

Using Web Services to Access Legacy C/C++ Code from Java

Paul Fremantle

Senior Technical Staff Member

IBM Hursley Park





Who Am I?

- pzf@uk.ibm.com
- Lead development architect for
 - IBM's Web Services Gateway
 - C/C++ Web Services Toolkit
- Co-author of *Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI (2nd Edition)* – an excellent book about WSinJ
- This session will cover the use of Apache Axis in Java and C/C++ to enable legacy integration





Contents

- What is the problem?
- Real scenarios
- Existing / other solutions
- Axis Java
- Axis C/C++
- Example – C++ calling Java services
- Example – Java calling C++ Services
- Performance
- Still to do
- Futures



Problems

- There are a large number of existing C/C++ applications
 - Typically resource pressure means rewriting these is not an option
 - Also some applications would not benefit from rewriting in Java – especially computationally intensive processing apps:
 - Market analysis
 - eScience applications
 - Restricted environments





Service Oriented Architecture

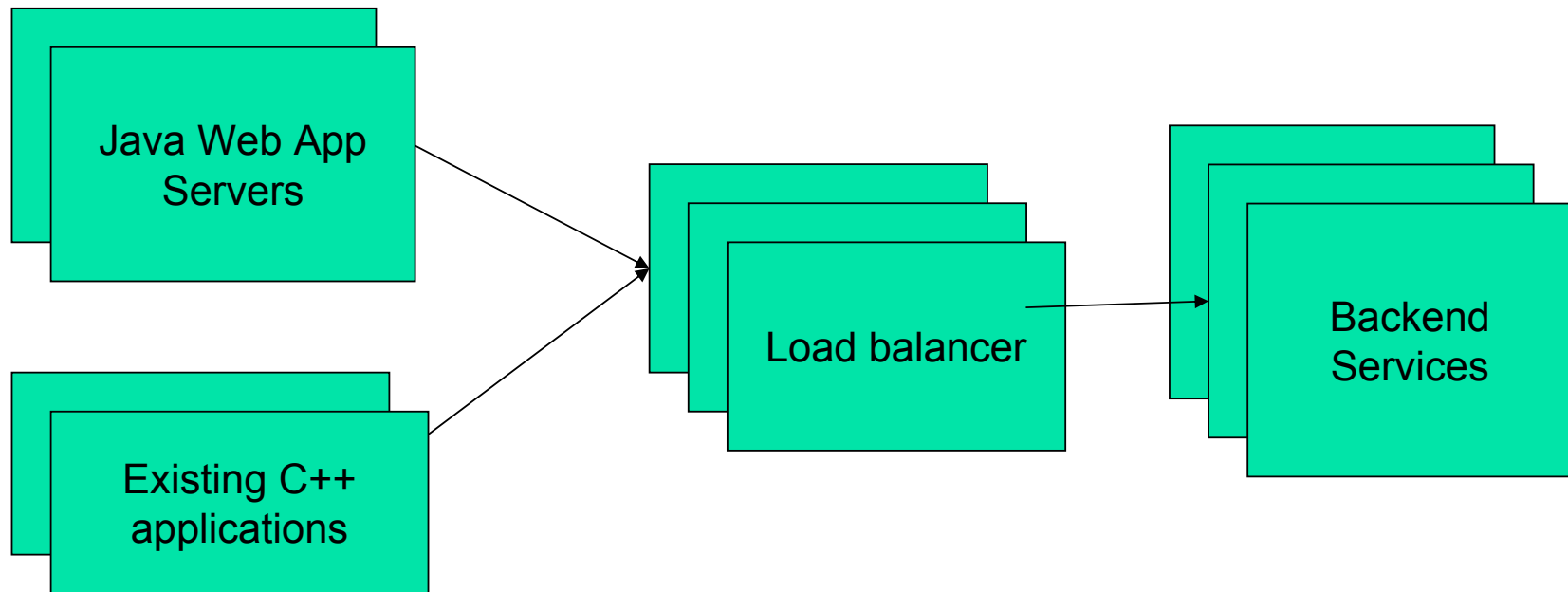
- The fundamental premise:
 - A service consumer doesn't care *how* the service is implemented – only *what* it offers

- Common interoperable interface defined in XML and Schema
 - SOAP provides the communication protocol
 - WSDL provides the information on the service interface and location
 - UDDI provides a global or local directory of services and additional metadata



Scenarios #1

Online services

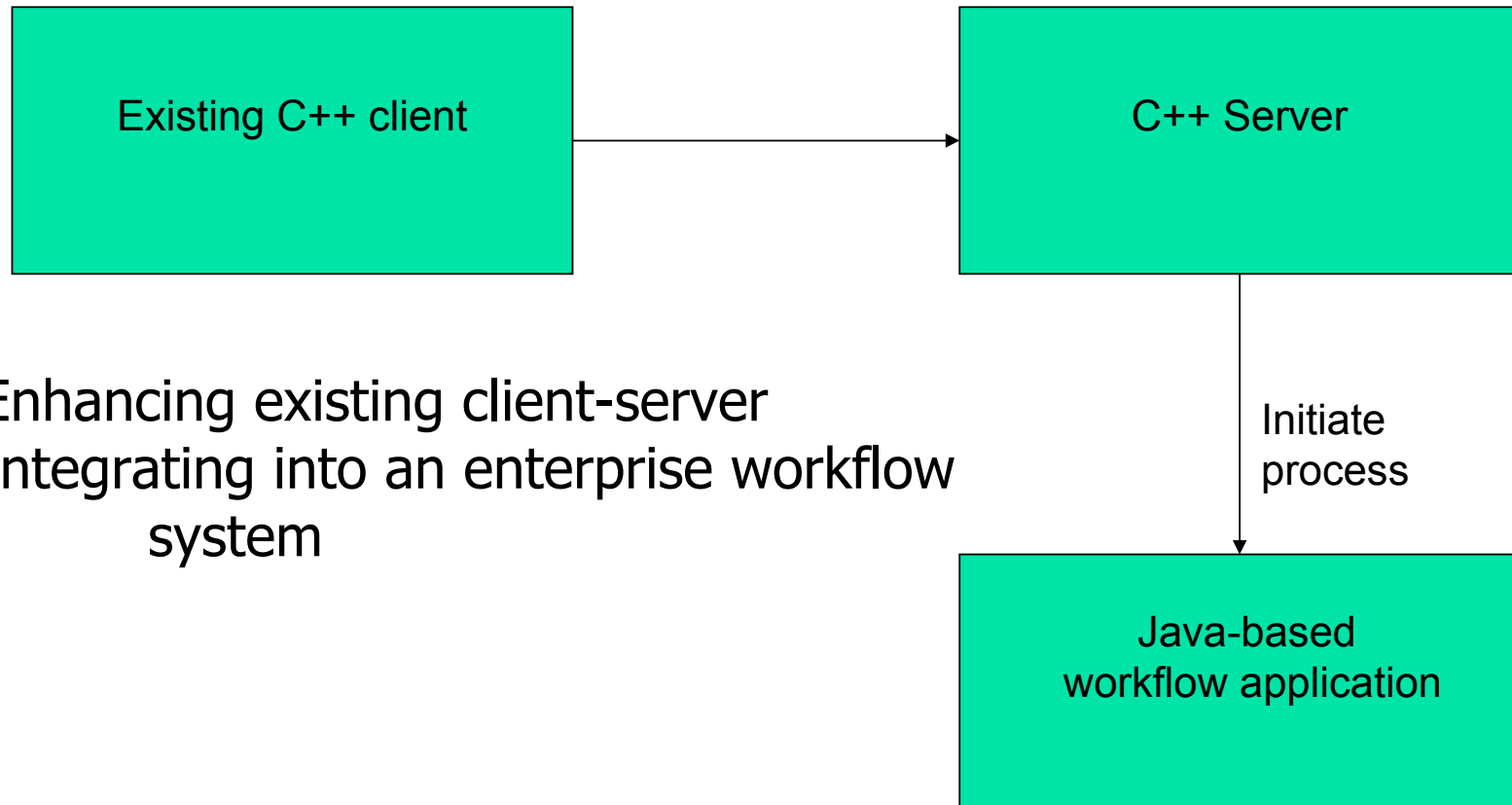


Re-architecting multi-tier computing based on
open standards
High performance, high scalability



Scenarios #2

Client Server enhancement

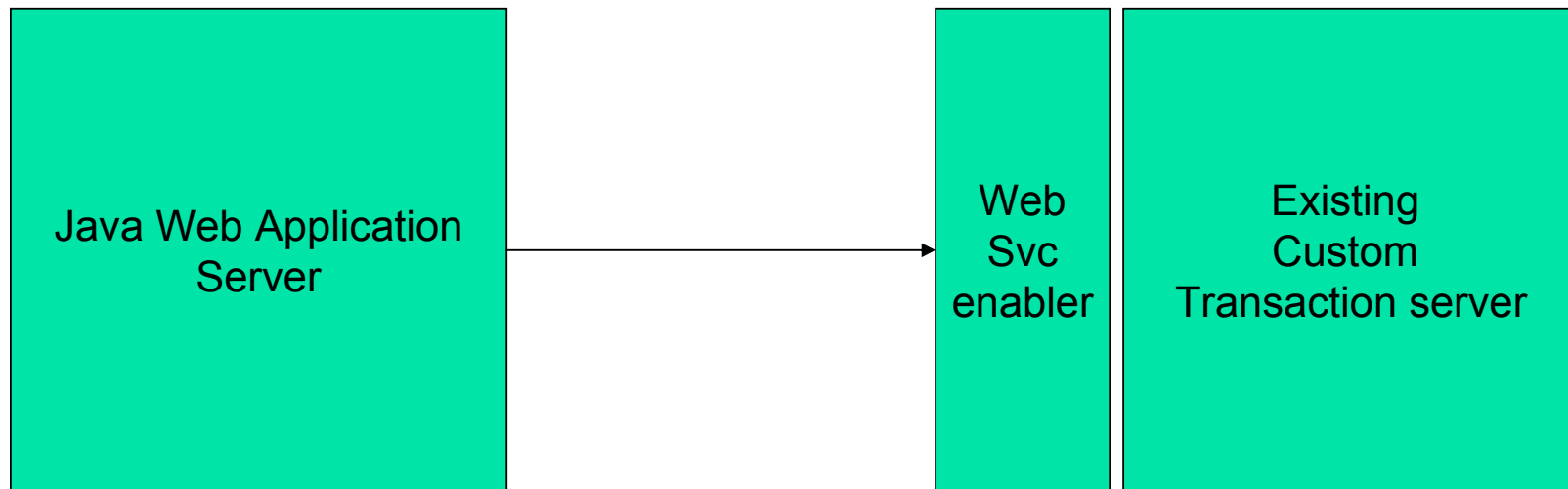


Enhancing existing client-server
Integrating into an enterprise workflow
system





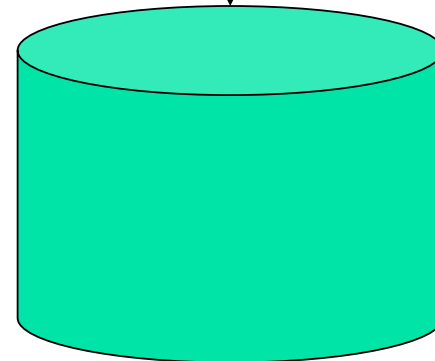
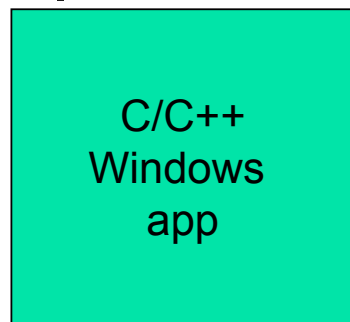
Scenario #3



Providing simple access to legacy transaction engines
and custom servers
Web refacing of client server systems



Scenario #4



Accessing Java server business logic from a C/C++ windows application





Other Approaches

- JNI – Java Native Interface
 - Good for a single machine, single process model
 - Difficult to scale or distribute
 - Unpleasant development model
- CORBA
 - JPH = Just Plain Hard!
- JCA connectors
 - Provide legacy system connectors for systems such as CICS, IMS, SAP, *etc.*
 - Good when they exist but hard to create
 - Latest specification supports both
 - Outbound (Java → legacy) and
 - Inbound (legacy → Java)





What Is Axis?

- An open source project in the Apache Software Foundation to provide high-quality SOAP engines in Java and C/C++
 - Operated on a “meritocracy”
 - The more you do, the more responsibility you obtain
 - Committers are active members on the project who have access to commit code to the repository
 - Java project made its first alpha available Aug 2001
 - Came out of Apache SOAP project initially donated by IBM**
 - C++ project made its first alpha available Dec 2003
 - Came out of an initiative from a team in Sri Lanka – Lanka Software Foundation
 - <http://www.opensource.lk/>

** see <http://www.manageability.org/blog/stuff/most-valuable-personalities-in-java>

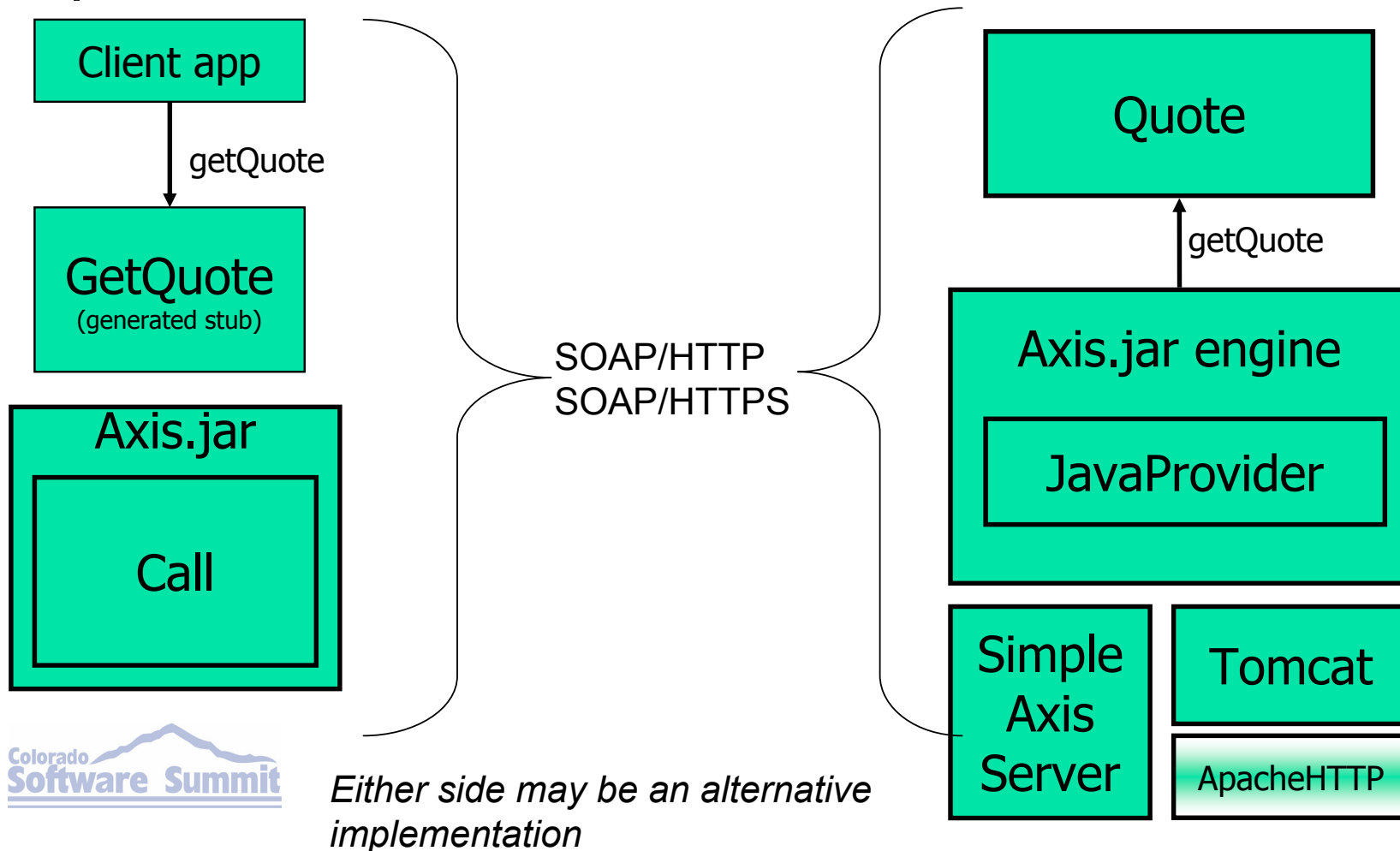




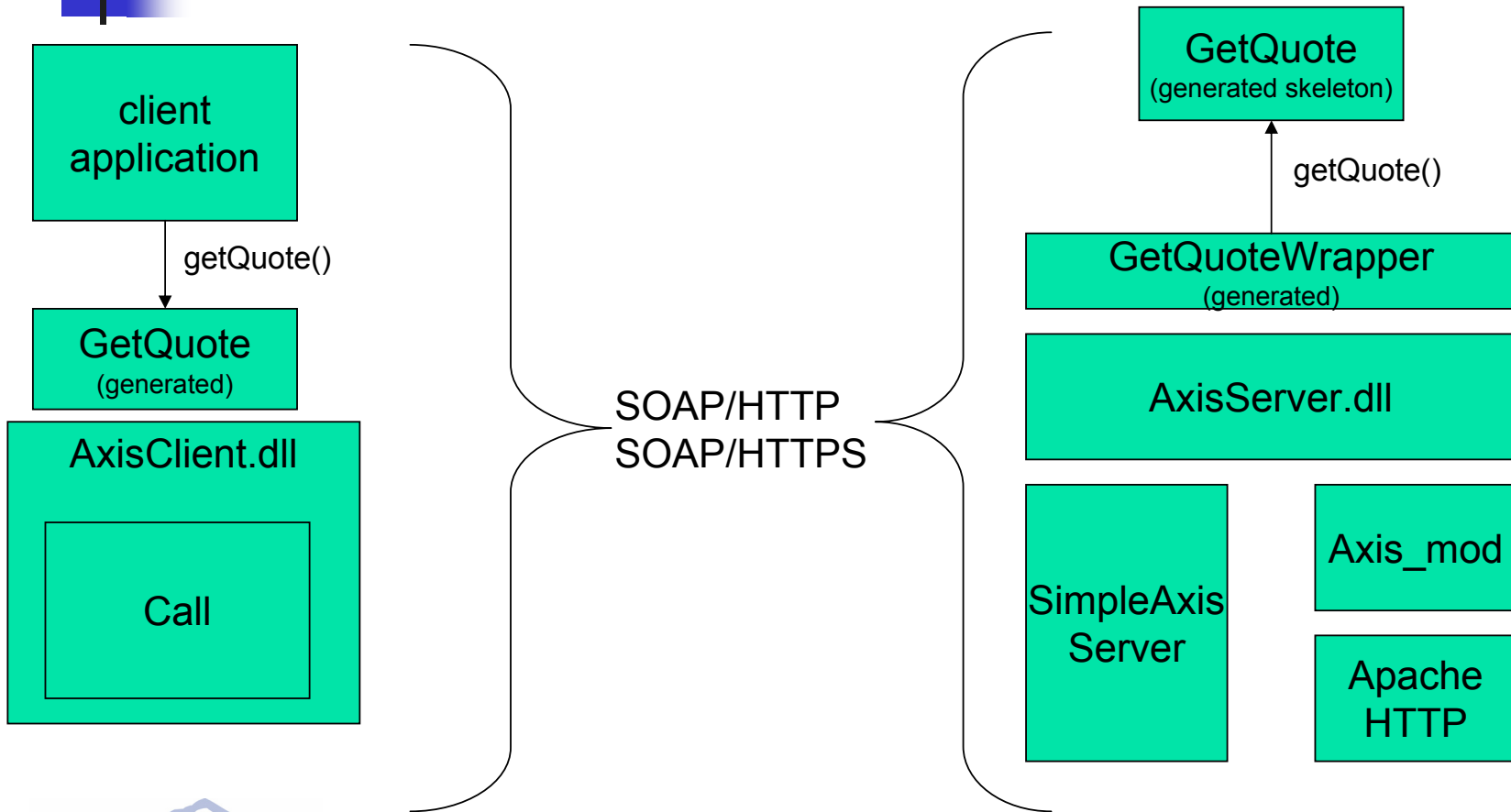
What Is a SOAP Engine?

- Main functions are:
 - Requester-side
 - Provide stubs or a dynamic invocation interface (DII)
 - Map local language objects/parameters to SOAP XML messages, *vice-versa*
 - Send those XML messages over a transport (HTTP)
 - Provider-side
 - Take SOAP messages and invoke local language objects or methods
 - Listen on a transport (HTTP), and plug-in to existing HTTP servers (Apache)
 - Tooling
 - Take WSDL and generate stubs and skeletons
 - (take existing language artefacts and generate WSDL)

Overview of Axis Java



Overview of Axis C/C++



Either side may be an alternative implementation



What Is Axis C/C++

- The main components are:
 - AxisClient.dll
 - The client code library
 - AxisServer.dll
 - The server code library
 - AxisTransport.dll
 - The HTTP transport library
 - WSDL2WS
 - A tool for generating stubs and skeletons from WSDL
 - Note.... This is written in Java so requires a JRE to run
 - mod_axis.dll, mod_axis2.dll
 - Apache HTTP 1.x, 2.x modules that link to AxisServer.dll





A Simple Example

- The Quote service is sort of the Hello World of Web Services ☺

- GetQuote.wsdl

```
<message name="testRequest"/>
<message name="testResponse">
  <part name="testResult" type="xsd:string"/>
</message>
```

This defines one interface with two methods

```
<message name="GetQuoteRequest">
  <part name="symbol" type="xsd:string"/>
</message>
```

“public String test()”

```
<message name="GetQuoteResponse">
  <part name="result" type="xsd:float"/>
</message>
```

“public float getQuote(String)”

```
<!-- Interface -->
<portType name="GetQuote">
  <operation name="getQuote" >
    <input message="tns:GetQuoteRequest"/>
    <output message="tns:GetQuoteResponse"/>
  </operation>
  <operation name="test" >
    <input message="tns:testRequest"/>
    <output message="tns:testResponse"/>
  </operation>
</portType>
```

Colorado
Software Summit

Client Side – C++

➤ **CLASSPATH=**
[path]\WSDL2Ws.jar;[path]\axis.jar;[path]\commons-discovery.jar;
[path]\commons-logging.jar;[path]\jaxrpc.jar;
[path]\log4j-1.2.8.jar;[path]\saaj.jar;[path]\wsdl4j.jar

➤ **java org.apache.axis.wsdl.wsdl2ws.WSDL2Ws**
GetQuote.wsdl –sclient –lc++
 getQuoteoperation name&&&&&&&&&&&&&&&&&&
 testoperation name&&&&&&&&&&&&&&&&&&
 floatLanguageName
 Output Parameter type Name :float
 xsd__stringLanguageName
 Output Parameter type Name :xsd__string
 C:\cppwin\.\GetQuote.cpp created.....
 C:\cppwin\.\GetQuote.h created.....
 faultInfoName is:Client
 C:\cppwin\.\AxisClientException.h created.....
 C:\cppwin\.\AxisClientException.cpp created.....





Client GetQuote.h

```
#include <axis/client/Stub.h>
#include "AxisClientException.h"
#include <axis/ISoapFault.h>

class GetQuote :public Stub
{
public:
    GetQuote(const char* pchEndpointUri, AXIS_PROTOCOL_TYPE eProtocol=APTHTTP);
    GetQuote();
public:
    virtual ~GetQuote();
public:
    float getQuote(xsd__string Value0);
    xsd__string test();
    int getFaultDetail(char** ppcDetail);
};
```






Example Client Code

```
int main(int argc, char* argv[])
{
    char endpoint[256];
    const char* server="localhost";
    const char* port="80";
    char* pcDetail;

    xsd__string symbol = "IBM";
    sprintf(endpoint, "http://%s:%s/axis/Calculator", server, port);

    GetQuote quote(endpoint);
    try {
        float fResult = quote.getQuote(symbol);
        printf("Result : %f\n", fResult);
    } catch(AxisException& e)
    {
        printf("Exception : %s\n", e.what());
    }
    return 0;
}
```





Java Client Side (for comparison)

- `set classpath=`
 - `lib\;lib\axis.jar;lib\commons-discovery.jar;`
 - `lib\commons-logging.jar;lib\jaxrpc.jar;`
 - `lib\log4j-1.2.8.jar;lib\saaj.jar;lib\wsdl4j.jar`

- `java org.apache.axis.wsdl.WSDL2Java GetQuote.wsdl -v`
 - Parsing XML file: `GetQuote.wsdl`
 - Generating `xmltoday_delayed_quotes\GetQuoteService.java`
 - Generating `xmltoday_delayed_quotes\GetQuoteServiceLocator.java`
 - Generating `xmltoday_delayed_quotes\GetQuote.java`
 - Generating `xmltoday_delayed_quotes\GetQuoteBindingStub.java`





GetQuote.java

```
package xmltoday_delayed_quotes;
```

```
public interface GetQuote extends java.rmi.Remote {  
    public float getQuote(java.lang.String symbol) throws  
        java.rmi.RemoteException;  
    public java.lang.String test() throws java.rmi.RemoteException;  
}
```





GetQuote Skeleton

```
#include "GetQuote.h"

GetQuote::GetQuote() {}

GetQuote::~~GetQuote() {}

void GetQuote::onFault() {}

void GetQuote::init() {}
void GetQuote::fini() {}

float GetQuote::getQuote(xsd__string Value0) {}

xsd__string GetQuote::test() {}
```





Java Server-side

- >java org.apache.axis.wsdl.WSDL2Java GetQuote.wsdl --server-side -v
 - Parsing XML file: GetQuote.wsdl
 - Generating xmltoday_delayed_quotes\GetQuoteService.java
 - Generating xmltoday_delayed_quotes\GetQuoteServiceLocator.java
 - Generating xmltoday_delayed_quotes\GetQuote.java
 - Generating xmltoday_delayed_quotes\GetQuoteBindingStub.java
 - Generating xmltoday_delayed_quotes\GetQuoteBindingImpl.java
 - Generating xmltoday_delayed_quotes\deploy.wsdd
 - Generating xmltoday_delayed_quotes\undeploy.wsdd





GetQuoteBindingImpl

```
package xmltoday_delayed_quotes;

public class GetQuoteBindingImpl implements xmltoday_delayed_quotes.GetQuote
{
    public float getQuote(java.lang.String symbol) throws java.rmi.RemoteException
    {
        return -3;
    }

    public java.lang.String test() throws java.rmi.RemoteException {
        return null;
    }
}
```





WSDD Files

- Web Services Deployment Descriptor
 - Configuration and deployment data required
- Server-side
 - The implementation class
 - Configuration properties
 - Handlers (explained later)
- Client side (optional)
 - Handlers





Deploy.wsdd C++

```
<?xml version="1.0" encoding="UTF-8"?>
<deployment xmlns="http://xml.apache.org/axis/wsdd/"
  xmlns:CPP="http://xml.apache.org/axis/wsdd/providers/_CPP">
  <service name="GetQuoteService" provider="CPP:RPC" description="Axis C++ web
  service">
    <parameter name="className"
      value="/user/local/apache/axis/GetQuoteService.so"/>
    <parameter name="allowedMethods" value="getQuote test "/>
  </service>
</deployment>
```





Deploy.wsdd Java

```

<deployment
  xmlns="http://xml.apache.org/axis/wsdd/"
  xmlns:java="http://xml.apache.org/axis/wsdd/providers/java">

  <service name="GetQuote" provider="java:RPC" style="rpc" use="encoded">
    <parameter name="wsdlTargetNamespace" value="urn:xmltoday-delayed-quotes"/>
    <parameter name="wsdlServiceElement" value="GetQuoteService"/>
    <parameter name="wsdlServicePort" value="GetQuote"/>
    <parameter name="className" value="xmltoday_delayed_quotes.GetQuoteBindingImpl"/>
    <parameter name="wsdlPortType" value="GetQuote"/>
    <operation name="getQuote" qname="operNS:getQuote" xmlns:operNS="urn:xmltoday-delayed-quotes"
      returnQName="result" returnType="rtns:float"
      xmlns:rtns="http://www.w3.org/2001/XMLSchema" >
      <parameter name="symbol" type="tns:string" xmlns:tns="http://www.w3.org/2001/XMLSchema"/>
    </operation>
    <operation name="test" qname="operNS:test" xmlns:operNS="urn:xmltoday-delayed-quotes"
      returnQName="testResult" returnType="rtns:string"
      xmlns:rtns="http://www.w3.org/2001/XMLSchema" >
    </operation>
    <parameter name="allowedMethods" value="getQuote test"/>

  </service>
</deployment>

```





Contract First?

- So far the work has all been WSDL → Java / C++
- Axis Java also supports Java2WSDL
 - Creates schema based on Java Types
- Axis C/C++ does not yet have this
 - Issues around too many (void *)!
- So for exposing C++ objects server side you must create a WSDL first
 - This is called “Contract-First”
 - A lot of good people believe in this:

- <http://pluralsight.com/blogs/aaron/archive/2004/08/27/2092.aspx>

- <http://www.gotdotnet.com/team/dbox/default.aspx?key=2004-08-31T01:54:05Z>



How Do I Create a WSDL?

- 1. If you are a Java programmer you can cheat 😊
 - Create a Java interface and run Java2WSDL on it
- 2. If you are an XML Schema programmer
(can I hire you – you must be smart?)
 - Write the WSDL using notepad / vi / emacs
- 3. Use a WSDL editing tool:
 - WebSphere Studio, XML Spy
- 4. Use an alternative SOAP toolkit to create the WSDL!
 - gSOAP

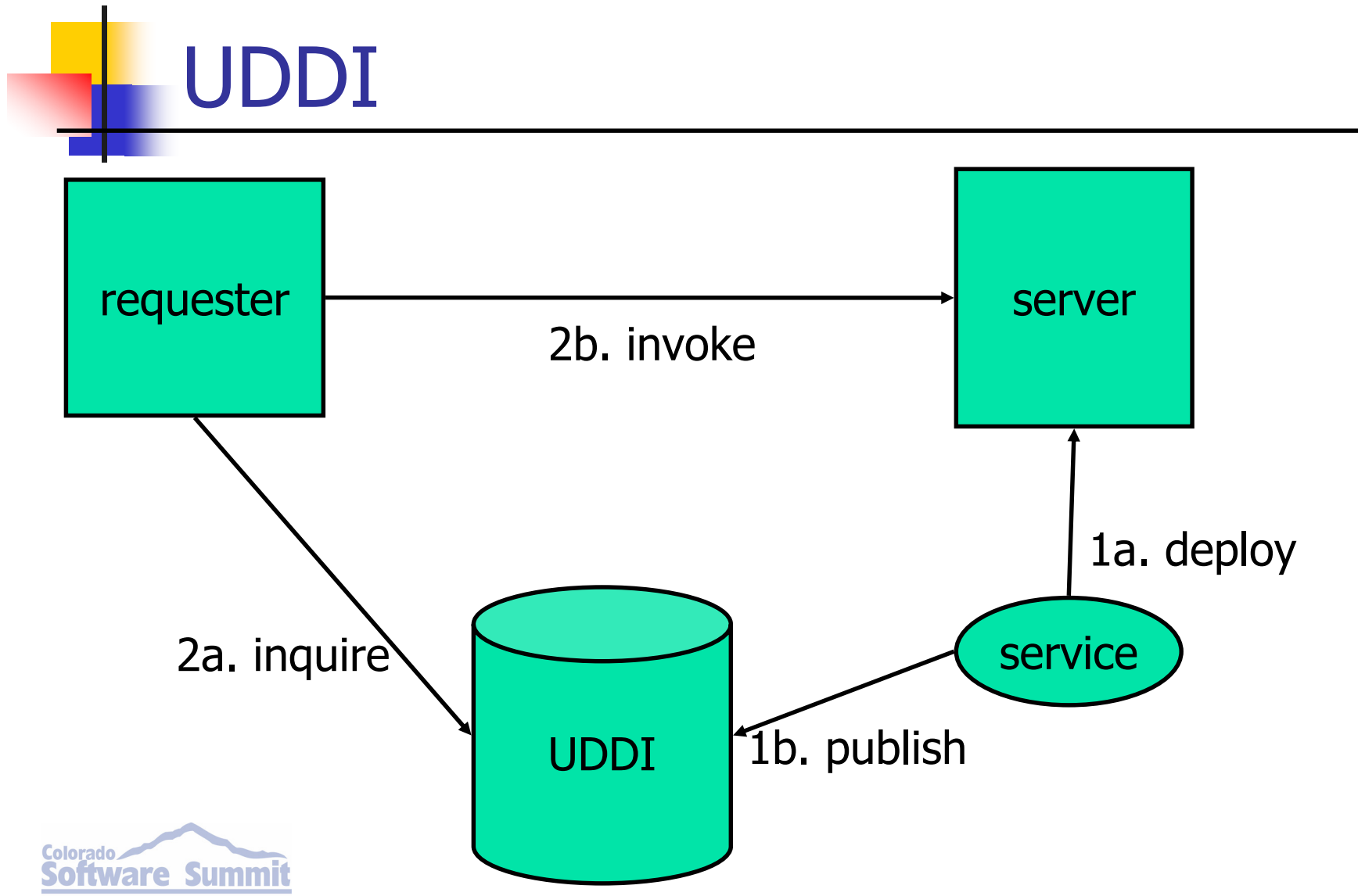




Using UDDI

- UDDI adds a level of “virtualisation” to the model
 - The actual target endpoints don’t have to be defined in the client code
 - The client does a lookup into UDDI and sets the correct endpoints







How to Use UDDI?

- With Java use UDDI4J
 - <http://Uddi4j.org>
- C++ compile the WSDLs using WSDL2WS
 - http://uddi.org/wsd/inquire_v2.wsdl
 - http://uddi.org/wsd/publish_v2.wsdl
 - Or similar v3 WSDLs available from UDDI.org
- Stub method:
 - `void AXISCALL setEndPoint(const char *pcEndPointURI);`



Issues with the Current Axis Codebase

■ Java

- 1.2 alpha at the time of writing
- Pretty stable and solid
 - Really a 2.x release as Apache SOAP was the 1.x codebase
- Used in a number of products as the SOAP engine
 - *e.g.* IBM WebSphere uses a heavily enhanced version

■ C/C++

- 1.3 alpha at the time of writing
- More like a 1.x codebase!
- C++ is usable, C is less well tested and stretched





SOAP HREFs

```
<soapenv:Body>  
  <ns1:echoIntegerResponse>  
    <return href="#id0"/>  
  </ns1:echoIntegerResponse>  
  <multiRef id="id0">42</multiRef>  
</soapenv:Body>
```

- Produced by Axis Java when using ENCODED (SOAP encoding)
 - <parameter name="sendMultiRefs" value="false"/> --- turns off
- NOT SUPPORTED by Axis C++
- Why?
 - WS-I does not support SOAP encoding



Handlers

- Handlers allow direct access to the SOAP message to perform additional processing
 - Key to implementing the SOAP processing model
 - *e.g.* WS-Addressing, WS-Security
 - Also can perform routing or logging





Handlers

```
#include <axis/server/Handler.h>

AXIS_CPP_NAMESPACE_USE

class THandler : public Handler
{
public:
    int AXISCALL fini();
    int AXISCALL init();
    void AXISCALL onFault(void* pvIMsg);
    int AXISCALL invoke(void* pvIMsg);
    void setOptionList(const map<string, string>* OptionList);
    const string& getOption(const string& sArg);
    THandler();
    virtual ~THandler();

protected:
    string m_sEmpty;

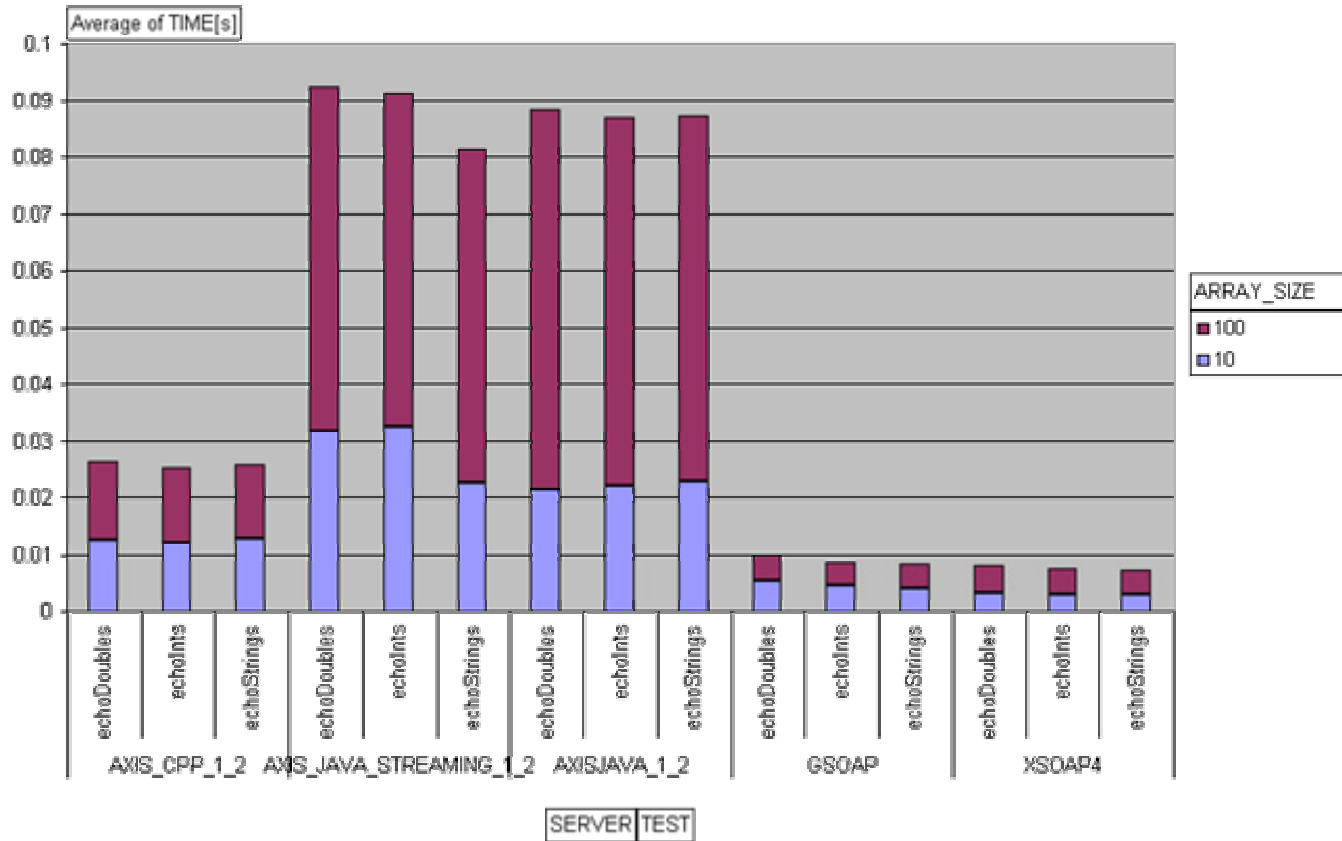
};
```



Performance

- Limited performance results available (Thanks to Alek Slominski)

MACHINE_NAME | RH73_VERONICA_LOOPBACK | CLIENT | XSOAP4





Futures

- Axis Java 1.x is being stabilised and work is starting on Axis Java 2.0
 - <http://wiki.apache.org/ws/FrontPage/Axis2>
- Axis C/C++ 1.4 will be next release
 - Aims:
 - Improved memory allocation
 - Cleaner separation of C and C++ to provide a cleaner memory model
 - Better support for threading
 - Continuous improvement of support for WSDL/Schema variants



Futures *(Continued)*

- Axis 2.0 design is common between C++ and Java
 - Aim to start on C++ once 1.x has stabilised and is useful
- Performance should be very fast...
 - Type Specific Pull Parser





Pull Parsers

- DOM
 - Read complete document into memory
- SAX
 - Register event handlers with parser, called back as each new element is handled
- Pull
 - Ask for each new event in a stream
- Typed Pull
 - Ask for next Integer, Long, *etc.*





More Information

- Axis homepage
 - <http://ws.apache.org/axis>
- Pull parsers
 - <http://xmlpull.org/>



Resources

- IBM developerWorks WebServices Zone
 - <http://www.ibm.com/developerworks/webservices/>
 - All the IBM / MS standards
- “The Hidden Impact of WS-Addressing”
 - <http://www-106.ibm.com/developerworks/webservices/library/ws-address.html>
- *Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI (2nd Edition)*
 - **Publisher:** Pearson Education;
 - **ISBN:** 0672326418

