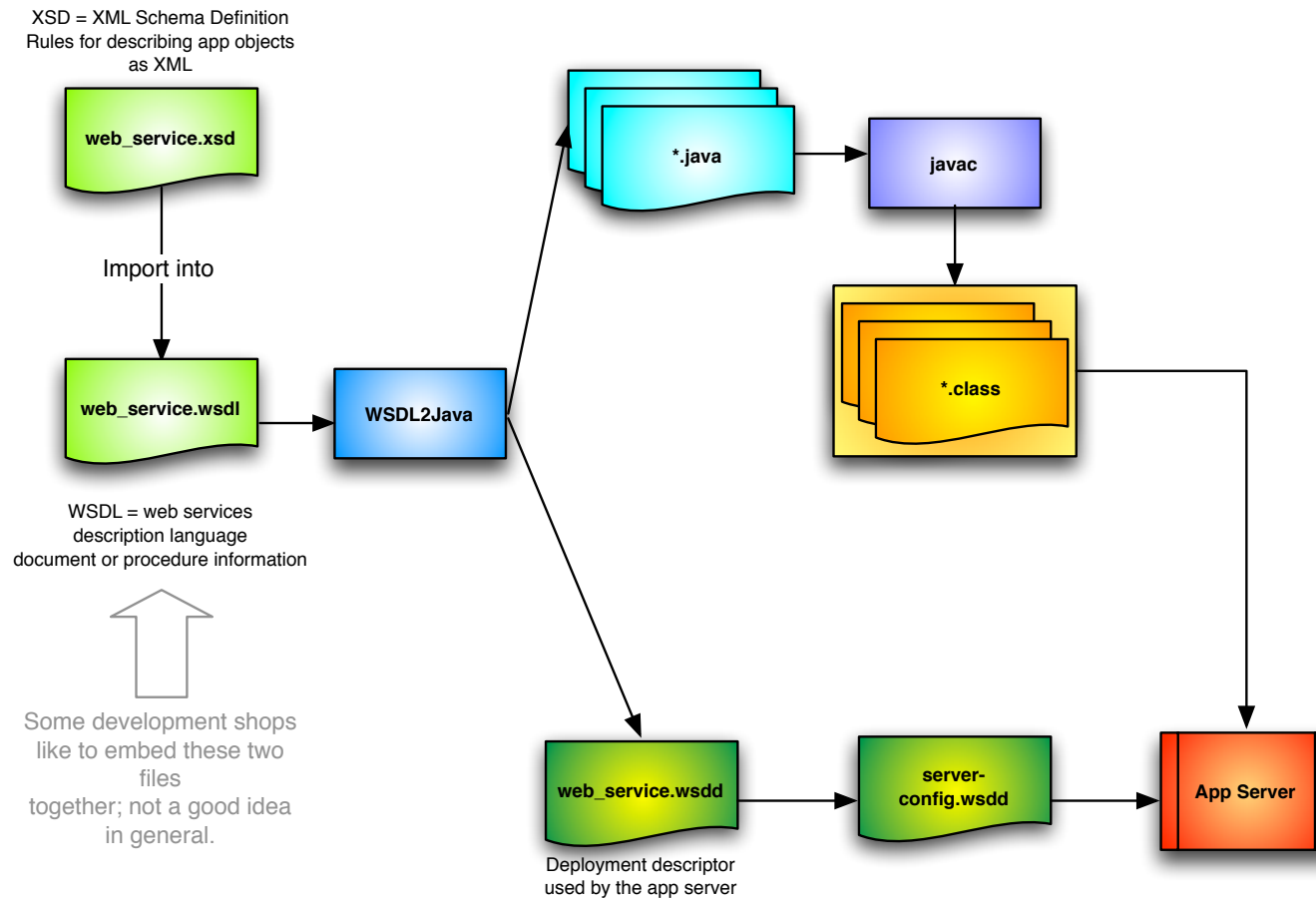
- **Don't try to learn all these concepts through Axis**
- **Don't unleash programmers to do Axis/web services if they're new to Java, regardless of how experienced they might be otherwise**



Axis configuration and deployment

- **App servers are different; check your vendor's documentation**
- **Set up:**
 - Libraries (including XML parsers)
 - Start the app server
 - Validate that Axis is running
 - Test a SOAP end point
 - Install new web services
 - Use web service deployment descriptors
 - Full instructions: <http://ws.apache.org/axis>
- **The rest of this presentation assumes that Axis is properly installed and configured**

Axis web services overview

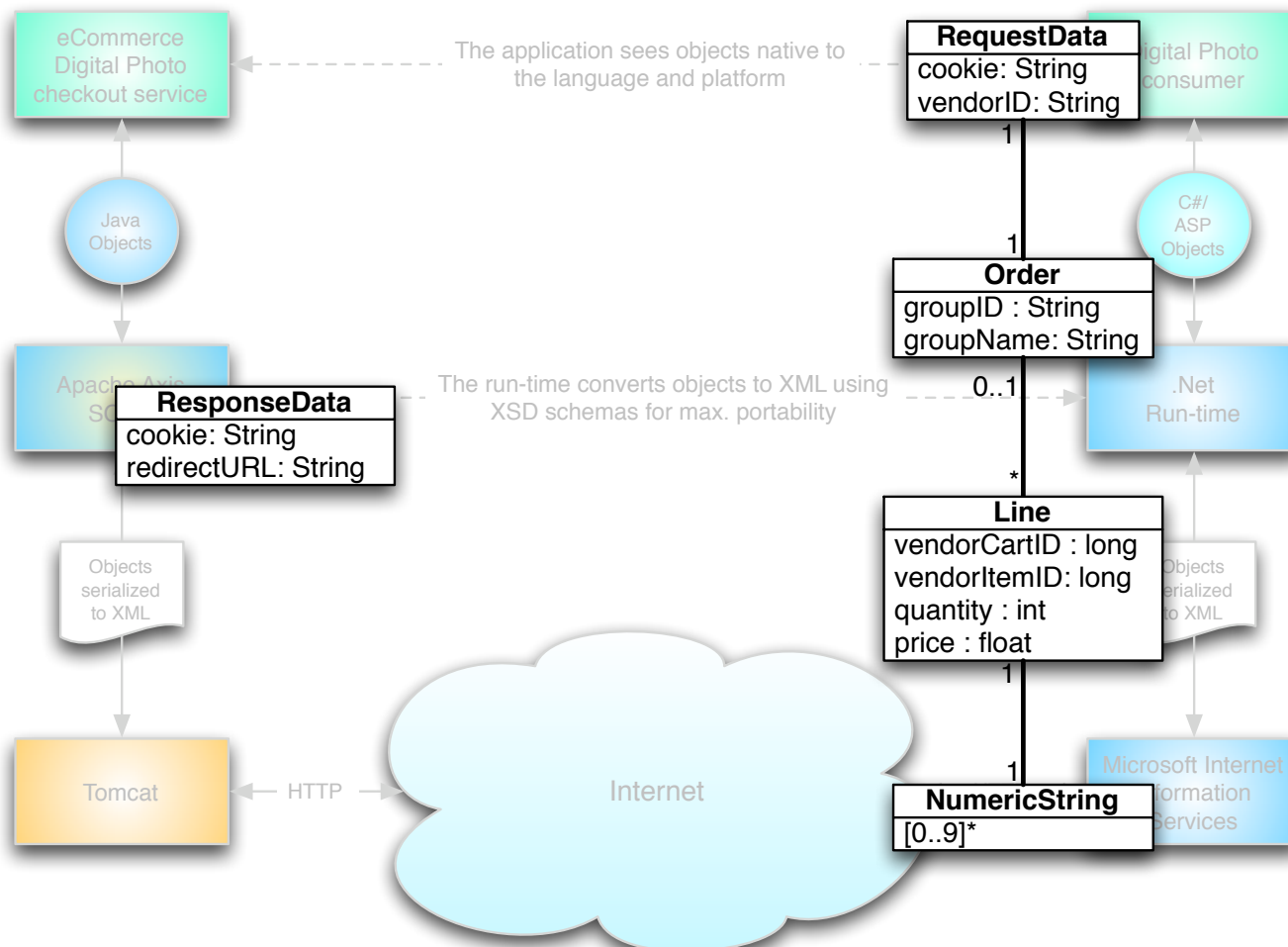




Contents of an XSD file

- **Some tools and some programmers include the schema information in the WSDL**
 - That's usually not a good idea because it couples the web services interface with the documents being exchanged
- **Some programmers like to start with an object or data model, then map that to XSD/XML using an automated tool**
 - This is also not a good idea in general because different languages may have slight data representation incompatibilities when mapped to XSD
- **Start with the XSD file, then generate your business objects from it**
 - A common schema is a better foundation than native business objects (Java, C#, Objective-C, whatever) <-- collections example
 - It's easier to generate Java/C#/C++/etc. from XSD than viceversa

Contents of an XSD file



Contents of an XSD file

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="http://www.mycompany.com/services/createcart"
  xmlns="http://www.mycompany.com/services/createcart">

  <xs:element name="RequestData" type="RequestData"/>

  <xs:complexType name="RequestData">
    <xs:all>
      <xs:element name="CustomerCookieValue" type="xs:string"
        minOccurs="1" maxOccurs="1"/>
      <xs:element name="VendorId" type="xs:string"
        minOccurs="1" maxOccurs="1"/>
      <xs:element name="Order" type="Order"
        minOccurs="1" maxOccurs="1"/>
    </xs:all>
  </xs:complexType>

  <xs:complexType name="Order">
    <xs:sequence>
      <xs:element name="Group" type="Group"
        minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="Group">
    <xs:sequence>
      <xs:element name="GroupId" type="xs:long"
        minOccurs="1" maxOccurs="1"/>
      <xs:element name="GroupName" type="xs:string"
        minOccurs="1" maxOccurs="1"/>
      <xs:element name="Line" type="Line"
        minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
```

```
<xs:complexType name="Line">
  <xs:sequence>
    <xs:element name="VendorCartId" type="xs:long"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="VendorItemId" type="xs:long"
      minOccurs="1" maxOccurs="1"/>
    <xs:element name="UPC" type="NumericString" minOccurs="1"
      maxOccurs="1"/>
    <xs:element name="Quantity" type="xs:int" minOccurs="1" maxOccurs="1"/>
    <xs:element name="Price" type="xs:float" minOccurs="0" maxOccurs="1" />
  </xs:sequence>
</xs:complexType>

<xs:element name="ResponseData" type="ResponseData"/>

<xs:complexType name="ResponseData">
  <xs:all>
    <xs:element name="CustomerCookieValue" type="xs:string" minOccurs="1"
      maxOccurs="1"/>
    <xs:element name="RedirectURL" type="xs:string" minOccurs="1"
      maxOccurs="1"/>
  </xs:all>
</xs:complexType>

<xs:simpleType name="NumericString">
  <xs:restriction base="xs:string">
    <xs:pattern value="([0-9])+"/>
  </xs:restriction>
</xs:simpleType>

</xs:schema>
```



Contents of a WSDL file

- **Specification by IBM and Microsoft for describing web services in a structured way**
- **It tells:**
 - The interface to the service
 - Where the service is located
 - The data types used by the service
- **Axis supports WSDL...**
 - by providing a browser-ready mechanism to view service descriptions
 - Through the Java2WSDL tool to generate descriptions from Java classes
 - Through the WSDL2Java tools to generate Java classes, proxies, and skeletons for the services in the WSDL



Contents of a WSDL file

```
<definitions targetNamespace="http://www.mycompany.com/services/createcart"
```

```
  xmlns="http://schemas.xmlsoap.org/wsdl/"
```

```
  xmlns:tns="http://www.mycompany.com/services/createcart"
```

```
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
```

```
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
```

```
  xmlns:types="http://www.mycompany.com/services/createcart">
```

```
  <import namespace="http://www.mycompany.com/services/createcart"
```

```
    location="CreateCartService.xsd"/>
```

```
  <message name="createCartRequest">
```

```
    <part name="parameters" element="types:RequestData" />
```

```
  </message>
```

```
  <message name="createCartResponse">
```

```
    <part name="parameters" element="types:ResponseData" />
```

```
  </message>
```

```
<portType name="CreateCartPortType">
```

```
  <operation name="createCart">
```

```
    <input message="tns:createCartRequest" />
```

```
    <output message="tns:createCartResponse" />
```

```
  </operation>
```

```
</portType>
```

```
<binding name="CreateCartSOAPBinding" type="tns:CreateCartPortType">
```

```
  <soap:binding style="document"
```

```
    transport="http://schemas.xmlsoap.org/soap/http" />
```

```
  <operation name="createCart">
```

```
    <soap:operation style="document"
```

```
      soapAction="createCart" />
```

```
  <input>
```

```
    <soap:body use="literal" />
```

```
  </input>
```

```
  <output>
```

```
    <soap:body use="literal" />
```

```
  </output>
```

```
</operation>
```

```
</binding>
```

Deploying Apache Axis in Mission-Critical Environments



Contents of a WSDL file

```
<service name="CreateCartService">  
  <port name="CreateCartSOAPPort"  
    binding="tns:CreateCartSOAPBinding">  
    <soap:address location=  
      "http://www.mycompany.com/services/CreateCartSOAPPort" />  
    </port>  
  </service>  
</definitions>
```



Putting it all together

- **WSDL2Java - stubs, skeletons and data types from WSDL**

WSDL clause	Generates...
Type section	Java class A holder if the type is used as an input/output parameter
Port type	Java interface
Binding	Stub class
Service	Service interface Service locator implementation

Putting it all together

XSD = XML Schema Definition
Rules for describing app objects
as XML



Import into



WSDL2Java

WSDL = web services
description language
document or procedure information

```

deploy.wsdd
Group.java
Line.java
makefile
NumericString.java
Order.java
RequestData.java
ResponseData.java
undeploy.wsdd
CreateCartPortType.java
CreateCartService.java
CreateCartServiceLocator.java
CreateCartServiceTestCase.java
CreateCartSOAPBindingImpl.java
CreateCartSOAPBindingStub.java
  
```

```
[pr3d4t0r@mac createcart]$ make
```

ResponseData
cookie: String
redirectURL: String

RequestData
cookie: String
vendorID: String

Order
groupID : String
groupName: String

Line
vendorCartID : long
vendorItemID: long
quantity : int
price : float

NumericString
[0..9]*



TechTarget
The Most Targeted
IT Media



Putting it all together

Example: Order.java

```

/**
 * Order.java
 * This file was auto-generated from WSDL
 * by the Apache Axis WSDL2Java emitter.
 */

package com.mycompany.www.services.createcart;

public class Order implements java.io.Serializable {
    private com.mycompany.services.createcart.Group[] group;

    public Order() {}

    public com.mycompany.services.createcart.Group[] getGroup() {
        return group;
    }

    public void setGroup(com.mycompany.services.createcart.Group[] group) {
        this.group = group;
    }

    // Type metadata
    private static org.apache.axis.description.TypeDesc typeDesc =
        new org.apache.axis.description.TypeDesc(Order.class);

    static {
        String namespaceURL = "http://www.mycompany.com/services/
createcart";

        typeDesc.setXmlType(new javax.xml.namespace.QName(
            namespaceURL, "Order"));
        org.apache.axis.description.ElementDesc elemField =
            new org.apache.axis.description.ElementDesc();
        elemField.setFieldName("group");
        elemField.setXmlName(new javax.xml.namespace.QName("",
            "Group"));
        elemField.setXmlType(new javax.xml.namespace.QName(
            namespaceURL, "Group"));
        typeDesc.addFieldDesc(elemField);
    } // class initializer

    /**
     * Return type metadata object
     */
    public static org.apache.axis.description.TypeDesc getTypeDesc() {
        return typeDesc;
    }

    /**
     * Get Custom Serializer
     */
    public static org.apache.axis.encoding.Serializer getSerializer(
        java.lang.String mechType,
        java.lang.Class _javaType,
        javax.xml.namespace.QName _xmlType) {
        return
            new org.apache.axis.encoding.ser.BeanSerializer(
                _javaType, _xmlType, typeDesc);
    }
} // end of Order class

```

Putting it all together

- **The stubs and skeletons will not be overwritten if they are not directly dependent on the WSDL or if they are intended for end-user updates**

```
java org.apache.axis.wsdl.WSDL2Java -s -v -a -o ./src -p \
com.mycompany.services.createcart ./htdocs/services/createcart/CreateCartService.wsdl
```

```
Parsing XML file: ./htdocs/photo/services/createcart/CreateCartService.wsdl
```

```
Generating ./src/com/mycompany/services/createcart/ResponseData.java
```

```
Generating ./src/com/mycompany/services/createcart/RequestData.java
```

```
Generating ./src/com/mycompany/services/createcart/Group.java
```

```
Generating ./src/com/mycompany/services/createcart/Order.java
```

```
Generating ./src/com/mycompany/services/createcart/Line.java
```

```
Generating ./src/com/mycompany/services/createcart/CreateCartServiceLocator.java
```

```
Generating ./src/com/mycompany/services/createcart/CreateCartServiceLocator.java  
CreateCartSOAPBindingImpl.java already exists, WSDL2Java will not overwrite it.
```

```
Generating ./src/com/mycompany/services/createcart/CreateCartPortType.java
```

```
Generating ./src/com/mycompany/services/createcart/CreateCartSOAPBindingStub.java
```

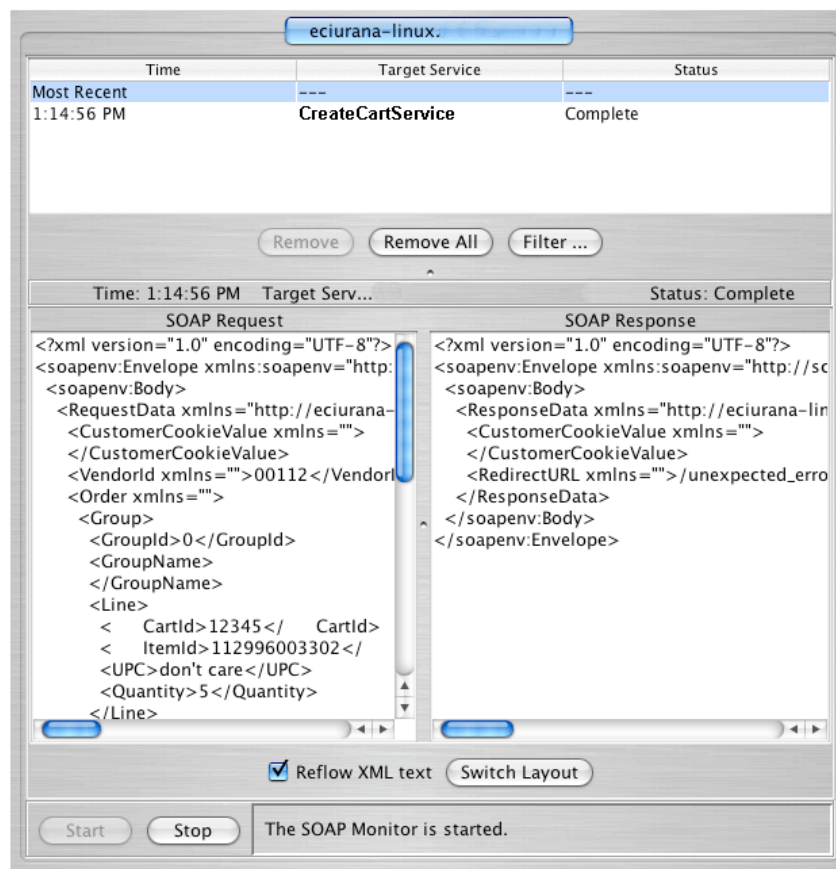
```
Generating ./src/com/mycompany/services/createcart/deploy.wsdd
```

```
Generating ./src/com/mycompany/services/createcart/undeploy.wsdd
```

Business logic is implemented in this file; the stub is only generated the very first time that WSDL2Java runs.

Putting it all together

- **So it's set up... how do I test if it's working?**

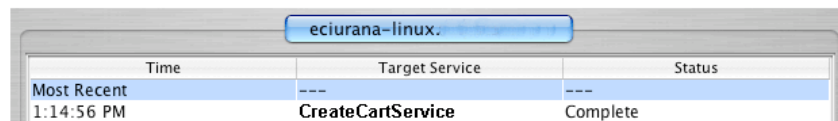


- * SOAPMonitorApplet - Axis
- * TCPMonitor - Axis
- * Web browser
- * Ethereal, Sniffer, etc.

You may need participation from the consumers to monitor the traffic at their end.

Putting it all together

- **So it's set up... how do I test if it's working?**



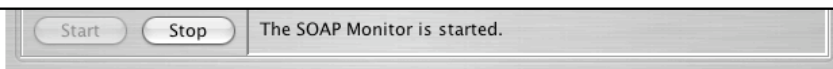
Time	Target Service	Status
Most Recent 1:14:56 PM	CreateCartService	Complete

* SOAPMonitorApplet - Axis

Logs... like /var/log/httpd/tomcat.log

```
1537 AxisFault
1538 faultCode: {http://xml.apache.org/axis/}Client
1539 faultSubcode:
1540 faultString: No such operation 'RequestData'
1541 faultActor:
1542 faultNode:
1543 faultDetail:
1544     {http://xml.apache.org/axis/}hostname:eciurana-linux.mycompany.com
1545
1546 No such operation 'RequestData'
1547     at org.apache.axis.message.SOAPFaultBuilder.createFault(SOAPFaultBuilder.java:221)
1548     at org.apache.axis.message.SOAPFaultBuilder.endElement(SOAPFaultBuilder.java:128)
1549     at org.apache.axis.encoding.DeserializationContext.endElement(DeserializationContext.java:1087)
```

n
nitor



Deploying Apache Axis in Mission-Critical Environments



Securing Axis

- **Pundits played chicken little when it comes to security (i.e. Schneier and Siddiqui, among others)**
- **Web services are as secure or insecure as you want to make any other web-related interaction**
 - Indeed, they can be secured further
- **Secure the server based on common sense, application of Axis tools, and other resources available in your system**



Securing Axis

Apache Axis recommendations

Disguise	Remove any headers that identify Axis in the response
Cut down the build	Rebuild Axis without unnecessary bits like 'instant SOAP service'
Rename things	AxisServlet, AdminService, etc. are in well-known locations; change them
Stop AxisServlet listing services	Change the Axis configuration files
Keep stack traces out of the responses	Axis has the ability to send stack traces across the wire for troubleshooting; avoid this
Stop auto-generating WSDL files	Use configuration parameters for this
Use servlet 2.3 filters	Add arbitrary authentication parameters to prevent tampering with Axis
Leverage your logs	Logs are your friends; use them either from your code, log4j, etc.



Securing Axis

Apache Axis recommendations

Run Axis with reduced Java rights

Axis doesn't need access to anything other than `server-config.wsdd` and a temporary storage for JWS files; everything else should be locked down

Run the web server with reduced rights

Monitor load

Load monitors are excellent for detecting DoS attacks; use `AxisBaseServlet` to monitor Axis performance

IDS systems

Honeypots and intrusion detection systems should be part of a production environment

Monitor the mailing lists

Axis-Dev and others discuss potential issues in a timely manner



Securing Axis

- **As you can see, security wasn't a priority**
 - IBM and other third parties provide WS extensions for security
- **Nothing wrong with a couple of external approaches**
 - Use SSL
 - Use filters for session authentication
- **Axis over SSL involves these steps:**
 - Configure Java Secure Socket Extensions' keystores
 - Generate the appropriate certificates
 - Configure your servlet container to handle SSL communications
 - Configure Axis to handle basic authentication

Securing Axis

- **As you can see, security wasn't a priority**

- IBM

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>Protected Area</web-resource-name>
    <!-- Define the context-relative URL(s) to be protected -->
    <url-pattern>/services/createcart</url-pattern>
    <!-- If you list http methods, only those methods are protected -->
    <http-method>GET</http-method>
    <http-method>POST</http-method>
  </web-resource-collection>
  <auth-constraint>
    <!-- Anyone with one of the listed roles may access this area -->
    <role-name>MyConsumer</role-name>
  </auth-constraint>
</security-constraint>

<login-config>
  <auth-method>BASIC</auth-method>
  <realm-name>Protected Area</realm-name>
</login-config>
```

- **Nothing**
- appro**

- Use

- Use

- **Axis**

- Con

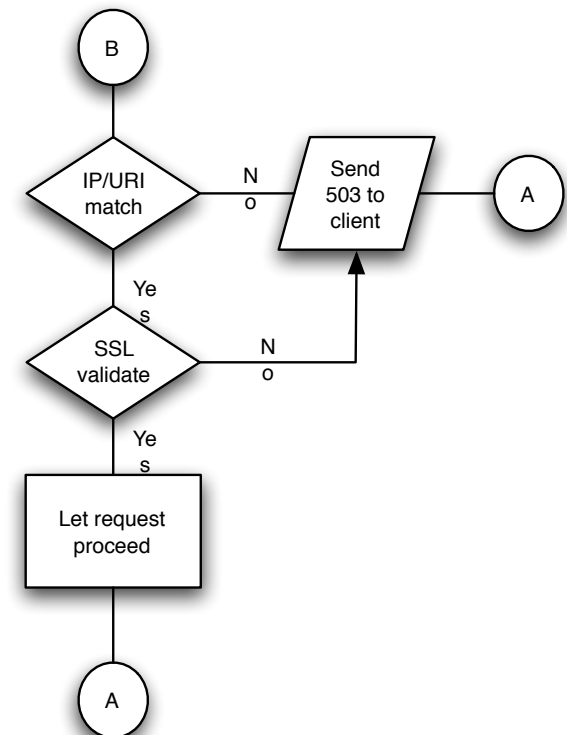
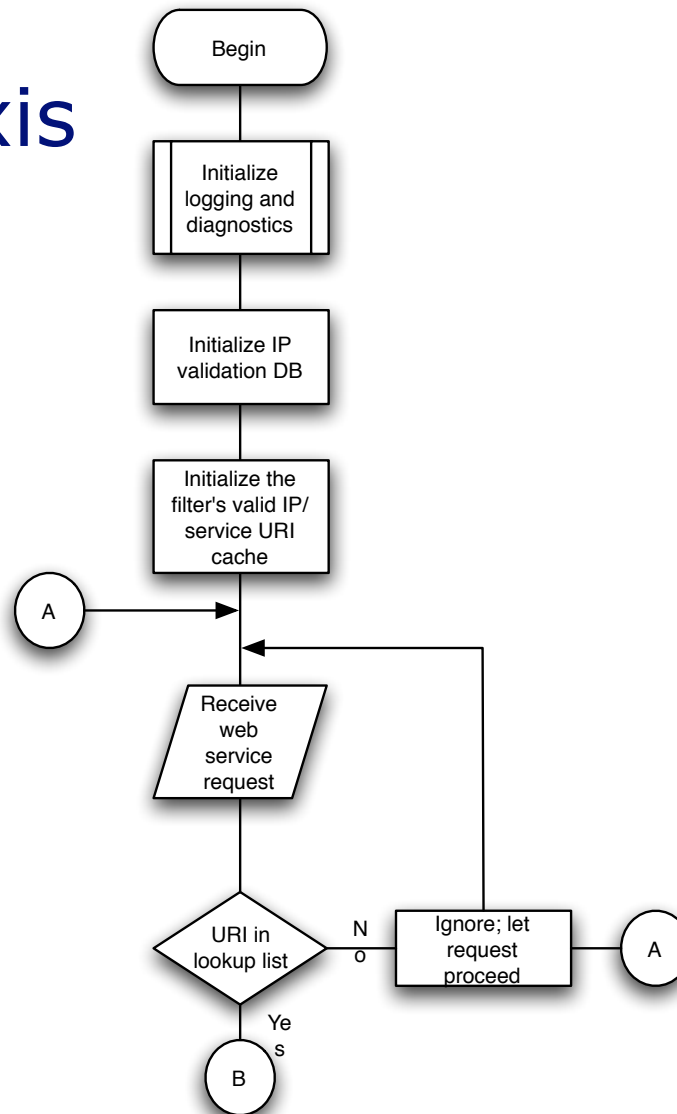
- Gen

- Con

- **Configure Axis to handle basic authentication**

Securing Axis

An additional way of securing web services could be through the use of a servlet filter. The filter can check things like valid IP addresses against a database, like the filter shown in this diagram...



Someone suggested using Apache's httpd.conf to block or allow certain IPs to access the web service... why is this not a good idea?



Deploying Axis in Mission-Critical Environments

Q&A

Eugene Ciurana

eugenex@walmart.com

You can find this presentation at:

<http://eugeneciurana.com/musings/JavalnAction.html>