Your document for this project will be submitted to BlackBoard

(Assignment 3) in one of the following file formats: ASCII, MS Word, PDF or ZIP

Refer to the course calendar for required submission date and syllabus for %.

Additional information on this project in the discussion forum of the class.

General Information:

Make sure you clearly understand chapter 1 terminology, it will allow for a much easier implementation of this project. Web Service is made up of the basic core services, HTTP/JMS/SNMP, XML, SOAP, and WSDL.

There is also UDDI which is used for service location, although not necessary of this project.

This project will be an implementation of these basic services listed above. Please refer to figure 4.1 to understand the process of message movement, section 4.4 for the communication models. For the understanding of WSDL please refer to chapter 5, figure 5.1 for an overview.

Prior to the main project there are three component exercises that will help with your understandings of XML, SOAP and WSDL

Complete this component exercises first from your textbook:

Exercise 3.4 - XML 10% of project Grade

Exercise 4.1 - SOAP 10% of project Grade

Exercise 5.3 – WSDL 10% of project Grade

Project Description: 70% of your project Grade

You are a linguistic working at an e-books distributor. Your management has interest in finding out if commonly used words used in a book makes the books more popular, therefore more books would be sold if the commonly used words in the book was higher. They believe that by knowing this beforehand would allow them to tailor their sales and marketing more closely to what would be purchased rather than just hoping because of the popularity of the author or other factors, would sell a book. So they want to test their theory by creating a web-services application that allows their customers to submit books to check the popularity of the words used in the book, thereby ranking the book as high, medium or low ranking, and then comparing that to the books ranking in sales.

You will create a RPC and a Document-style Web service to process a document (simple text). The processing done by the web service will be the following:

**Word Count**: Simply the number of words in the document

**Word Rank:** You may use many of the internet sites that have listing of the most commonly used words, words that are not in the list will be ranked lowest (depending on your table length)

**Word Repeatability**: – all words counted as to the number of time they are used in the document

For example of word rank:

Rank Word

1. the
2. of
3. to
4. and
5. a
6. in
7. is
8. it
9. you
10. that

You will use the test document below to test the three actions your web service will take on a document

The document message is as follows:

“Message-oriented choreography

The Paper Trail concept is that the state of a multi agent multi-process system can be looked at, sometimes rather effectively, as a function of the documents which have been transmitted.

The process-oriented attitude to a bank-customer relationship may be "In parallel, the customer writes checks, merchants pay in checks, credit card transactions happen, all month. Then, the charges, interest are assessed and a bank statement sent from the bank to the customer". The document-, or message-oriented one is more like "Every month a bank balance lists valid transaction dated that month. A cleared incoming check in a valid transaction. A cleared outgoing check is a valid transaction. A validated credit card debit is a valid transaction. A check is cleared if it is incoming and there is a matching transfer from the payee bank", and so on. This builds the relationships up in a bottom-up, web like way. The process-oriented attitude suggests the bank be written as a procedure in a top-down way using for example WSCI and BPL. The document-oriented attitude suggests the use of business rules systems triggered by the receipt of new information -- new documents, in this case new web services messages.

(Web service messages are of course documents just like documents sent in email. Messages are particular in that they have a particular time of transmission, and their document content does not change. They do of course generally have identifiers, and even though they can only be accessed by sender and explicit receivers, they can still be regarded as part of the web by those parties.)

Whether the design process is a top-down process-oriented one or a bottomup document-oriented one, the design will have to be translated into a set of agents and their responses to incoming messages. This manipulation can of course be done automatically.

A concern in all this frantic design is it evolution with time. A BPEL script sets out to be a description of a business process at a high level. The critical values which decide on conditional execution, or which correlate a particular process with a given transaction, are expressed as parts of the structure of the XML messages. This may lead to what has been called "DTD fragility". What happens which you change the DTD? The design of the message types with XML schema is the sort of thing which is difficult to get everyone in a company to agree on, and tents to change with time. There are many arbitrary choices made as to how the knowledge in the message is serialized as XML. Moving to RDF may, by removing a layer of arbitrary design, reduce that fragility and allow web service choreography to evolve with time within and outside a company.”

Reference: <http://www.w3.org/DesignIssues/WebServices.html>

Error Handling:

The Mandatory fields that are to be added are:

Last name

First name

Address Street

Address city

Address state and zip code

When was the book purchased?

Where was the book purchased?

What is highest level of education?

Are you married or single?

Do you have children?

If you feel that more fields should be added you may add if you believe it will help in the research.

The results of the processing of the document must be sent back to the sender,

basically Client=>Web Service processing=>Client results

Additional information:

Error Handling:

Please implement the soap methodology for handling faults in which you use the special-purpose messages such as env:fault.

Development kits:

There are many: VS, Netbeans, Eclipse, Java SDK/JDK or EE, choose the one of your liking.

For example: Java EE 7 SDK with JDK 7

<http://www.oracle.com/technetwork/java/javaee/downloads/index.html>

Java EE 7 SDK with JDK 7 Update 21

A free integrated development kit used to build, test, and deploy Java EE 7 applications.

It includes :

* JDK 7 Update 21
* GlassFish Open Source Edition 4.0
* Java EE 7 Code Samples
* Java EE 7 API Documentation
* Java EE 7 Tutorial
* Your First Cup: An Introduction to the Java EE Platform

Most coding does not require these development kits but they are usually helpful for cohesion

Deliverables

1. The three exercises that precluded the project component in source coded and examples of the output
2. You must provide a methodology and design specification that shows how you defined the solution of the project
3. You must include all source code on both sides, client and web service side
4. You must provide instructions for running the application and how your tested
5. You must provide a result log or listing of the message being processed (as returned to the client)
6. Submit your project in ZIP file to BlackBoard under the assignment, with clearly defined parts. The submission will include Word or Text files and the source files, including any external reference to internet sites. Each deliverable must be clearly labeled, including your name and course title.