

# Contract First Approach for Creating SOAP Web Service

Prof. (Dr.) Vipul K. Dabhi  
Assoc. Professor,  
Department of Information Technology,  
D. D. University

# Approaches for creating SOAP web-service

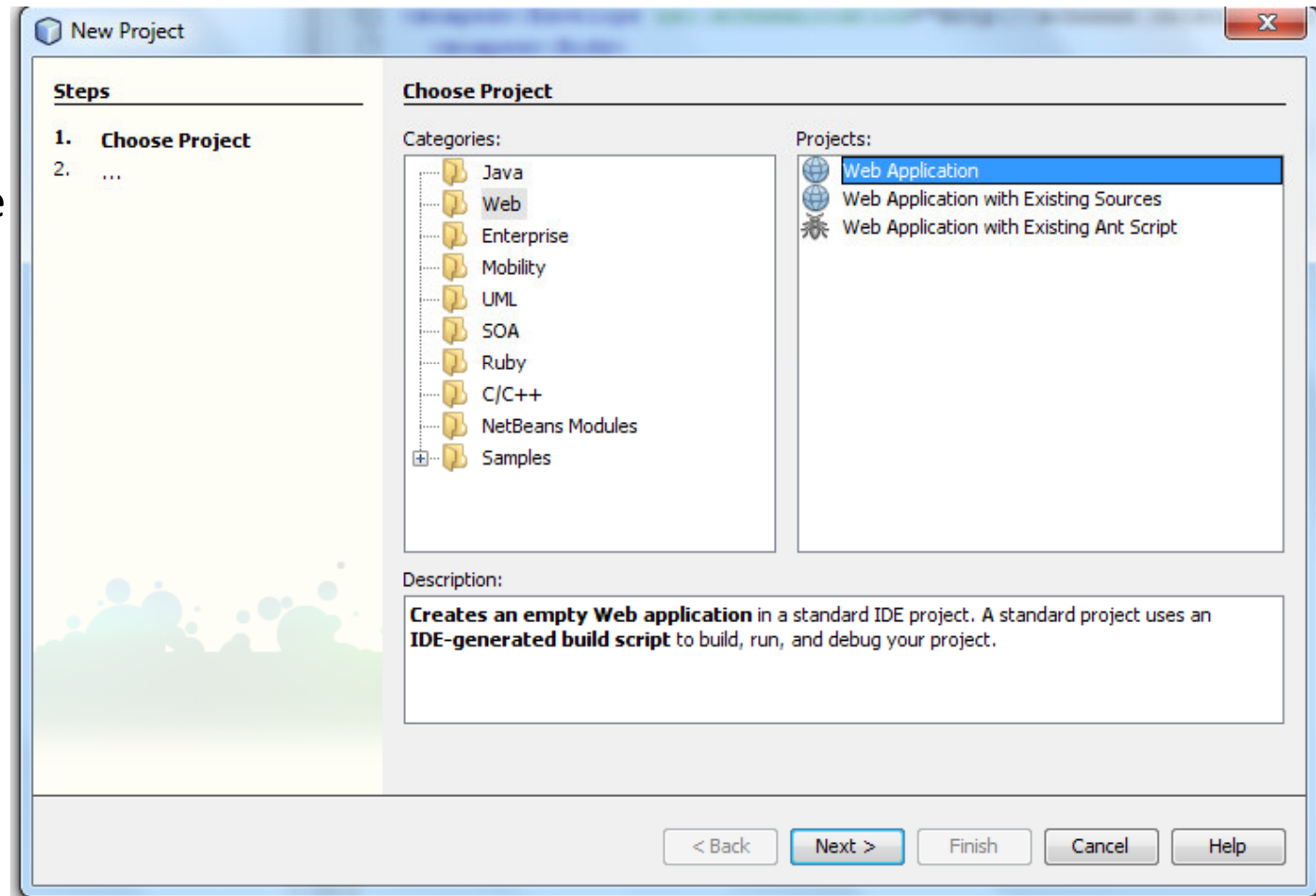
- **Code First Approach**
  - Writing source code for web service and then WSDL File
- **Contract First Approach**
  - Creating Web-service based on given WSDL File

# SECOND APPROACH FOR WEB-SERVICE CREATION

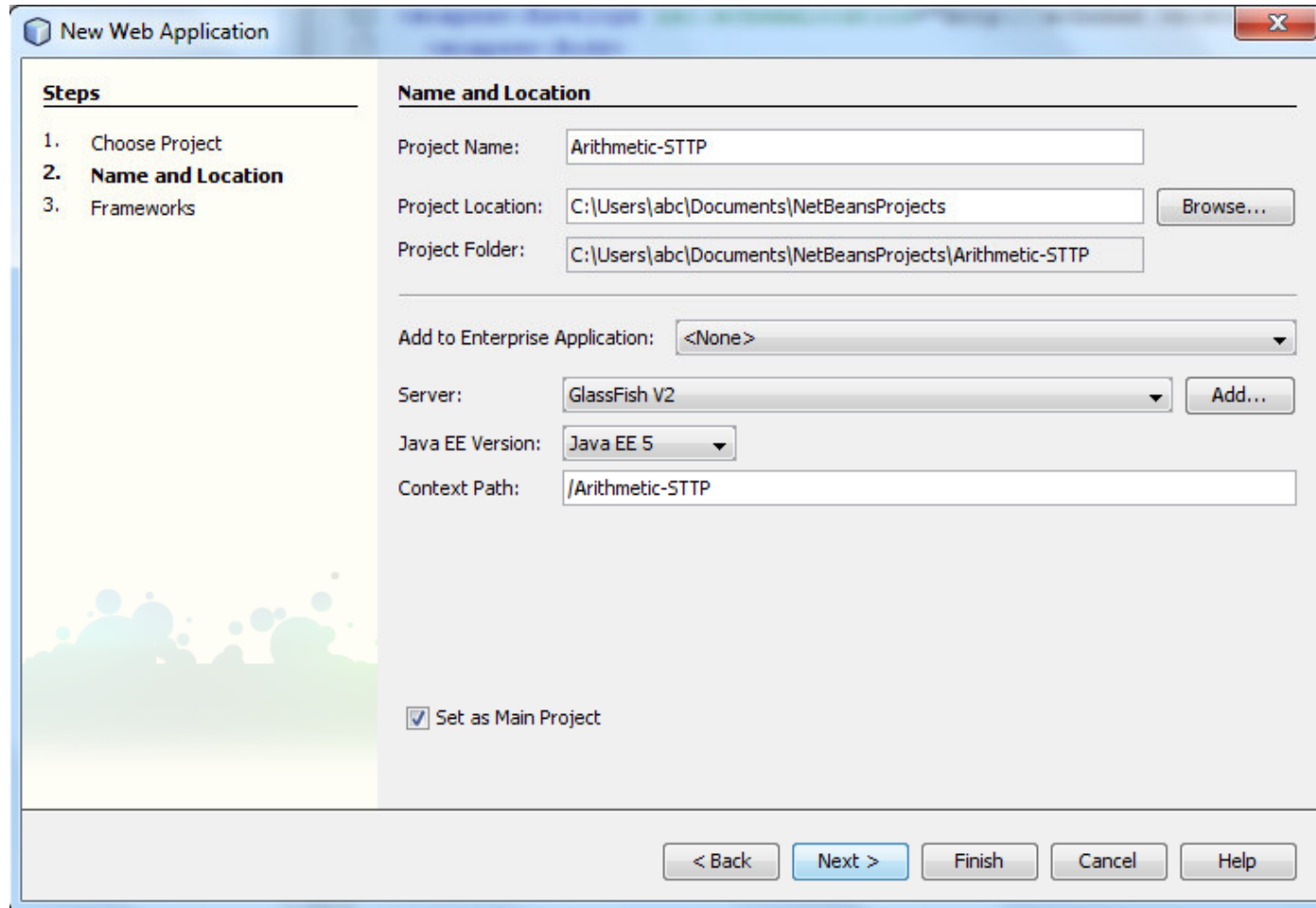
- CONTRACT FIRST APPROACH
  - BUILDING WEB-SERVICE BASED ON GIVEN WSDL FILE.

# Creating Web Service using Contract First Approach

- Create Web Application
  - Give it name Arithmetic



# Creating Web Service using Contract First Approach



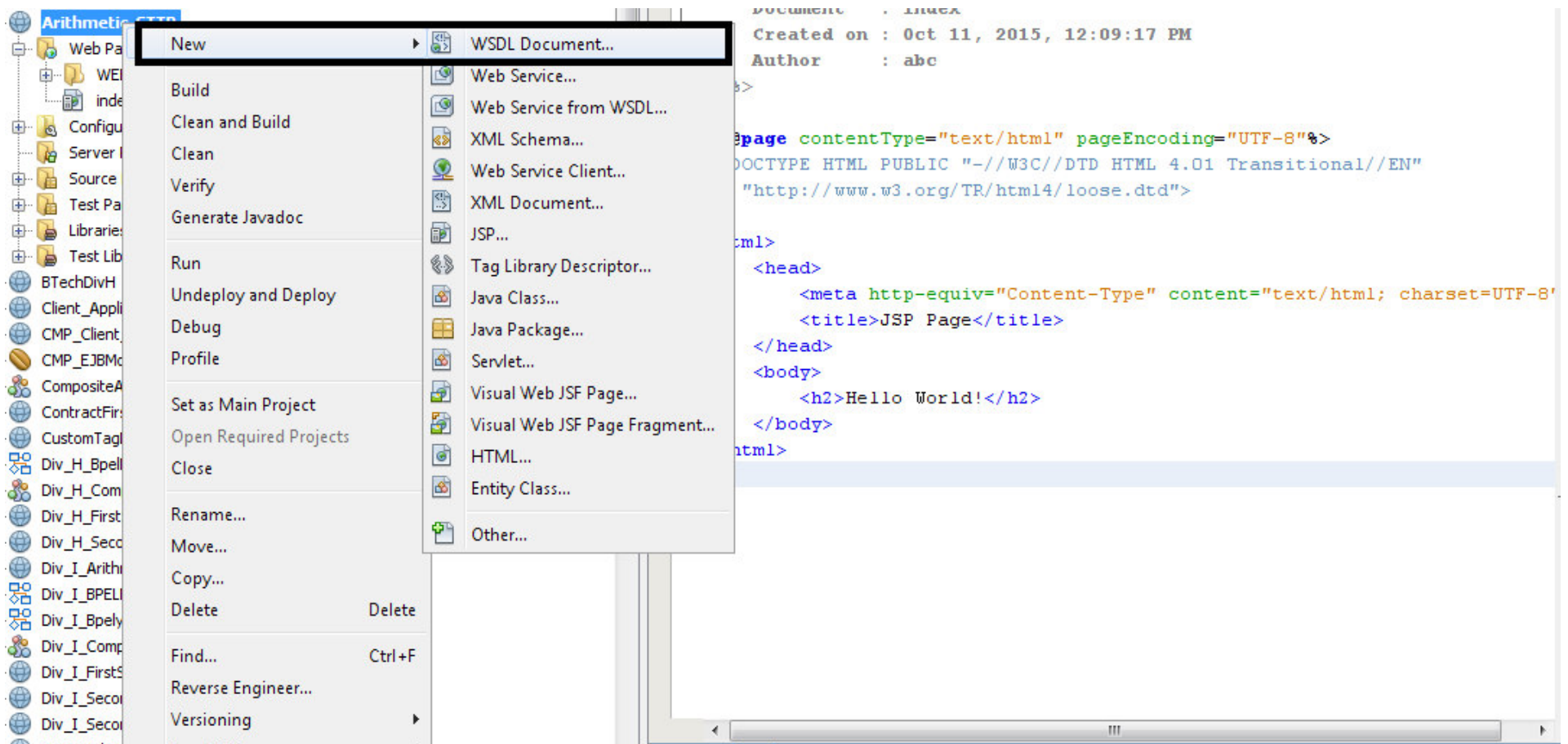
The screenshot shows the 'New Web Application' dialog box in NetBeans IDE. The 'Steps' pane on the left indicates the current step is '2. Name and Location'. The 'Name and Location' tab is active, showing the following fields:

- Project Name:** Arithmetic-STTP
- Project Location:** C:\Users\abc\Documents\NetBeansProjects (with a 'Browse...' button)
- Project Folder:** C:\Users\abc\Documents\NetBeansProjects\Arithmetic-STTP
- Add to Enterprise Application:** <None>
- Server:** GlassFish V2 (with an 'Add...' button)
- Java EE Version:** Java EE 5
- Context Path:** /Arithmetic-STTP
- ☒ Set as Main Project

At the bottom of the dialog are buttons for '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.

# Creating Web Service using Contract First Approach

- Right click on project folder and select New -> WSDL Document (other wise go to Other-> XML -> WSDL Document) Give it name STTPScientificWSDL



# Creating Web Service using Contract First Approach

The screenshot shows the 'New WSDL Document' dialog box in NetBeans IDE. The dialog is titled 'New WSDL Document' and has a close button (X) in the top right corner. On the left side, there is a 'Steps' panel with a list of four steps: 1. Choose File Type, 2. **Name and Location**, 3. Abstract Configuration, and 4. Concrete Configuration. The second step, 'Name and Location', is currently selected and highlighted. The main area of the dialog is divided into two sections. The top section, titled 'Name and Location', contains several input fields: 'File Name:' with the text 'STTPScientificWSDL', 'Project:' with the text 'Arithmetic-STTP', 'Folder:' with the text 'src\java' and a 'Browse...' button, and 'Created File:' with the text ':rs\abc\Documents\NetBeansProjects\Arithmetic-STTP\src\java\STTPScientificWSDL.wsdl'. The bottom section contains a 'Target Namespace:' field with the text 'http://j2ee.netbeans.org/wsdl/STTPScientificWSDL', an unchecked checkbox for 'Import XML Schema File(s)', and an 'XML Schema(s):' field with a 'Browse...' button. At the bottom of the dialog, there are five buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'. The 'Next >' button is highlighted in blue.

**Steps**

1. Choose File Type
2. **Name and Location**
3. Abstract Configuration
4. Concrete Configuration

**Name and Location**

File Name: STTPScientificWSDL

Project: Arithmetic-STTP

Folder: src\java Browse...

Created File: :rs\abc\Documents\NetBeansProjects\Arithmetic-STTP\src\java\STTPScientificWSDL.wsdl

Target Namespace: http://j2ee.netbeans.org/wsdl/STTPScientificWSDL

☐ Import XML Schema File(s)

XML Schema(s): Browse...

< Back **Next >** Finish Cancel Help

Contract First Approach for SOAP Web  
Service by Prof. Vipul K. Dabhi, D. D. Univ.

# Creating Web Service using Contract First Approach

- A Dialogbox will appear asking for “abstract configuration”

## Port name

STTPScientificWSDLPortType

## Operation name

Sin

## Operation Type

Request-Response

## Input

## Message Part

SinRequestPart

## Element

xsd:double

## Output

## Message Part

SinResponsePart

## Element

xsd:double

The screenshot shows a 'New WSDL Document' dialog box with a 'Steps' pane on the left and an 'Abstract Configuration' pane on the right. The 'Steps' pane lists four steps: 1. Choose File Type, 2. Name and Location, 3. Abstract Configuration (selected), and 4. Concrete Configuration. The 'Abstract Configuration' pane contains the following fields:

- Port Type Name: STTPScientificWSDLPortType
- Operation Name: Sin
- Operation Type: Request-Response Operation
- Input: A table with two columns: Message Part Name and Element Or Type. The first row contains 'SinRequestPart' and 'xsd:double'.
- Output: A table with two columns: Message Part Name and Element Or Type. The first row contains 'SinResponsePart' and 'xsd:double'.
- Fault: A table with two columns: Message Part Name and Element Or Type. It is currently empty.

At the bottom of the dialog box, there are buttons for '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'.



# Creating Web Service using Contract First Approach

The screenshot shows a dialog box titled "New WSDL Document" with a close button (X) in the top right corner. The dialog is divided into two main sections: "Steps" on the left and "Concrete Configuration" on the right.

**Steps:**

1. Choose File Type
2. Name and Location
3. Abstract Configuration
4. **Concrete Configuration**

**Concrete Configuration:**

Binding Name:

Binding Type:

Binding Subtype: ☒ RPC Literal  
☐ Document Literal  
☐ RPC Encoded

Service Name:

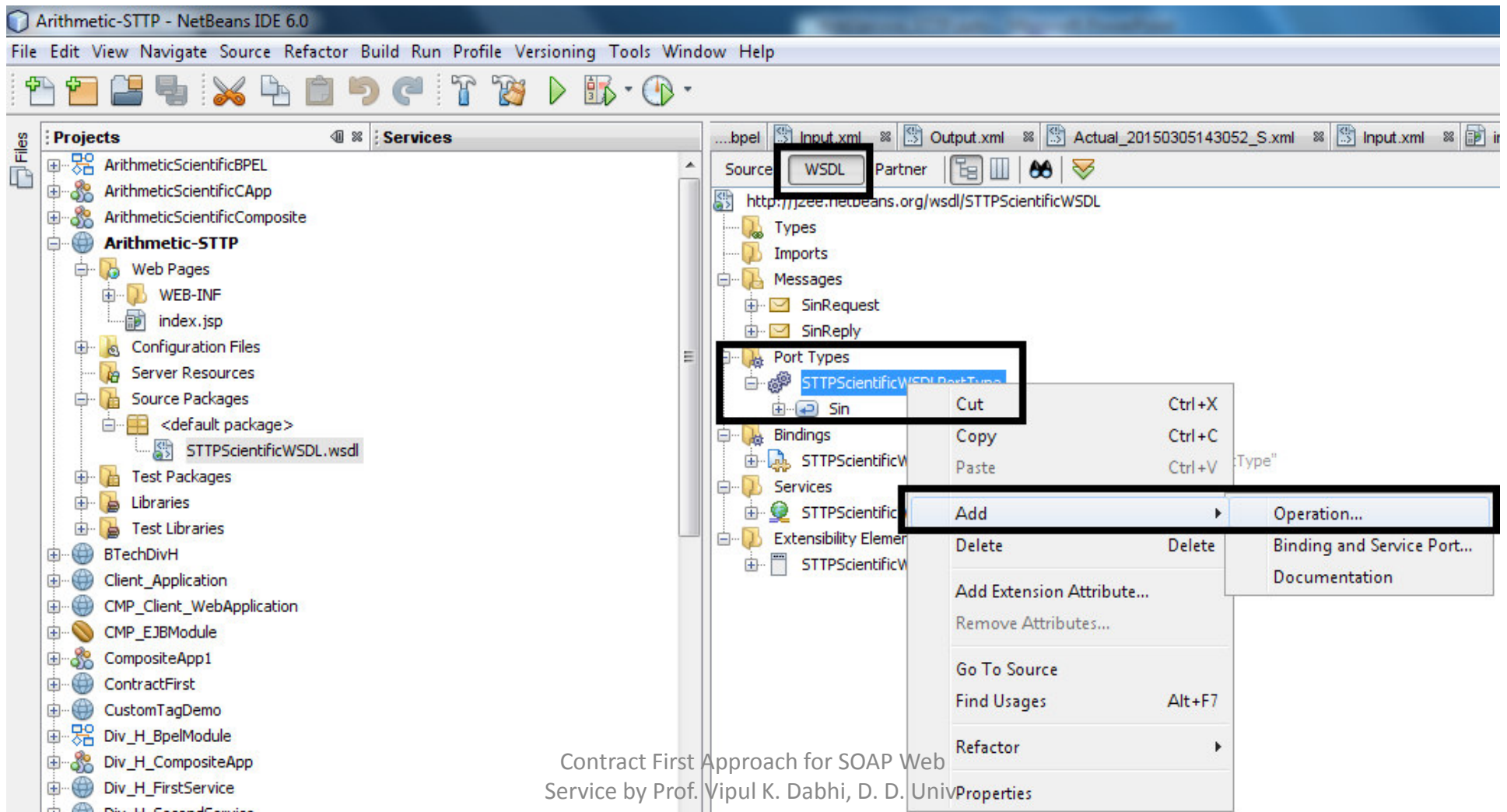
Port Name:

At the bottom of the dialog, there are five buttons: "< Back", "Next >", "Finish" (highlighted in blue), "Cancel", and "Help".

Contract First Approach for SOAP Web  
Service by Prof. Vipul K. Dabhi, D. D. Univ.

# Creating Web Service using Contract First Approach

- In WSDL view, right click Port Types->STTPScientificWSDLPortType ->Add->Operation



A dialog box “Create New Operation” will appear asking for configuration of new operation.

**Operation name**

Cos

**Operation Type**

Request-Response

**Input**

**Message Part**

CosRequestPart

**Element**

xsd:double

**Output**

**Message Part**

CosResponsePart

**Element**

xsd:double

Click Finish

**Create New Operation**

Operation Name: Cos

Operation Type: Request-Response Operation

Input:

CosRequest

| Message Part Name | Element Or Type |
|-------------------|-----------------|
| CosRequestPart    | xsd:double      |

Add Remove

Output:

CosResponse

| Message Part Name | Element Or Type |
|-------------------|-----------------|
| CosResponsePart   | xsd:double      |

Add Remove

Fault:

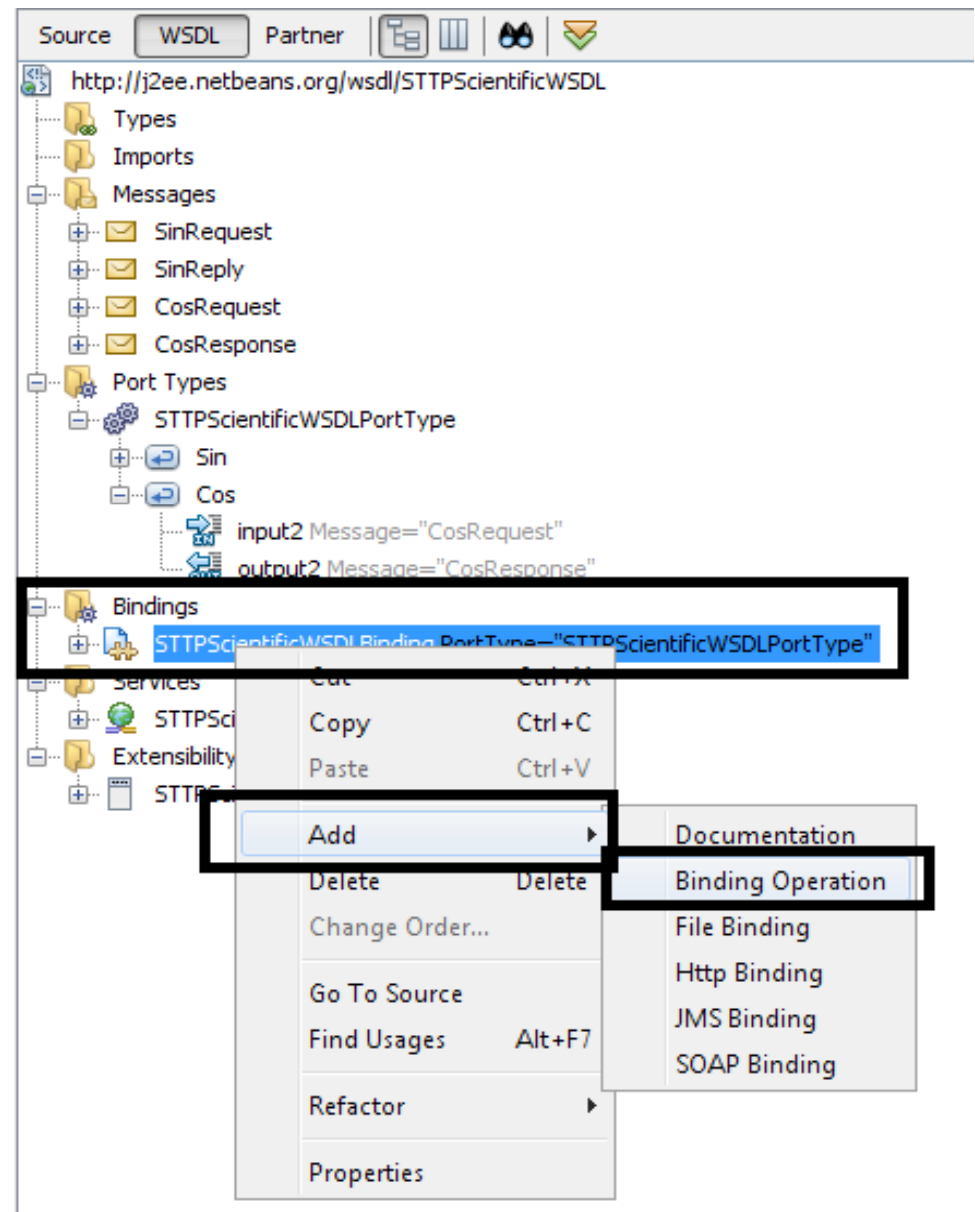
| Message Part Name | Element Or Type |
|-------------------|-----------------|
|-------------------|-----------------|

Add Remove

OK Cancel Help

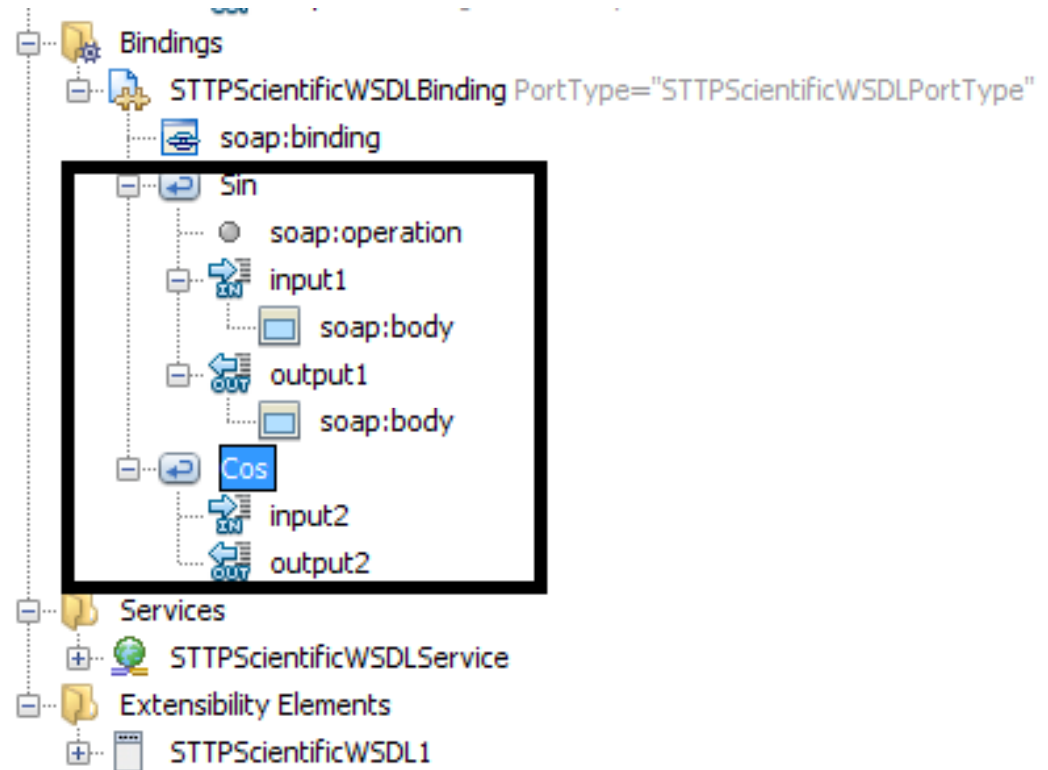
# Creating Web Service using Contract First Approach

**Right click Binding-  
>STTPScientificWSDLBinding->Add-  
>Binding Operation**



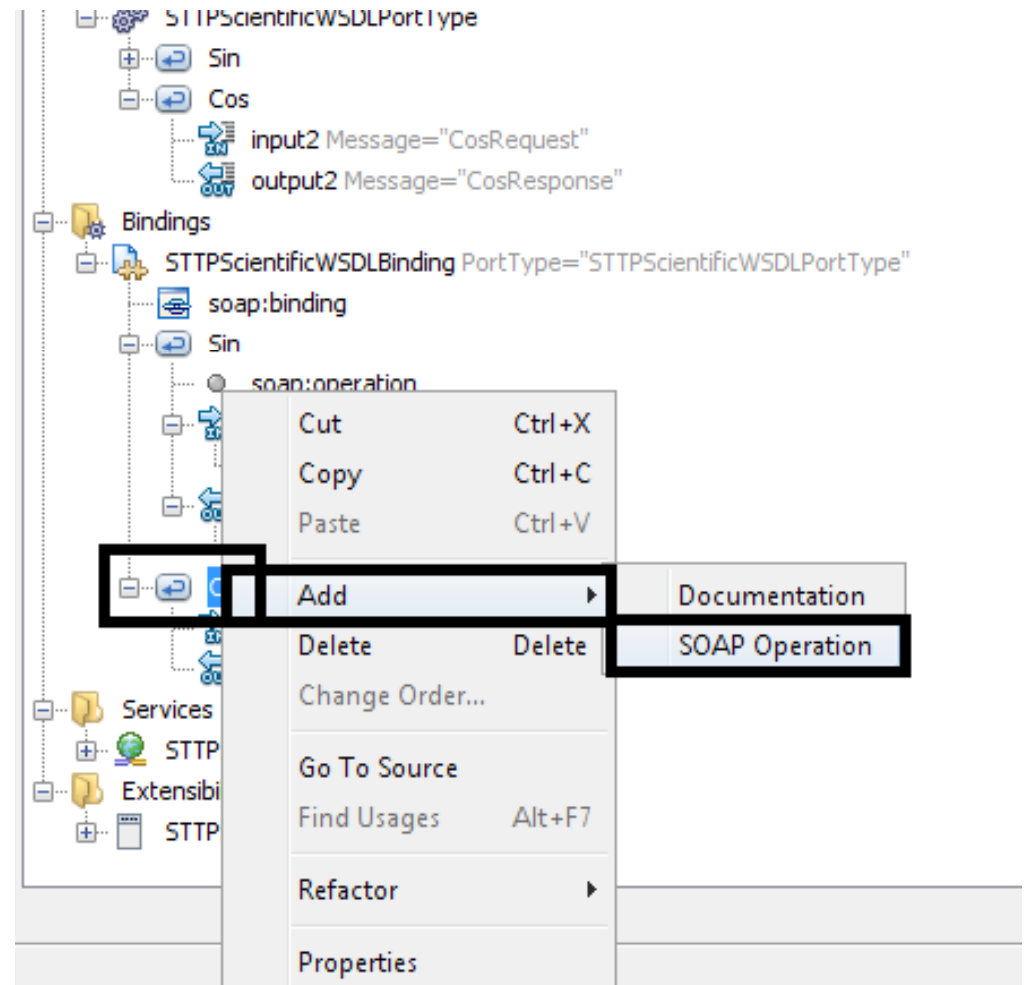
# Creating Web Service using Contract First Approach

**The input and output parts of Cos operation do not have soap:body**



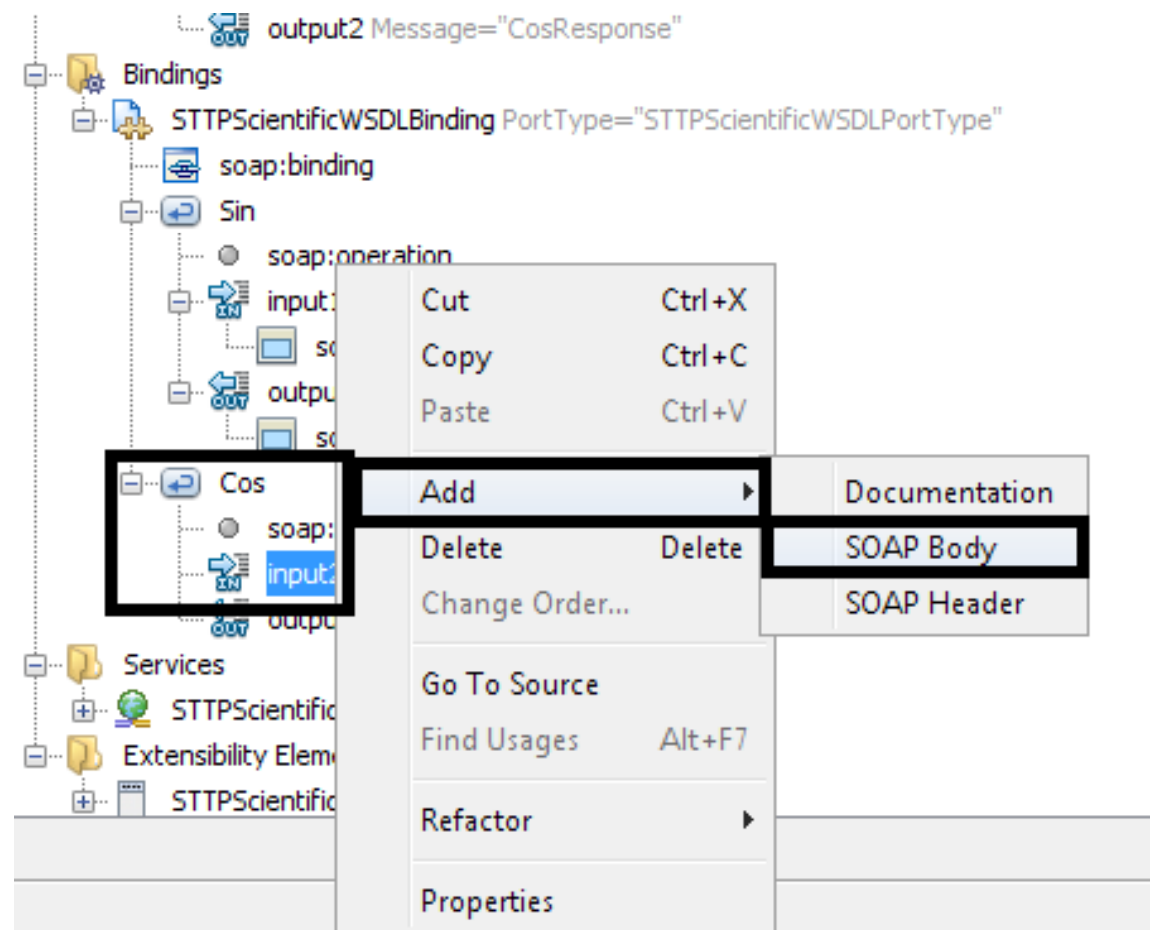
# Creating Web Service using Contract First Approach

**Right click Cos->Add->SOAP Operation**



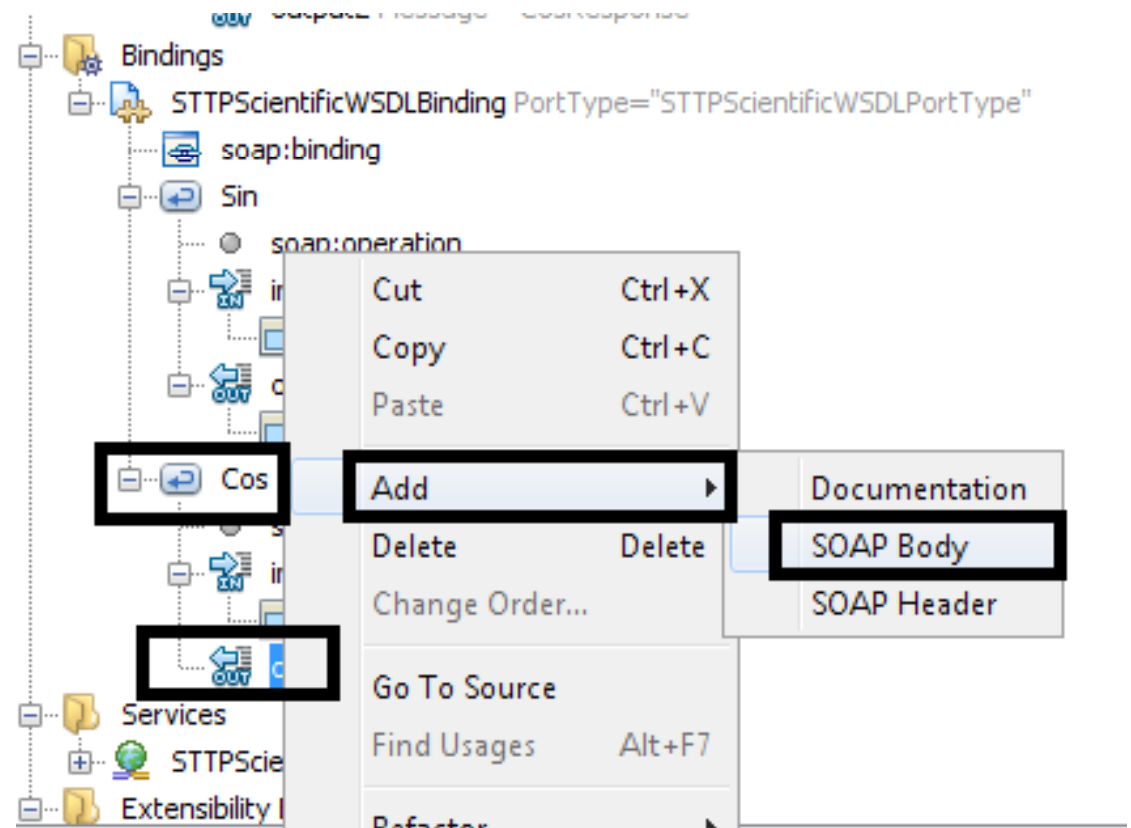
# Creating Web Service using Contract First Approach

**Right click Cos->input2->Add->SOAP Body**



# Creating Web Service using Contract First Approach

**Right click Cos->output2->Add->SOAP Body**

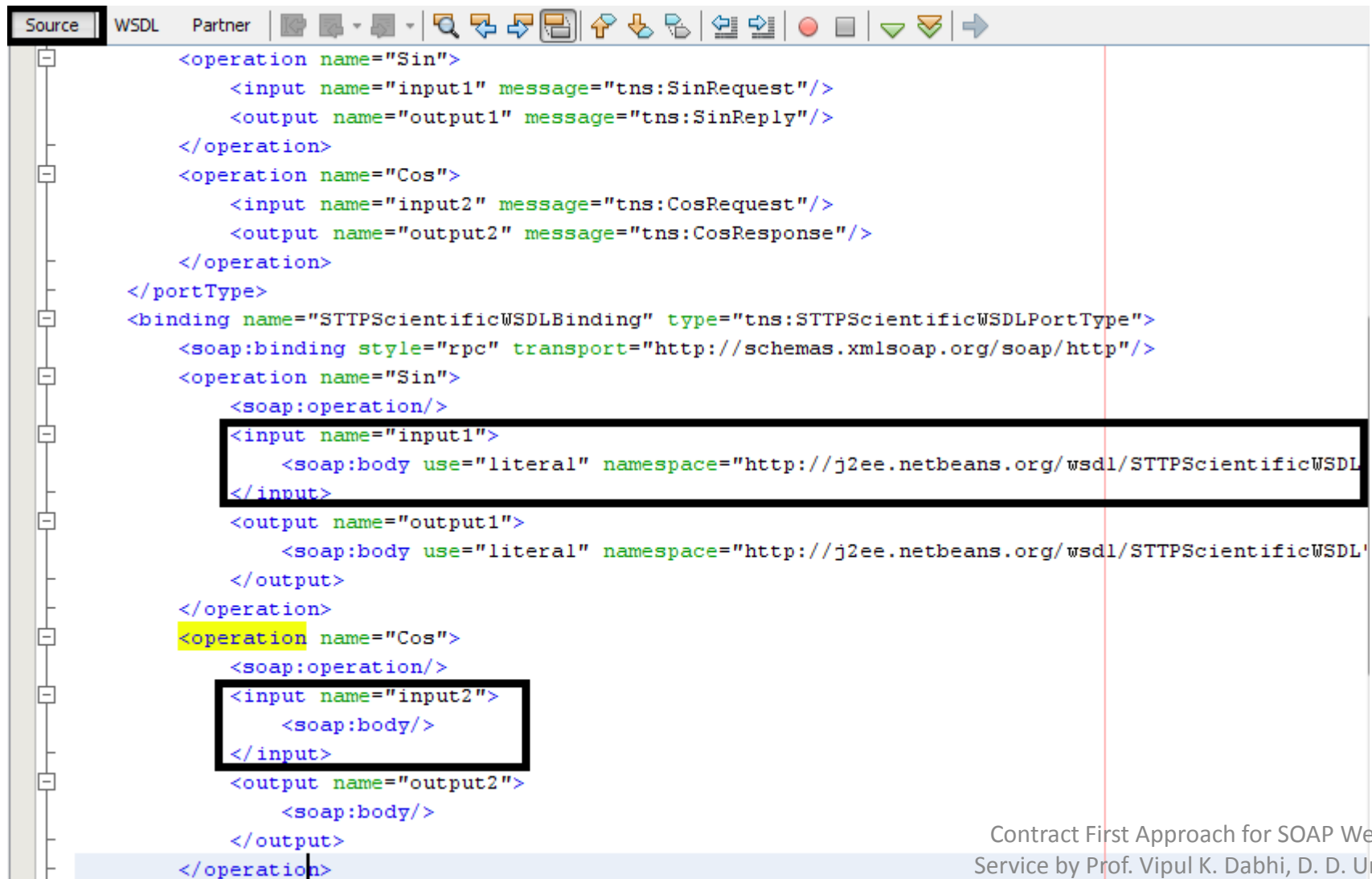




# Creating Web Service using Contract First Approach

Go in source view and add the following details:

`<soap:body use="literal" namespace=http://j2ee.netbeans.org/wsdl/STTPScientificWSDL/>`



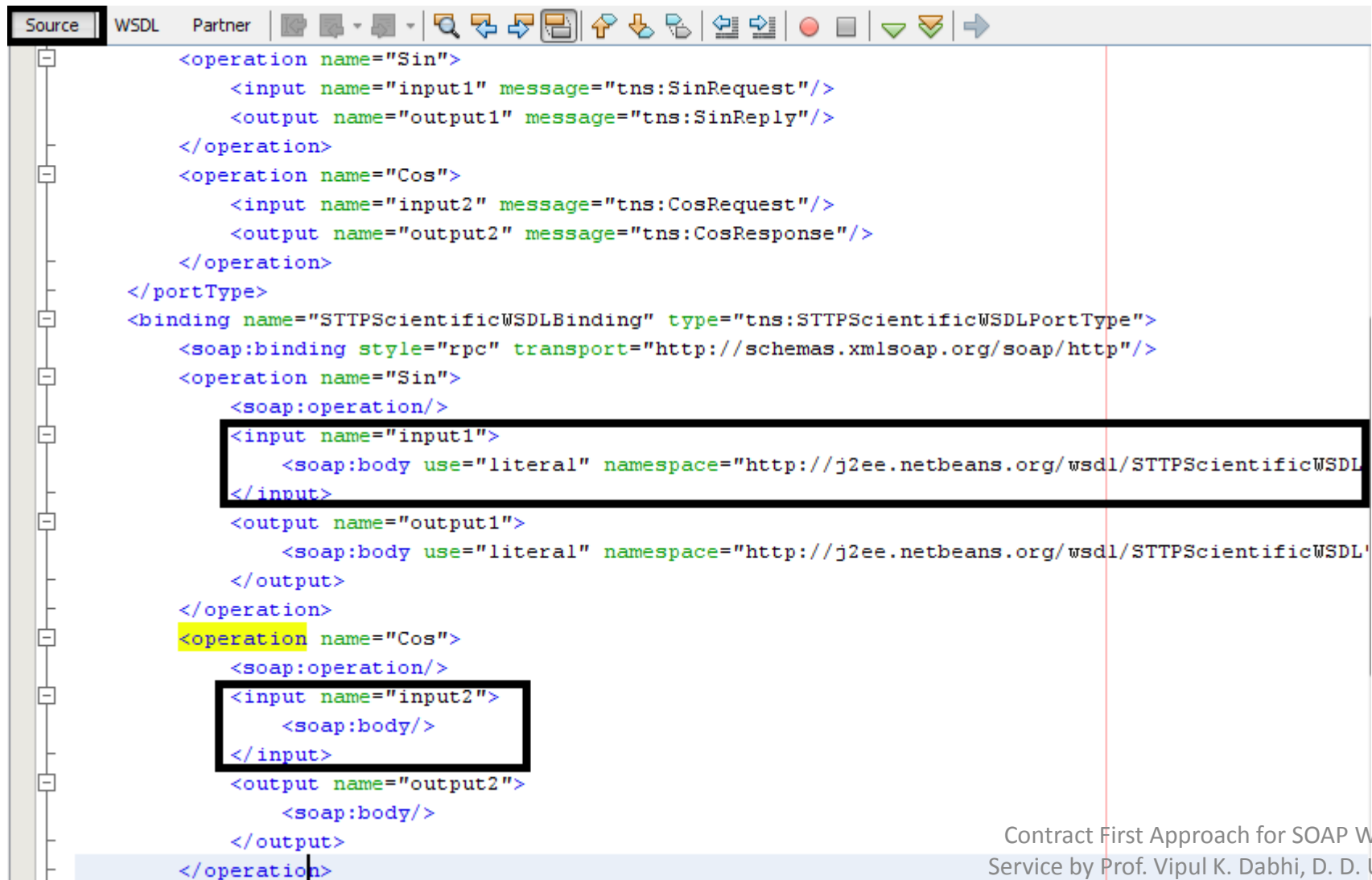
The screenshot shows an IDE window with the 'Source' tab selected, displaying WSDL code. The code defines two operations: 'Sin' and 'Cos'. The 'Sin' operation has an input 'input1' and an output 'output1', both with 'literal' body usage. The 'Cos' operation has an input 'input2' and an output 'output2', also with 'literal' body usage. The namespace for the body elements is 'http://j2ee.netbeans.org/wsdl/STTPScientificWSDL/'. Annotations include a thick black box around the 'input1' body element, a yellow highlight on the 'Cos' operation start tag, and another black box around the 'input2' body element.

```
<operation name="Sin">
  <input name="input1" message="tns:SinRequest"/>
  <output name="output1" message="tns:SinReply"/>
</operation>
<operation name="Cos">
  <input name="input2" message="tns:CosRequest"/>
  <output name="output2" message="tns:CosResponse"/>
</operation>
</portType>
<binding name="STTPScientificWSDLBinding" type="tns:STTPScientificWSDLPortType">
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="Sin">
    <soap:operation/>
    <input name="input1">
      <soap:body use="literal" namespace="http://j2ee.netbeans.org/wsdl/STTPScientificWSDL"/>
    </input>
    <output name="output1">
      <soap:body use="literal" namespace="http://j2ee.netbeans.org/wsdl/STTPScientificWSDL"/>
    </output>
  </operation>
  <operation name="Cos">
    <soap:operation/>
    <input name="input2">
      <soap:body/>
    </input>
    <output name="output2">
      <soap:body/>
    </output>
  </operation>
</binding>
</service>
```

# Creating Web Service using Contract First Approach

Go in source view and add the following details:

`<soap:body use="literal" namespace=http://j2ee.netbeans.org/wsd/STTPScientificWSDL/>`

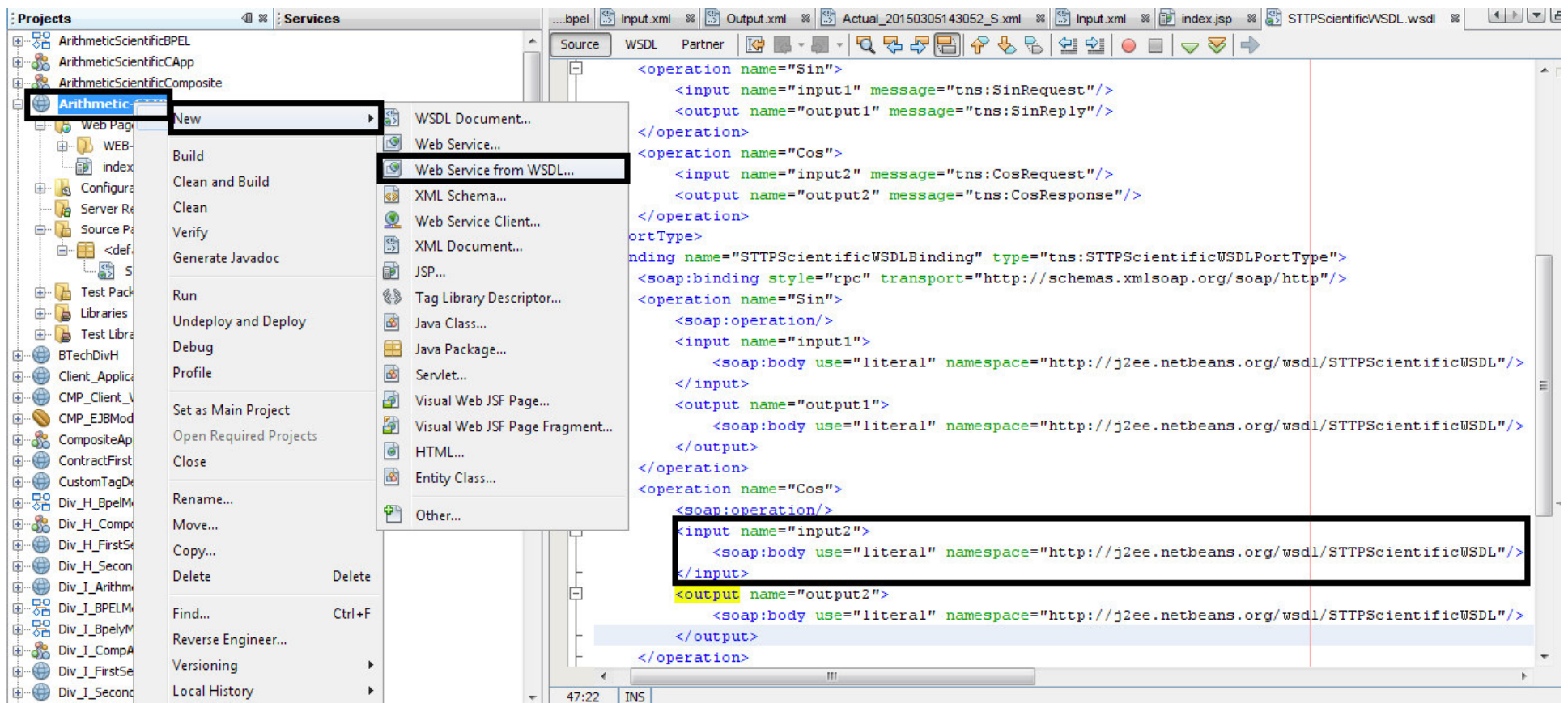


The screenshot shows an IDE window with the 'Source' tab selected, displaying WSDL code. The code defines two operations: 'Sin' and 'Cos'. The 'Sin' operation has an input 'input1' and an output 'output1', both with 'literal' body usage. The 'Cos' operation has an input 'input2' and an output 'output2', also with 'literal' body usage. The namespace for the bodies is 'http://j2ee.netbeans.org/wsd/STTPScientificWSDL/'. Annotations include a thick black box around the first 'literal' namespace declaration, a yellow highlight on the 'Cos' operation name, and a black box around the 'input2' body declaration.

```
<operation name="Sin">
  <input name="input1" message="tns:SinRequest"/>
  <output name="output1" message="tns:SinReply"/>
</operation>
<operation name="Cos">
  <input name="input2" message="tns:CosRequest"/>
  <output name="output2" message="tns:CosResponse"/>
</operation>
</portType>
<binding name="STTPScientificWSDLBinding" type="tns:STTPScientificWSDLPortType">
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="Sin">
    <soap:operation/>
    <input name="input1">
      <soap:body use="literal" namespace="http://j2ee.netbeans.org/wsd/STTPScientificWSDL"/>
    </input>
    <output name="output1">
      <soap:body use="literal" namespace="http://j2ee.netbeans.org/wsd/STTPScientificWSDL"/>
    </output>
  </operation>
  <operation name="Cos">
    <soap:operation/>
    <input name="input2">
      <soap:body/>
    </input>
    <output name="output2">
      <soap:body/>
    </output>
  </operation>
</binding>
</service>
```

# Creating Web Service using Contract First Approach

- Right click project folder and select New->Web Service from WSDL



# Creating Web Service using Contract First Approach

- A Dialogbox will appear asking for “configuration”

## WebService name

STTPScientific

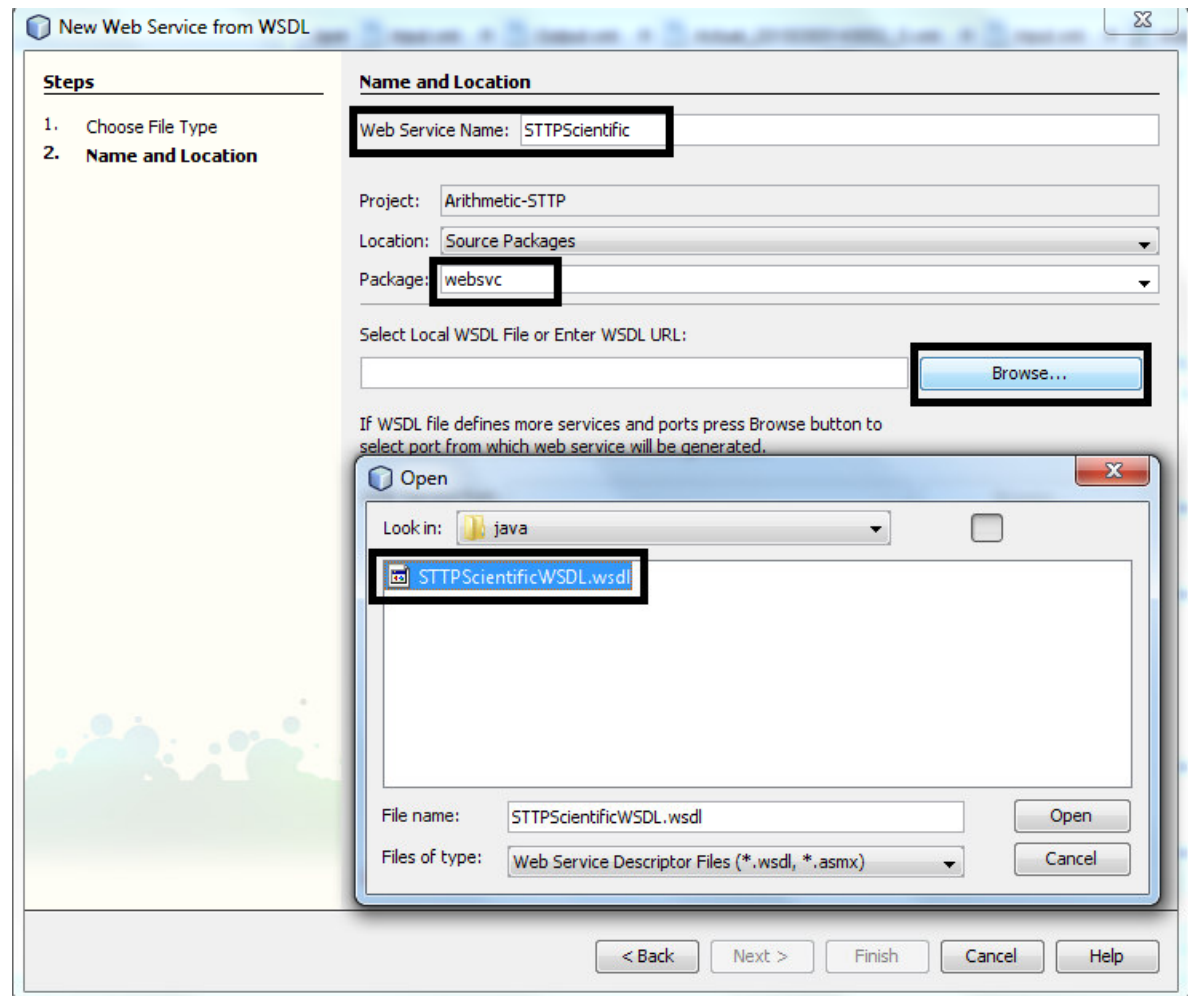
## Package Name

websvc

## Select WSDL

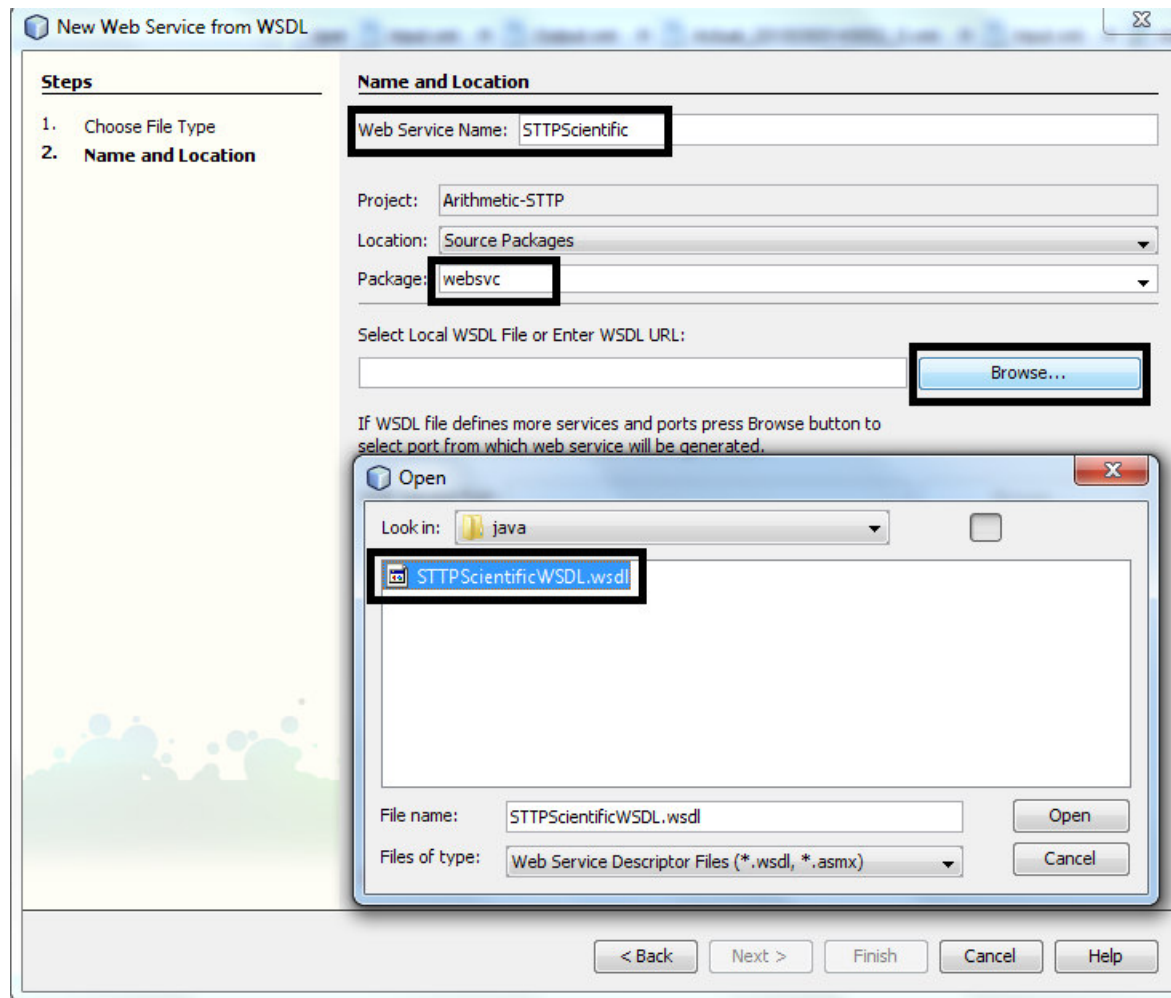
Browse and select WSDL File

WSDL file is stored at:  
Project Folder/src/java  
Click Finish



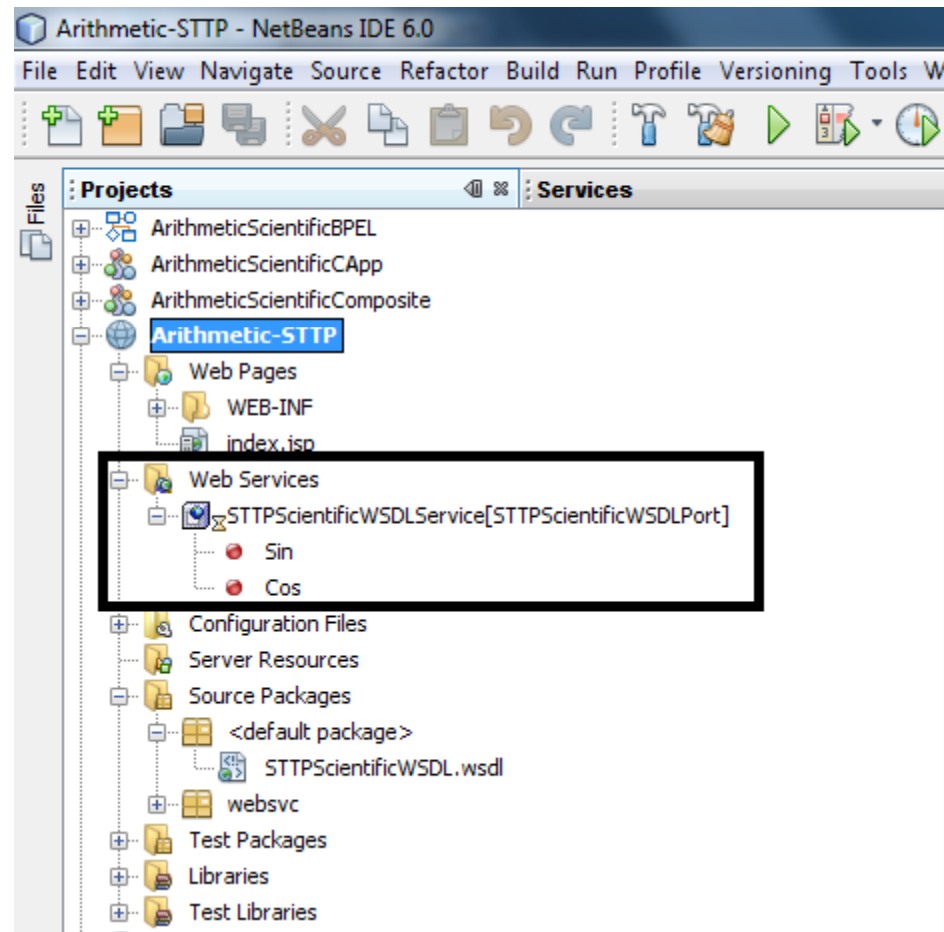
# Creating Web Service using Contract First Approach

- Select WSDL File



# Creating Web Service using Contract First Approach

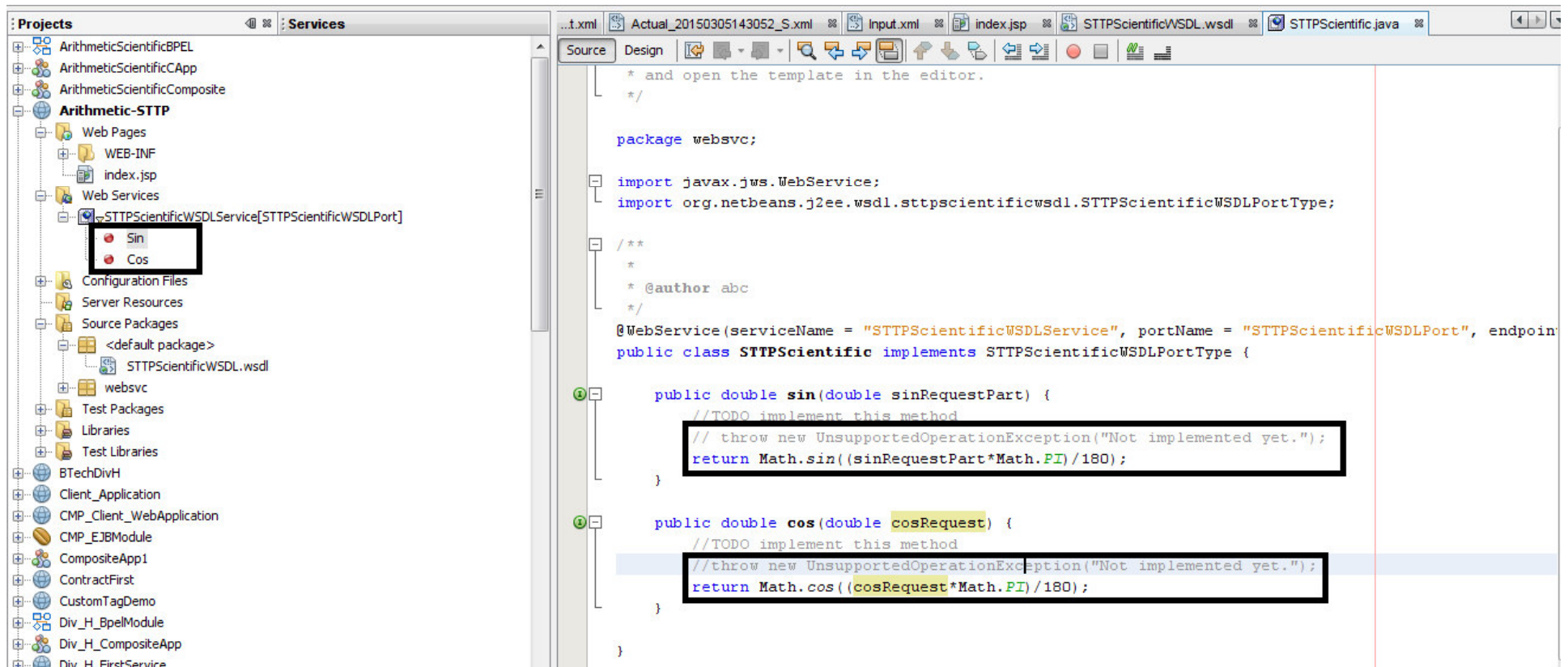
- Open Scientific.java in Source View





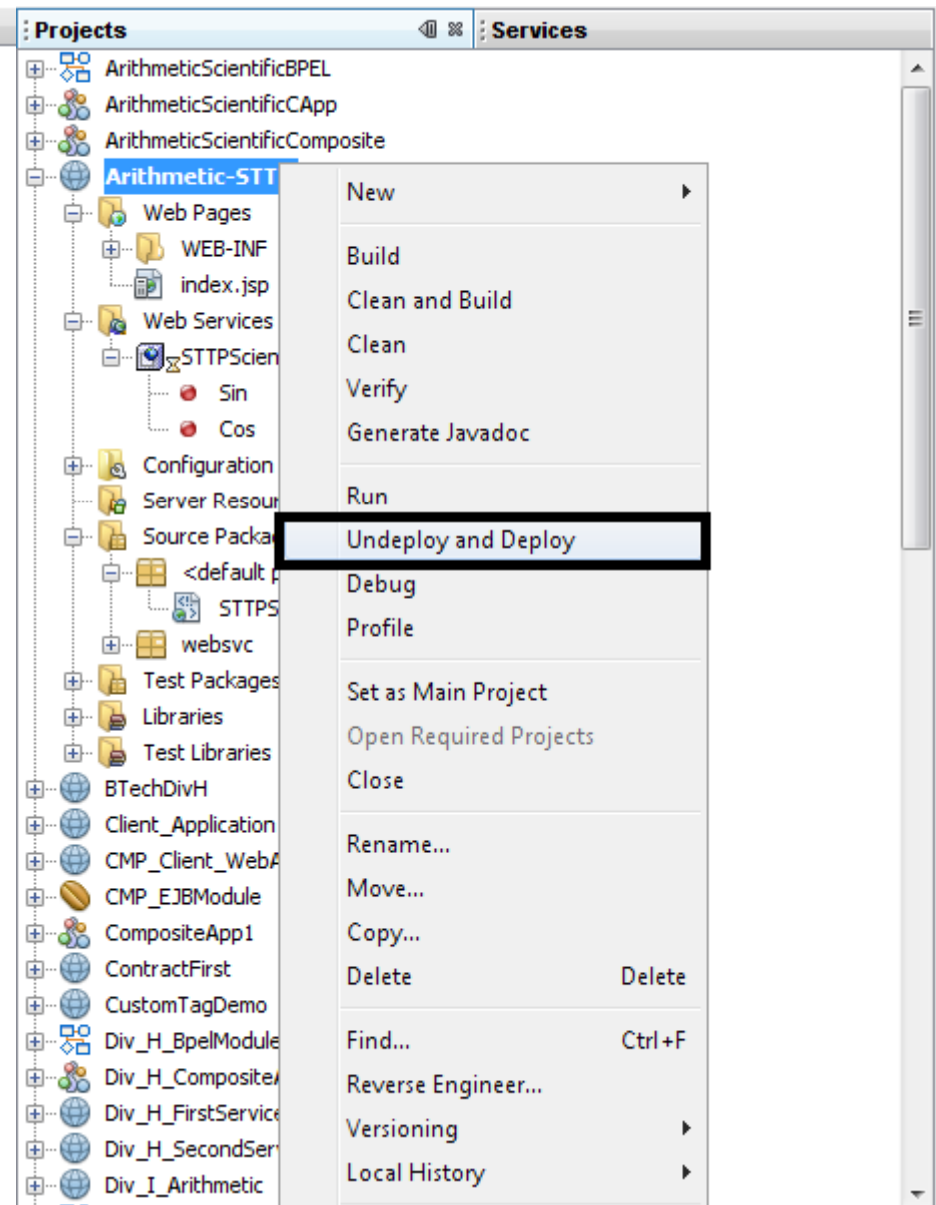
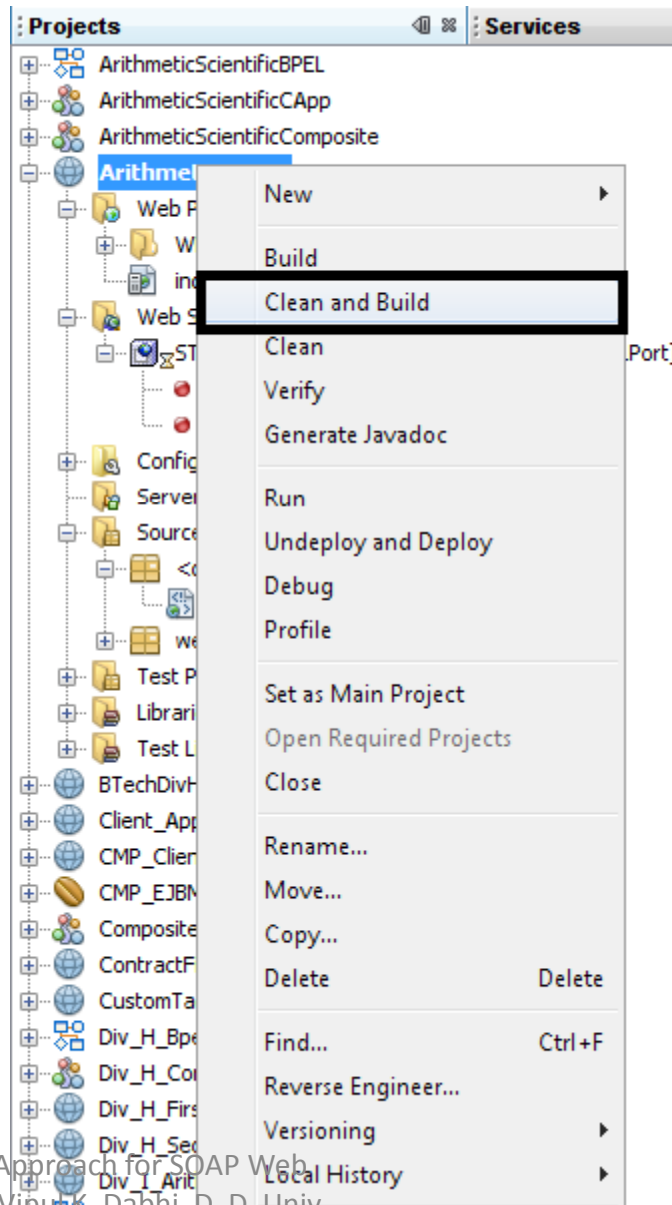
# Creating Web Service using Contract First Approach

- Open Scientific.java in Source View
- Add the code in it



# Creating Web Service using Contract First Approach

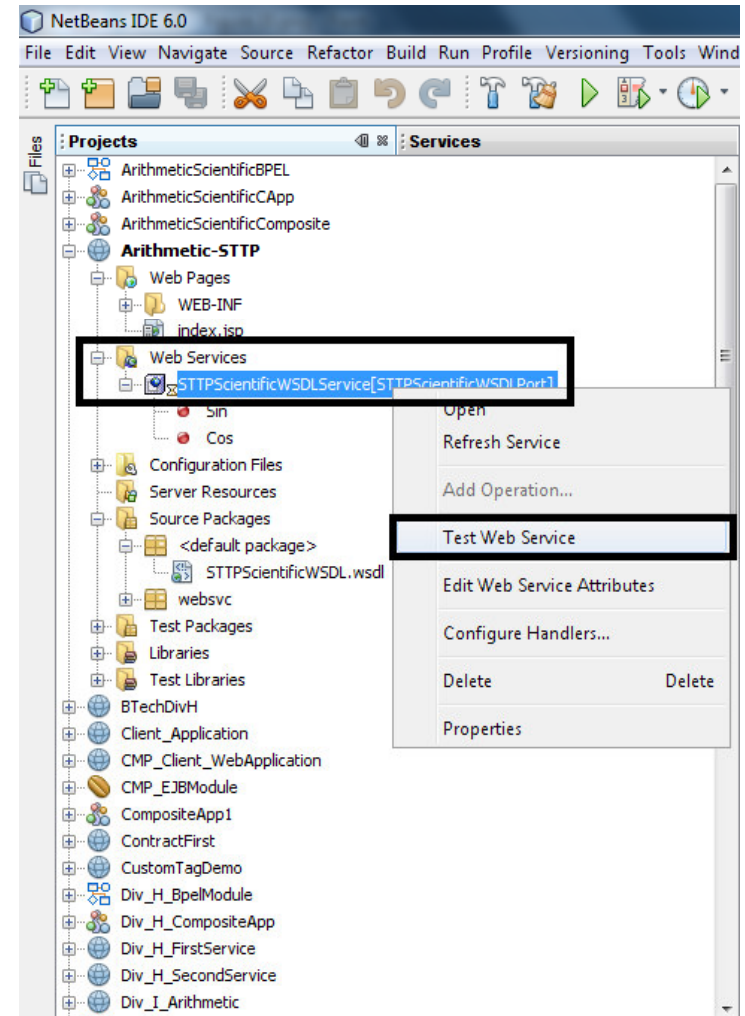
- Right click project and select Clean and Build
- Right click Project and select Deploy





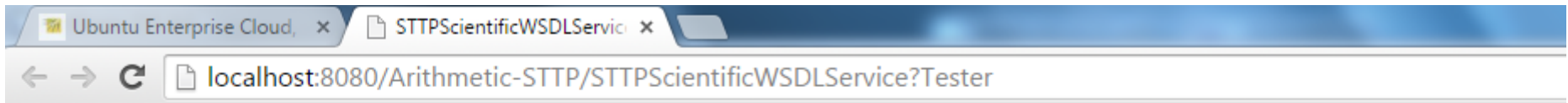
# Creating Web Service using Contract First Approach

- Right Click WebServices->STTPScientificWSDLService->Test Web Service



# Creating Web Service using Contract First Approach

- Right Click WebServices->STTPScientificWSDLService->Test Web Service



## STTPScientificWSDLService Web Service Tester

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

### Methods :

public abstract double org.netbeans.j2ee.wsdl.sttpscientificwsdl.STTPScientificWSDLPortType.sin(double)

sin (  )

public abstract double org.netbeans.j2ee.wsdl.sttpscientificwsdl.STTPScientificWSDLPortType.cos(double)

cos (  )

Ubuntu Enterprise Cloud, x Method invocation trace x

localhost:8080/Arithmetic-STTP/STTPScientificWSDLService?Tester

## sin Method invocation

---

### Method parameter(s)

| Type   | Value |
|--------|-------|
| double | 30    |

---

### Method returned

double : "0.49999999999999994"

---

### SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
  <S:Header/>
  <S:Body>
    <ns2:Sin xmlns:ns2="http://j2ee.netbeans.org/wsdl/STTPScientificWSDL">
      <SinRequestPart>30.0</SinRequestPart>
    </ns2:Sin>
  </S:Body>
</S:Envelope>
```

---

### SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?>
```

euca2-admin-x509.zip