Apache Axis2 in action

Deepal Jayasinghe - WSO2 Inc.
About me

• I am a senior software engineer working at WSO2
• PMC Member Apache Web services
• Working in Axis2 projects since its day one
• Contribute to a number of Apache projects
About WSO2

- Open source Web service middleware technology company.
- World-class open source leadership
- Focus on Apache Web services projects
- Creating easy to use, high performance products
- XML and Web centric services and integration ready applications
- Development in Java, C and PHP
- Emphasis on interoperability standards, integration and SOA
- Head quartered in Boston, USA
- Office in UK and Sri Lanka
Agenda

- Motivation
- Understanding and working with Axiom
- Learning Axis2 basics
- Understanding the deployment model
- Writing a service and deploying
- Writing a module and deploying
- Working with client API
- A practical scenario
- Stub and skeleton generation
- Axis2 and POJOs
Motivation for Axis2

History of ASF SOAP engines
- Apache SOAP
- Axis 1.x designed as a follow-on

Why do we need a new SOAP engine?
- Changes to the Web services landscape
  - WS-A, WS-RM
- Performance
  - Parsers, Optimizing based on use
- Ease of use
  - Deployment of new capabilities, service deployment
AXIOM
New XML Infoset Representation

Known as AXIOM (AXIS Object Model)
NOT, just another XML object model
API is more like a simplified DOM

Fundamental difference?
Objects are created “on demand” using a pull model
Allows direct access to the underlying pull stream with or without building the tree
Support for storing binary data
New XML Infoset Representation
Cont...

- API also provides a StAX parser interface at any element
- Allows the event based navigation of the OM tree.
New XML Infoset Representation
Cont...
New XML Infoset Representation

Cont...

- In built binary storage support
- Can store binary (unchanged)
- Supports XOP/MTOM
- XOP (XML Optimized Packaging)/MTOM (Message Transmission Optimization Mechanism)??
AXIOM and Axis2

AXIOM is the primary means of representing / manipulating the XML message inside Axis2
Time to Dig Into Code.. 😊

- Code samples to explain AXIOM
- Serialization
- Deserialization
- XPath navigation
Axis2 Basis
Phases, Modules, Handlers
Extensible Messaging Engine
Message Processing Stages

There are three main stages:

- **Transport Receiver**
  - Transport related processing
- **Dispatching**
  - Finding service and operation
- **Message Receiver**
  - Last handler of the chain
Dispatching

Two types of dispatching
- Finding the corresponding descriptions
- Finding the corresponding contexts

Default dispatchers
- AddressingBasedDispatcher
- RequestURIBasedDispatcher
- SOAPActionBasedDispatcher
- SOAPMessageBodyBasedDispatcher
Dispatching ctd...

POST /axis2/services/EchoXMLService/echoOMEElement HTTP/1.1
User-Agent: Axis2
Host: 127.0.0.1
Content-Type: application/soap+xml; charset=UTF-8;action="EchoOMEElement";

....................

<?xml version='1.0' encoding='UTF-8'?>
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
<soapenv:Header>
<wsa:To>http://127.0.0.1:5556/axis2/services/EchoXMLService/echoOMEElement</wsa:To>
<wsa:ReplyTo>
<wsa:Address>http://www.w3.org/2005/08/addressing/anonymous</wsa:Address>
</wsa:ReplyTo>
<wsa:MessageID>urn:uuid:AD147449058471C81E11506120248601</wsa:MessageID>
<wsa:Action>urn:EchoOMEElement</wsa:Action>
</soapenv:Header>
<soapenv:Body>
<ns1:echoOMEElement xmlns:ns1="http://org.apache.axis2/xsd">
<ns1:myValue>Isaac Asimov, The Foundation Trilogy</ns1:myValue>
</ns1:echoOMEElement>
</soapenv:Body>
</soapenv:Envelope>
Message Receiver

- The last handler of the execution chain
- MEP dependent (MEP ?? )
- Does the actual business logic invocation
- Ability to write custom Message Receivers
- Injects dependencies to services
- Some default Message Receivers
  - RawXMLINOnlyMessageReceiver
  - RawXMLINOutMessageReceiver
  - RPC*MessageReceiver
Message Exchange Patterns - MEP

Describes the exchange pattern of SOAP messages per given operation.

E.g.

- In – Out
- In Only
- In – In – Out!

WSDL 2.0 defines 8 standard MEPs.

Axis2 supports all inbound MEPs
Contexts and Descriptions Hierarchy

- Descriptors keep static information
  - Information extracted from deployment descriptors
- Contexts keep runtime information
  - This Information needs to be in various scopes
  - Good to keep them separate!
Parameters and Properties

Parameters
- Defining a parameter
- The “locked” attribute
- Setting and getting
- Parameter can be any object
- Getting the original OMEElement from the parameter
- Retrieving parameters (getParameter)

Properties
- Difference between property and parameter
- Retrieving properties (getProperty)
Deployment Model
What's the Fuss with Deployment?

Axis 1.x deployment requires you to:

- Either modify the XML files
- or
- Call the admin client
- Add to the classpath
- Restart the server

For a beginner, a bit of headache 😞
New Deployment Model

- Archive based deployment
  - Bundle all together and drop in
- Directory based deployment (similar structure as archive)
- Hot Deployment 😊😊
- Archive file can contain:
  - Class files
  - Third party libraries
  - Any other resources required by the service
Axis2 Service
Axis2 Service

- Can be deployed as an archive (.aar) file or as a directory with all necessary resources
- Isolated – separate Class loader
Service Descriptor

- Service configurations are given by the `services.xml`
  - No need to have a WSDL around to be a valid service !!!
- Contains
  - ServiceClass parameter
  - Name spaces
  - Expose transports
  - Service scope
- Operation
  - actionMapping
  - MessageReceiver
- Modules to be engaged
Service vs. Service Group

- Deploying multiple services together
- Share data across services in a group
- Maintain sessions across a service group using contexts

Example use case of a Service Group
- Login
- Do something
- Log out
Service Scope

- Request scope
- SOAP session scope
  - Service group ID
- Transport session scope
  - Cookies
- Application scope
Back to Coding

- Writing services.xml
  - With single service
  - For a service group

- Writing service class
  - Explain dependency injection
  - Methods for life cycle management

- Co-relating WSDL file to a service
Axis2 Module
What is a Module?

- Modules define the way of extending Axis2
- Encapsulates a specific functionality (mostly a WS-* function)
  - e.g. Addressing module adds WS-Addressing support
- Usually consists of a set of handlers
- Modules are not hot deployable
  - Because they change the overall behaviour of the system
Inside an Axis2 Module

What does it contain?
- Module descriptor: module.xml
  (more in the next slide)
- Module implementation class
- Handlers
- Third party libraries

Can be deployed as an archive file 😊
  Bundle all together and deploy

Can be deployed as a directory as well

Isolated – separate class loader
<module name="addressing">
  <Description/>
  <InFlow>
    <handler name="AddressingFinalInHandler"
      class="org.apache.axis2.handlers.addressing.AddressingFinalInHandler">
      <order phase="PreDispatch"/>
    </handler>
    ............
  </InFlow>
  <OutFlow>
    <handler name="AddressingOutHandle"
      class="org.apache.axis2.handlers.addressing.AddressingOutHandler">
      <order phase="MessageOut"/>
    </handler>
  </OutFlow>
  <OutFaultFlow>
    ............
  </OutFaultFlow>
  <InFaultFlow>
    ............
  </InFaultFlow>
</module>
Availability and Engaging of Modules

- **Concept of Availability**
  - Presence of the module in the system

- **Concept of Engaging**
  - Activating the module
  - Can be done
    - Per System
    - Per Service group
    - Per Service
    - Per Operation
Sample module with two handlers
Sample module with a module implementation class
Explains
  • Init
  • engageNotify
  • shutdown etc.
Client API
ServiceClient

- Supports both blocking and non-blocking invocations models
- Concept of callbacks for the client for non-blocking case
- Can handle both transport dependent and transport independent asynchrony.
Invocation Patterns

- sendRobust
- fireAndForget
- sendReceive
- sendReceiveNonBlocking
Operation Client

Why do we need Operation client?

Service Client has a set of operation clients

- If you need more control use OperationClient
What are Options?

- Why do we need options for the client?
- What is included in options?
  - Addressing information
  - SOAP action (wsa:action)
  - Transport data
  - Properties
There's Nothing Like Code to Explain it!

- Simple Client written from scratch
- Invoke using all the available patterns
- Working with operation client
- An example dynamic client
- How to use RPCServiceClient
Practical use case

- Creating a session aware service
- Implementing a client to invoke the service
- Deploy the service in different scopes and see the behaviour
Code Generation
Code Generation

java org.apache.axis2.wSDL.WSDL2Code

Usage WSDL2Code -uri <Location of WSDL> : WSDL file location
-o <output Location> : output file location
-a : Generate async style code only. Default if off
-s : Generate sync style code only. Default if off. Takes precedence over -a
-p <package name> : set custom package name
-l <language> : valid languages are java and csharp. Default is java
-t : Generate Test Case to test the generated code
-ss : Generate server side code (i.e. skeletons). Default is off
-sd : Generate service descriptor (i.e. axis2.xml). Default is off. Valid with -ss
-u : unpack classes
-f : flatten output
-ns2p : namespace to package mapping
-d: choose databinding model – adb, xmlbeans, jibx none
Generated Code: Client

Structure

- Stub
- Empty Callback Handler
- Databinding classes - Depends on the selected databinding framework
- Ant build file
Generated Code: Service

Structure

- Skeleton
- Custom Message Receiver
- services.xml
- WSDL file
Code Again...

- Codegen demonstration with the command line tool
- Generate skeleton, fill that and deploy
- Generate stub and invoke a service
Resources and Contribution

For more information and latest news:
- http://ws.apache.org/axis2
- www.wso2.net

All the samples and presentation slides are available to download:
- http://people.apache.org/~deepal/xml06/
Questions?

axis-user@ws.apache.org

Don’t forget to use [Axis2] prefix in the subject

IRC channel

#apache-axis
Thank you!!!!!!