<u>**Lab 7: WSDL Based: Implement Arithmetic Service that implements Add, and Subtract operations**</u>

To implement an ArithmeticService with WSDL (Web Services Description Language) that provides two operations, "add" and "subtract," you need to create a WSDL file and then implement the service using a programming language and a web service framework. Here is an basic example of how to do this using Java and Apache CXF. Here are the steps:

WSDL, which stands for Web Services Description Language, is an XML-based language used to describe the functionality of web services. It serves as a contract or interface definition that specifies how to interact with a web service. WSDL is a fundamental component in the world of web services and plays a crucial role in enabling interoperability between different systems and programming languages.

Key features and components of WSDL include:

Service Definition: WSDL defines a web service, including the service's name, operations, and the input and output messages for each operation.

Operations: Each operation in the WSDL describes a specific action or functionality provided by the web service. These operations are analogous to methods or functions in traditional programming.

Input and Output Messages: For each operation, WSDL specifies the structure of the input and output messages, including the data types and elements involved. This allows clients to understand what data to send and what to expect in response.

Data Types: WSDL can define complex data types using XML Schema (XSD). These data types can be reused across different parts of the WSDL document.

Port Types: A port type describes a set of operations that a service provides, including their input and output messages. It's a logical grouping of related operations.

Bindings: WSDL bindings define how the service is accessed, typically specifying the communication protocol (e.g., SOAP, HTTP) and message format (e.g., XML). Different bindings can be defined for the same port type, allowing for various access methods.

Services: A service groups a set of related ports together and defines the physical location or endpoint URL where the service can be accessed.

Here's a basic example of a simple WSDL document:

Create a WSDL file (arithmetic.wsdl) to describe your service. This file defines the service operations and their input/output parameters. Save this file as "arithmetic.wsdl":

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:tns="http://example.com/arithmetic"
    targetNamespace="http://example.com/arithmetic">

    <wsdl:types>
        <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://example.com/arithmetic">
            <xs:element name="addRequest">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="a" type="xs:int"/>
                        <xs:element name="b" type="xs:int"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="addResponse">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="result" type="xs:int"/>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
            <xs:element name="subtractRequest">
                <xs:complexType>
                    <xs:sequence>
                        <xs:element name="a" type="xs:int"/>
                        <xs:element name="b" type="xs:int"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="subtractResponse">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="result" type="xs:int"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:schema>
</wsdl:types>

<wsdl:message name="addRequestMessage">
    <wsdl:part name="parameters" element="tns:addRequest"/>
</wsdl:message>

<wsdl:message name="addResponseMessage">
    <wsdl:part name="parameters" element="tns:addResponse"/>
</wsdl:message>

<wsdl:message name="subtractRequestMessage">
    <wsdl:part name="parameters" element="tns:subtractRequest"/>
</wsdl:message>

<wsdl:message name="subtractResponseMessage">
    <wsdl:part name="parameters" element="tns:subtractResponse"/>
</wsdl:message>
```

```
<wsdl:portType name="ArithmeticPortType">
    <wsdl:operation name="add">
        <wsdl:input message="tns:addRequestMessage"/>
        <wsdl:output message="tns:addResponseMessage"/>
    </wsdl:operation>
    <wsdl:operation name="subtract">
        <wsdl:input message="tns:subtractRequestMessage"/>
        <wsdl:output message="tns:subtractResponseMessage"/>
    </wsdl:operation>
</wsdl:portType>

<wsdl:binding name="ArithmeticBinding" type="tns:ArithmeticPortType">
    <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="add">
        <soap:operation soapAction="http://example.com/arithmetic/add"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="subtract">
        <soap:operation soapAction="http://example.com/arithmetic/subtract"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
```

```
          <wsdl:output>
            <soap:body use="literal"/>
          </wsdl:output>
        </wsdl:operation>
      </wsdl:binding>

      <wsdl:service name="ArithmeticService">
        <wsdl:port name="ArithmeticPort" binding="tns:ArithmeticBinding">
          <soap:address location="http://localhost:8080/arithmetic"/>
        </wsdl:port>
      </wsdl:service>
    </wsdl:definitions>
```

Java program for WSDL based: Implement ArithmeticService that implements add, and subtract operations:

To implement an ArithmeticService in Java based on the WSDL, you can use the JAX-WS (Java API for XML Web Services) library. First, you need to generate Java classes from the provided WSDL, and then you can implement the service operations. Here's a step-by-step guide:

Generate Java classes from the WSDL using the wsimport tool (included with Java):

Assuming you have your "arithmetic.wsdl" file, open a command prompt and navigate to the directory containing the WSDL file. Use the following command to generate Java classes:

```
wsimport -d generated -s generated -p com.example.arithmetic
http://example.com/arithmetic/arithmetic.wsdl
```

This command will create Java classes in the "generated" package based on the WSDL definitions.

Create an implementation of the ArithmeticService:

```
package com.example.arithmetic;
```

```
import javax.jws.WebService;
@WebService(endpointInterface = "com.example.arithmetic.ArithmeticPortType")
public class ArithmeticService implements ArithmeticPortType {
    @Override
    public AddResponse add(AddRequest parameters) {
        int result = parameters.getA() + parameters.getB();
        AddResponse response = new AddResponse();
        response.setResult(result);
        return response;
    }

    @Override
    public SubtractResponse subtract(SubtractRequest parameters) {
        int result = parameters.getA() - parameters.getB();
        SubtractResponse response = new SubtractResponse();
        response.setResult(result);
        return response;
    }
}
```

Create a main class to publish the service:

```
package com.example.arithmetic;
import javax.xml.ws.Endpoint;
public class ArithmeticServicePublisher {
    public static void main(String[] args) {
        String address = "http://localhost:8080/arithmetic";
        ArithmeticService arithmeticService = new ArithmeticService();
        Endpoint.publish(address, arithmeticService);

        System.out.println("ArithmeticService is published at: " + address);

    }

}
```

Compile your Java classes.

Run the ArithmeticServicePublisher class to publish your ArithmeticService.

Your service is now published and ready to accept requests. You can access it by using the WSDL definition and SOAP requests. Be sure to replace the package and URL references with your specific values as needed.