Tutorial I

Web Services

1. What is a Web Service?
2. An example Web Service
3. OmniEditor: Wrapping a text editor into a WS
4. OmniGraphEditor: supporting a graphic editor

References

OmniEditor is a result of the course project for CSC408H.
What is Web Service

• Historic perspectives
  – Centralized versus Distributed Programming Models: Client/Server, Master/Slave, SPMD, etc…
  – Middleware architecture
    2-tier, 3-tier, n-tier
  – Technical evolution
    • IPC (inter process call):
      RPC, Java RMI, CORBA, DCOM
    • Shared Memory, Distributed Shared Memory and Message Passing

• Interoperability, interoperability, interoperability
What is a Web Service?

• Web Services Standards (XML-based)
  – SOAP (Simple Object Access Protocol)
  – WSDL (Web Service Description Language)
  – UDDI (Universal Description, Discovery, Integration)
  – WS-I (integration), WS-Policy, WS-Security, etc.

• Chances and Challenges
  – Application Wrappers to WS
  – Web Services Compositions
  – Holy Grail: large reusable library of WS with interoperability:
    1,000 public WS ~ 1 Million private WS
  – ROI (Return of Investments): > 10x ?
  – Issues: Privacy, Security, Performance
An example web service

*xmethods* provides a web services to get the stock price

- Example SOAP messages
- Structure of WSDL descriptions
- A program that invokes the web service
SOAP architecture

Service requestor

application object (OpenOME)

SOAP-based middleware

Service provider

application object (OmniGraphEditor)

SOAP-based middleware

request

response
Example

SOAP message: request

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:ns="urn:xmethods-delayed-quotes">
  <ns:getQuote>
    <symbol>IBM</symbol>
  </ns:getQuote>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
Example

SOAP message: response

```xml
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
 xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns:ns="urn:xmethods-delayed-quotes">
 <SOAP-ENV:Body
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/
  <ns:getQuoteResponse>
   <Result>86.05</Result>
  </ns:getQuoteResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```
WSDL architecture

WSDL (url)

WSDL compiler (client)

WSDL compiler (server side)

Service requestor

stub

SOAP-based middleware

Service provider

stub

SOAP-based middleware

response

request
Example

WSDL structure: message

```xml
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="quote"
    xmlns:tns="urn:xmethods-delayed-quotes"
    ... name spaces ...
    ... types ...
    <message name="getQuoteRequest">
        <part name="symbol" type="xsd:string"/>
    </message>
    <message name="getQuoteResponse">
        <part name="Result" type="xsd:float"/>
    </message>
    ... portType, binding, service ...
</definitions>
```
Example

WSDL structure: portType

```xml
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="quote"
    … name spaces …>
    … types, messages, …
    <portType name="quotePortType">
        <operation name="getQuote">
            <input message="tns:getQuoteRequest"/>
            <output message="tns:getQuoteResponse"/>
        </operation>
    </portType>
    … binding, service …
</definitions>
```
Example

WSDL structure: binding

```xml
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="quote"
   ... name spaces ...
   ... types, messages, portType ...
   <binding name="quote" type="tns:quotePortType">
      <SOAP:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="getQuote">
         <SOAP:operation soapAction=""/>
         <input>
            <SOAP:body use="encoded" namespace="urn:xmethods-delayed-quotes"
               encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
         </input>
         <output>
            <SOAP:body use="encoded" namespace="urn:xmethods-delayed-quotes"
               encodingStyle="http://schemas.xmlsoap.org/soap/encoding"/>
         </output>
      </operation>
   </binding>
   ... service ...
</definitions>
```
Example

WSDL structure: service

```xml
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="quote"
    ... name spaces ...
    ... types, messages, portType, binding ...

    <service name="quote">
        <port name="quote" binding="tns:quote">
            <SOAP:address location="http://services.xmethods.net/soap"/>
        </port>
    </service>

</definitions>
```
Web Services Invocation

```
<ns:getQuote>
  <symbol>IBM</symbol>
</ns:getQuote>

<ns:getQuoteResponse>
  <Result>86.05</Result>
</ns:getQuoteResponse>
```
Invoking the web service

gsoap/soapcpp2/samples/quote

//gsoap ns service name: quote
//gsoap ns service style: rpc
//gsoap ns service encoding: encoded
//gsoap ns service namespace: urn:xmethods-delayed-quotes
//gsoap ns service location: http://services.xmethods.net/soap
int ns__getQuote(char *symbol, float *Result); /* quote.h */
/* quote.c */
int main(int argc, char **argv)
{
    struct soap soap;
    float q;
    char *sym;
    if (argc > 1)
        sym = argv[1];
    else
    {
        fprintf(stderr, "Usage: quote <ticker>\n");
        exit(1);
    }
    soap_init(&soap);
    if (soap_call_ns__getQuote(&soap, "http://services.xmethods.net/soap", ",", sym, &q) == 0)
        printf("\nCompany - %s Quote - %f\n", sym, q);
    else
    soap_print_fault(&soap, stderr);
    soap_end(&soap);
    soap_done(&soap);
    return 0;
}

%> quote IBM
Company - IBM   Quote - 86.269997
%>
Whenever upload is called, the state of the channel in the server is updated or a new channel is created, and only when the editors need to refresh the buffer, they pull the information (downloading). Make sure the difference between downloaded buffer and the current buffer can be displayed.
Whenever upload is called, inform other editors involved in the channel to refresh the changed state.
The OmniEditor Web Service

• Tools you need: Axis-1_1, Jakarta Tomcat server
  /u/prof/yijun/software/axis-1_1
  /u/prof/yijun/software/jakarta-tomcat-5.0.28
• A few implementations of the OmniEditor
  http://skywolf.cdf.toronto.edu:8081/axis
• We put them in the ECF
  /u/prof/yijun/axis
  /u/prof/yijun/deployed
• Phase A. You need to study one of them to understand how it works.
• Phase B. You can develop your own, or modify the existing ones to a number of operations for the graph editor
• Phase C. You need to study invoke the web service in your graph editor.
Deployed Web Services of CSC408H

Here is a list of published web services by the end of phase B:

- Team 01's Homepage (WSDL), forum, supporter email
- Team 02 (WSDL)
- Team 03 (WSDL)
- Team 04's home page (WSDL)
- Team 05's web service (WSDL)
- Team 06's Web Page (WSDL)
- Team 08's Home Page (WSDL)
- Team 09 (cancelled support?)
- Team 10's Home page (WSDL)
- Team 11's web service (WSDL)
- Team 12's web site (WSDL) (support email)
- Team 14's web service (WSDL) support email
- Team 15's Home page (WSDL)

- View the list of Web services deployed at this Tomcat/Axsis server
- Call a local endpoint that list's the caller's http headers (or see its WSDL).

Validating Axis

- Validate the local installation's configuration
- Administer Axis
- SOAPMonitor
An example requirements specification

```java
float getQuote(String name);
    // precondition: name = ticker symbol
    // postcondition: return -1 if name does not exist

float getQuote(String name);
    // precondition: name = part of the full name
    // postcondition: return -1 if name doesn’t exist,
    //                -2 if multiple matches
```

What are the Goal (purpose), Input, Output, Pre/post-conditions and Exceptions?
Required functions for the OmniGraphEditor Web Services

- Upload
- Download
- Insert
- Delete
- Select (highlight)

For each function, you may have several alternatives. You shall make decision based on non-functional requirements (quality attributes)