



z/TPF V1.1

## TPF Users Group - 2011

# Experiences building WebSphere Applications that connect to zTPF using Web Services

Name: Colette A. Manoni  
Venue: Education

AIM Enterprise Platform Software  
IBM z/Transaction Processing Facility Enterprise Edition 1.1.0

Any reference to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

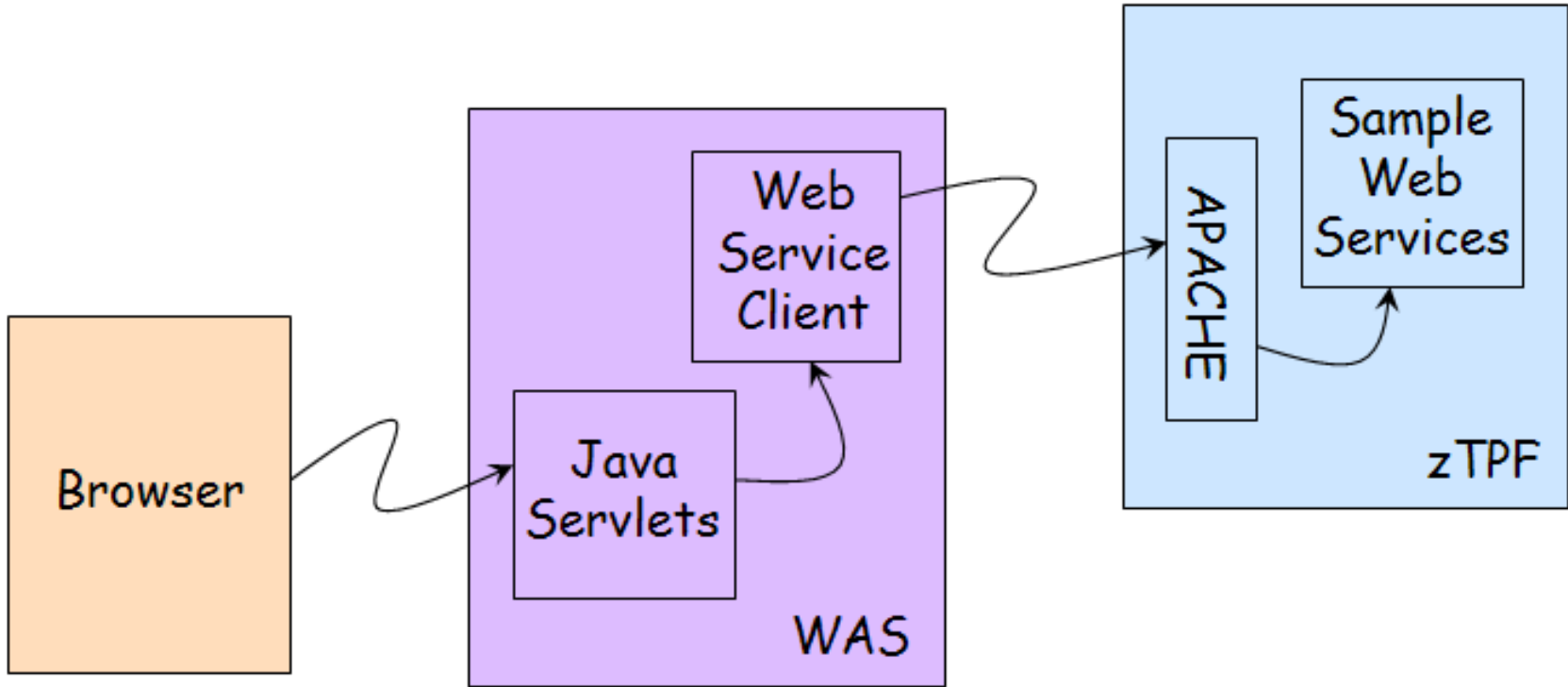
© 2011 IBM Corporation

# Agenda

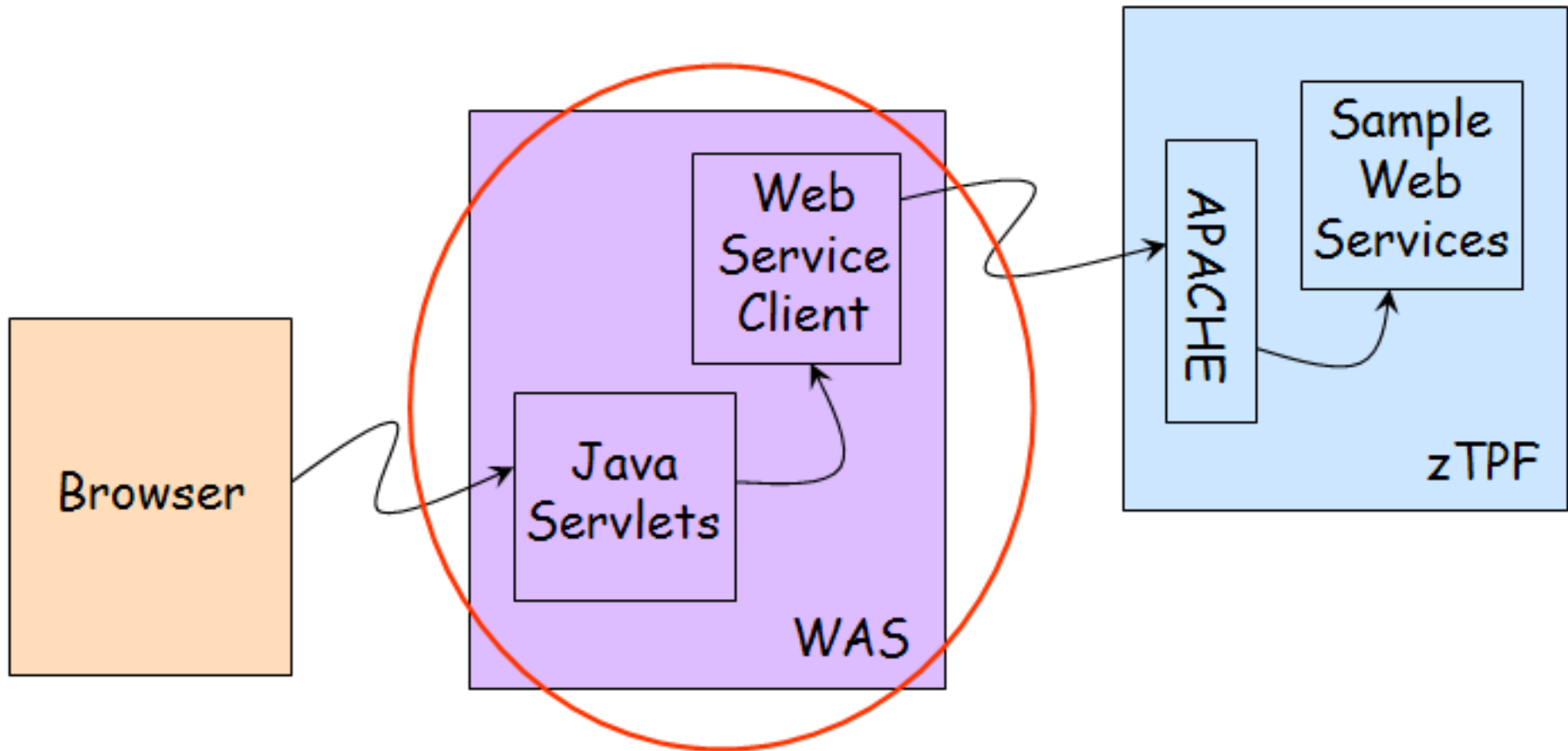
- **Demo using Rational Software Architect (RSA)**
- **Lessons learned when building client application**
- **Best practices when designing messages**
- **References for more information**

# Demo “Application”

...



# Demo “Application”

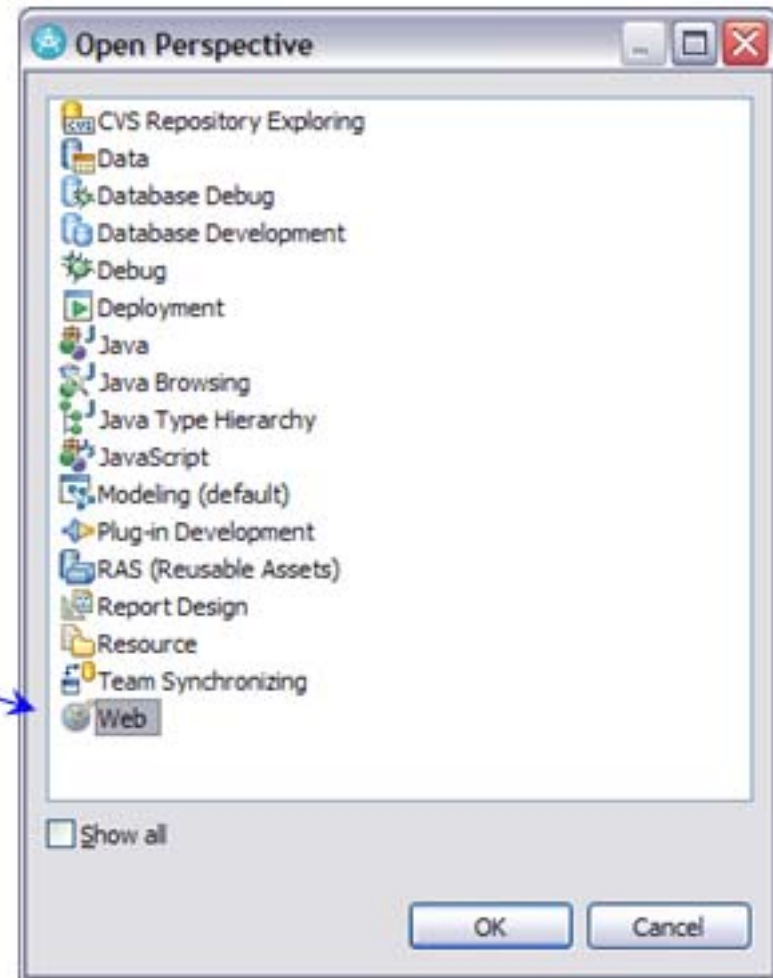


< Run the sample application >

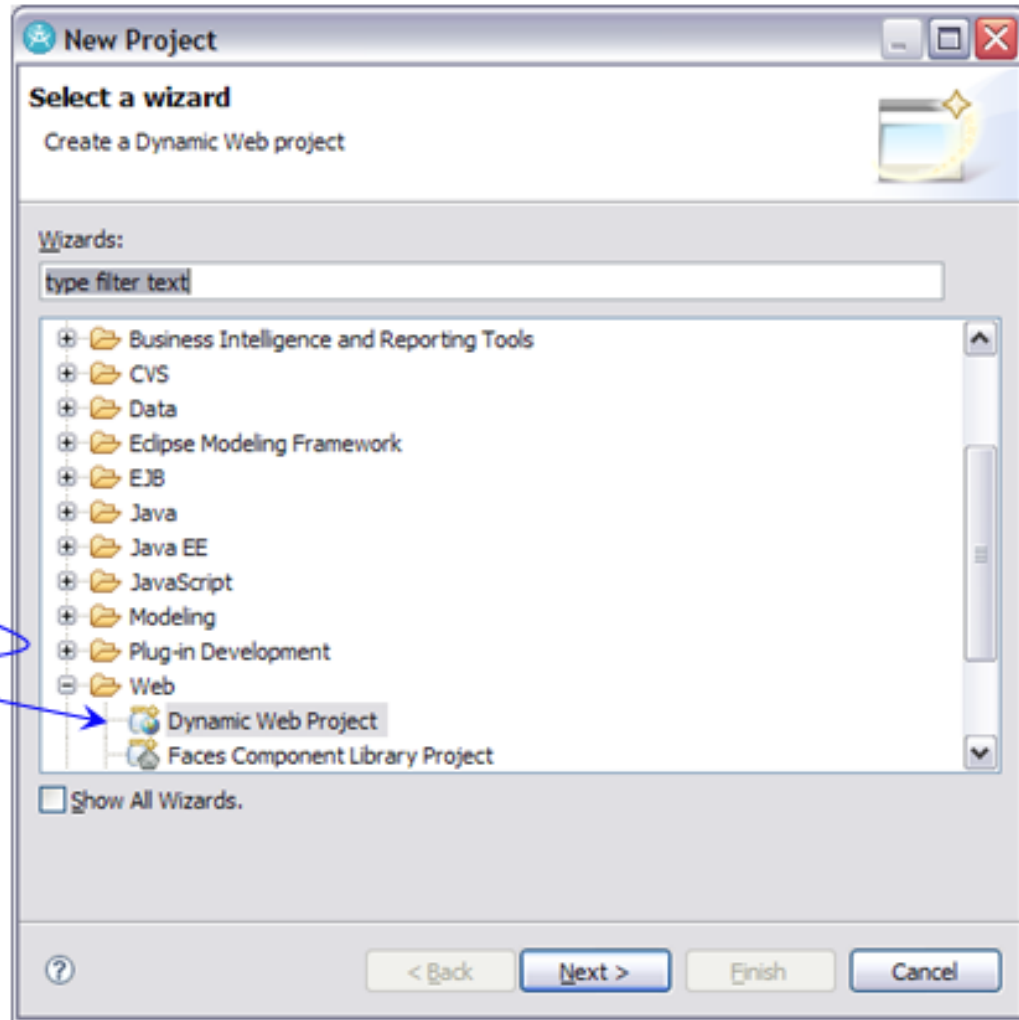
<Some RSA setup>

# Project setup in Rational Software Architect (RSA) for WebSphere

First set web perspective ...



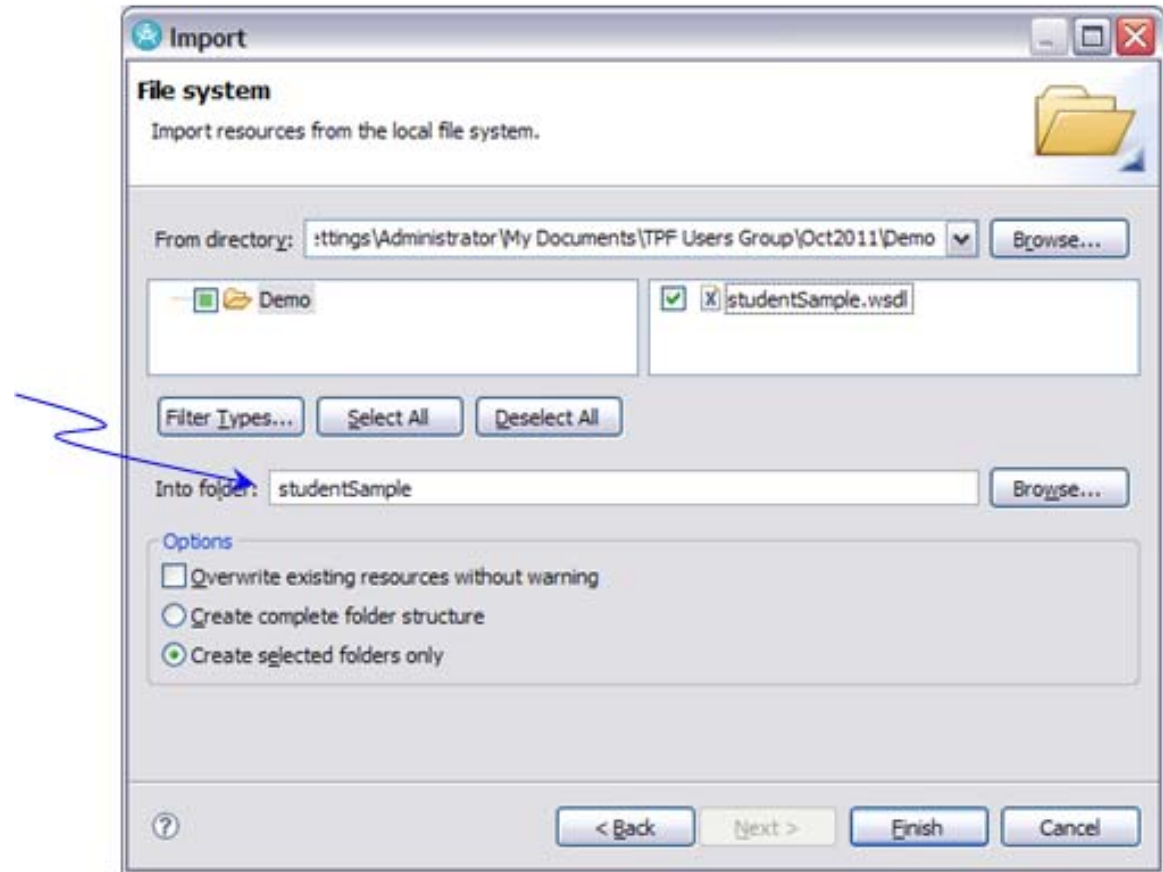
Create a web project



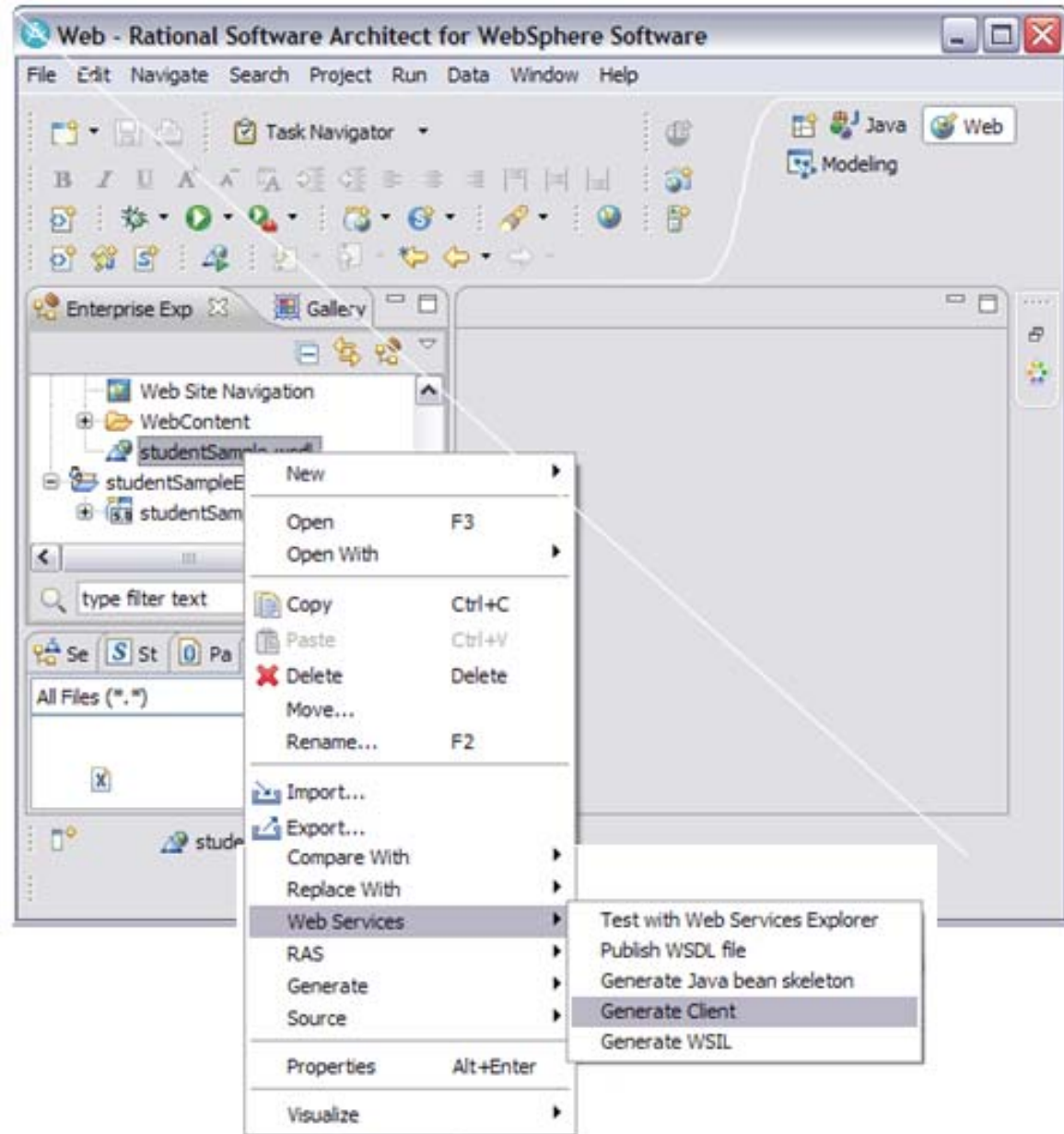


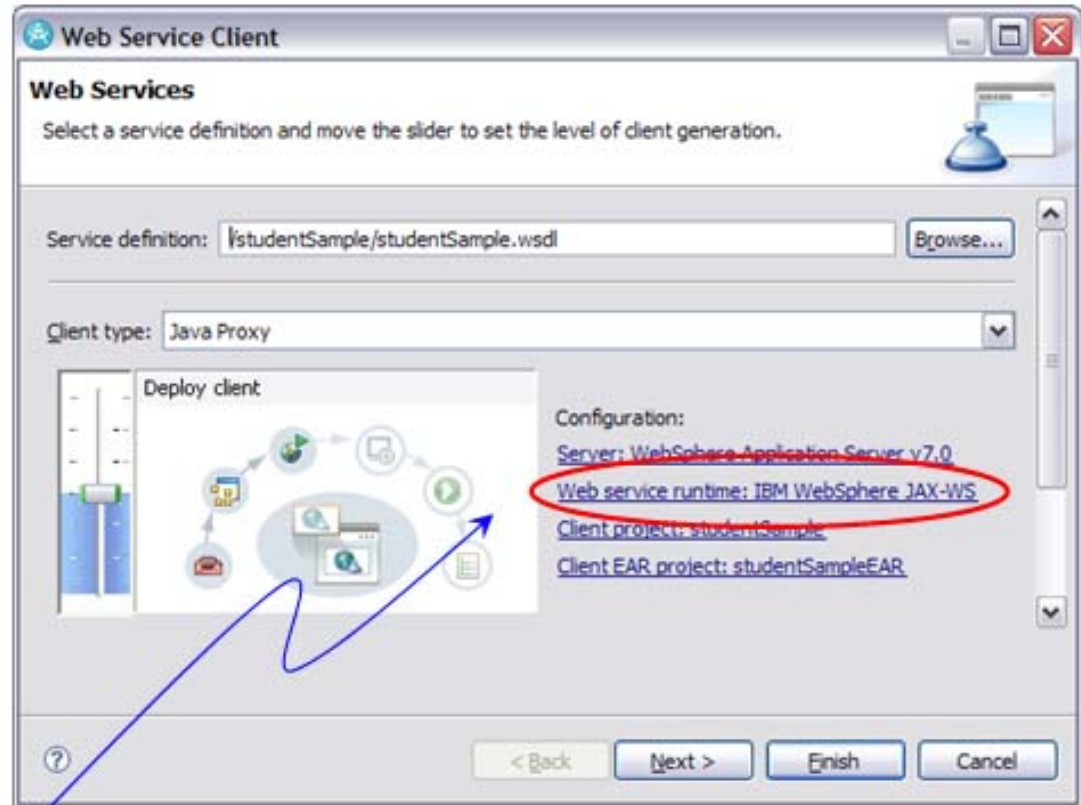
Import WSDL files  
for services being  
called in the  
application

Name of web project created



# Generate client stubs





Click on link to set the correct web service runtime:

JAX-WS or JAX-RPC

The screenshot displays the IBM Enterprise Explorer interface. The main window shows a tree view of 'Java Resources' under the package 'com.ibm.www'. The tree is expanded to show a list of generated Java classes. The classes are organized into several categories:

- com.ibm.www**: A list of 24 classes, including:
  - AddRequestType\_Deser.java
  - AddRequestType\_Helper.java
  - AddRequestType\_Ser.java
  - AddRequestType.java
  - AddResponseType\_Deser.java
  - AddResponseType\_Helper.java
  - AddResponseType\_Ser.java
  - AddResponseType.java
  - RemoveRequestType\_Deser.java
  - RemoveRequestType\_Helper.java
  - RemoveRequestType\_Ser.java
  - RemoveRequestType.java
  - RemoveResponseType\_Deser.java
  - RemoveResponseType\_Helper.java
  - RemoveResponseType\_Ser.java
  - RemoveResponseType.java
  - StudentSample\_PortType.java
  - StudentSample\_PortTypeProxy.java
  - StudentSample\_Service.java
  - StudentSample\_ServiceInformation.java
  - StudentSample\_ServiceLocator.java
  - StudentSampleSOAPStub.java
  - StudentType\_Deser.java
  - StudentType\_Helper.java
  - StudentType\_Ser.java
  - StudentType.java
- com.ibm.www.holders**: A package containing one class: AbstractServiceBean.java
- pagecode**: A package containing one class: StudentSample\_PortType\_AddStudent.java
- services**: A package containing one class: StudentSample\_PortType\_AddStudent.java

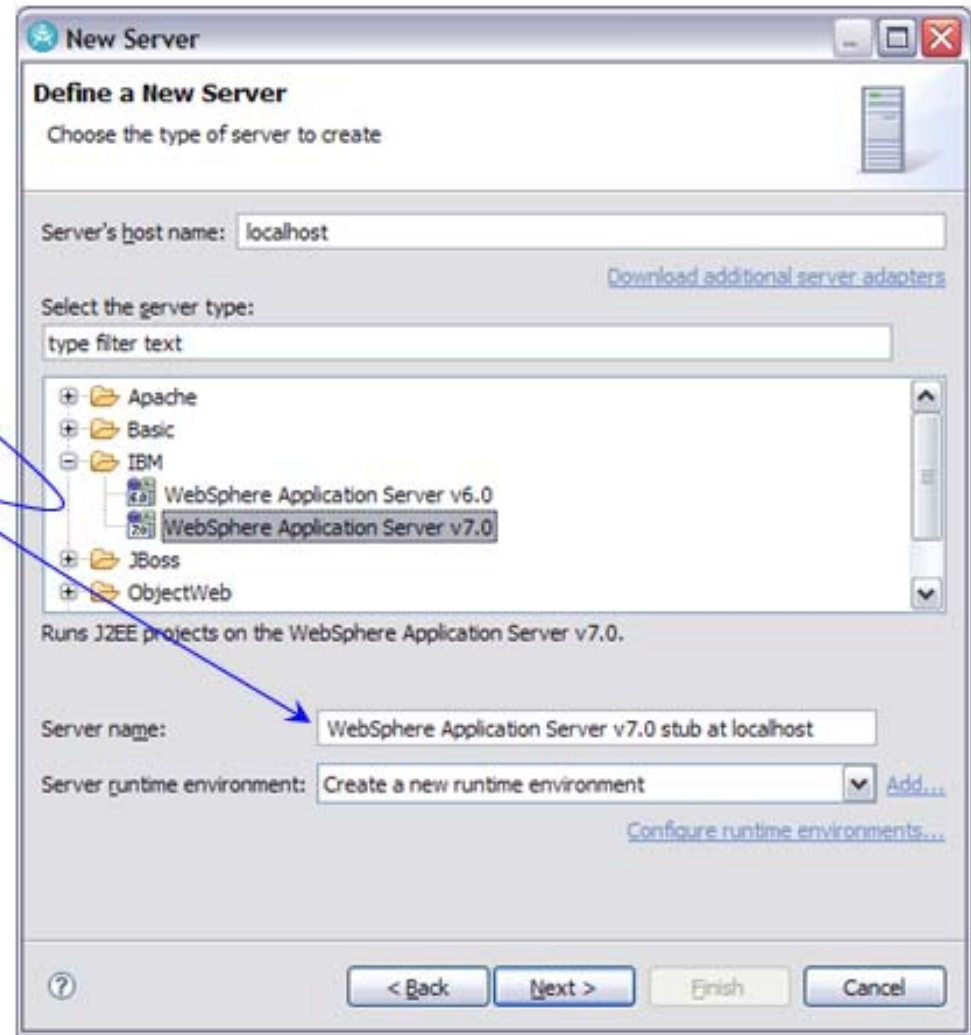
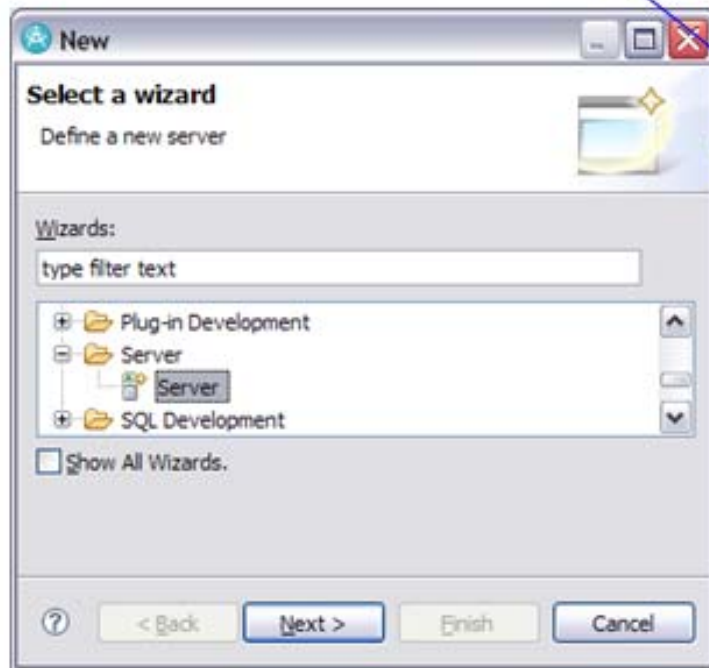
The search bar at the bottom of the tree view contains the text 'type filter text'.

List of  
generated  
classes for all  
methods in  
WSDL



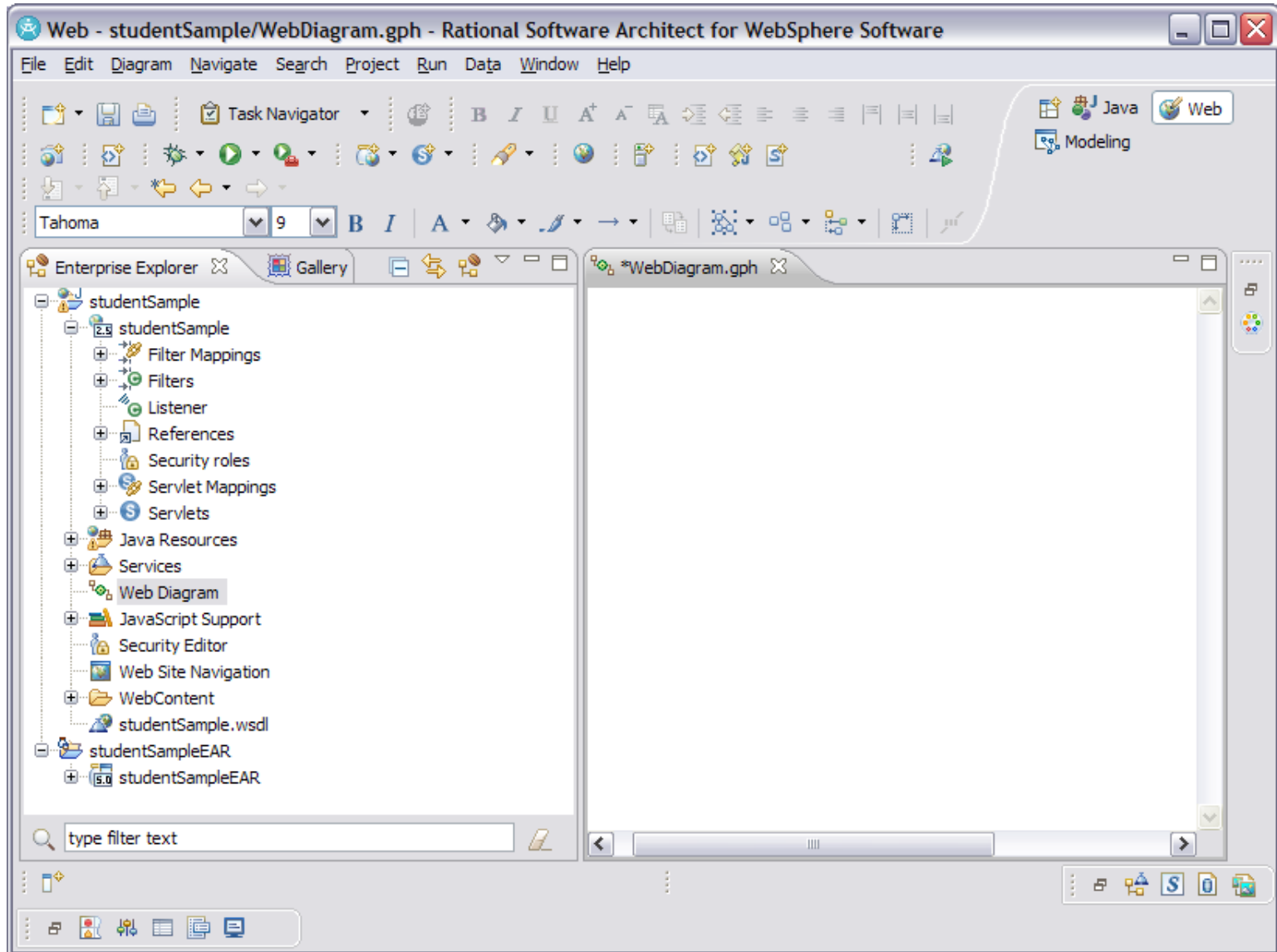
# Create a server to run the code

Can create a local server or define connection to a shared server.

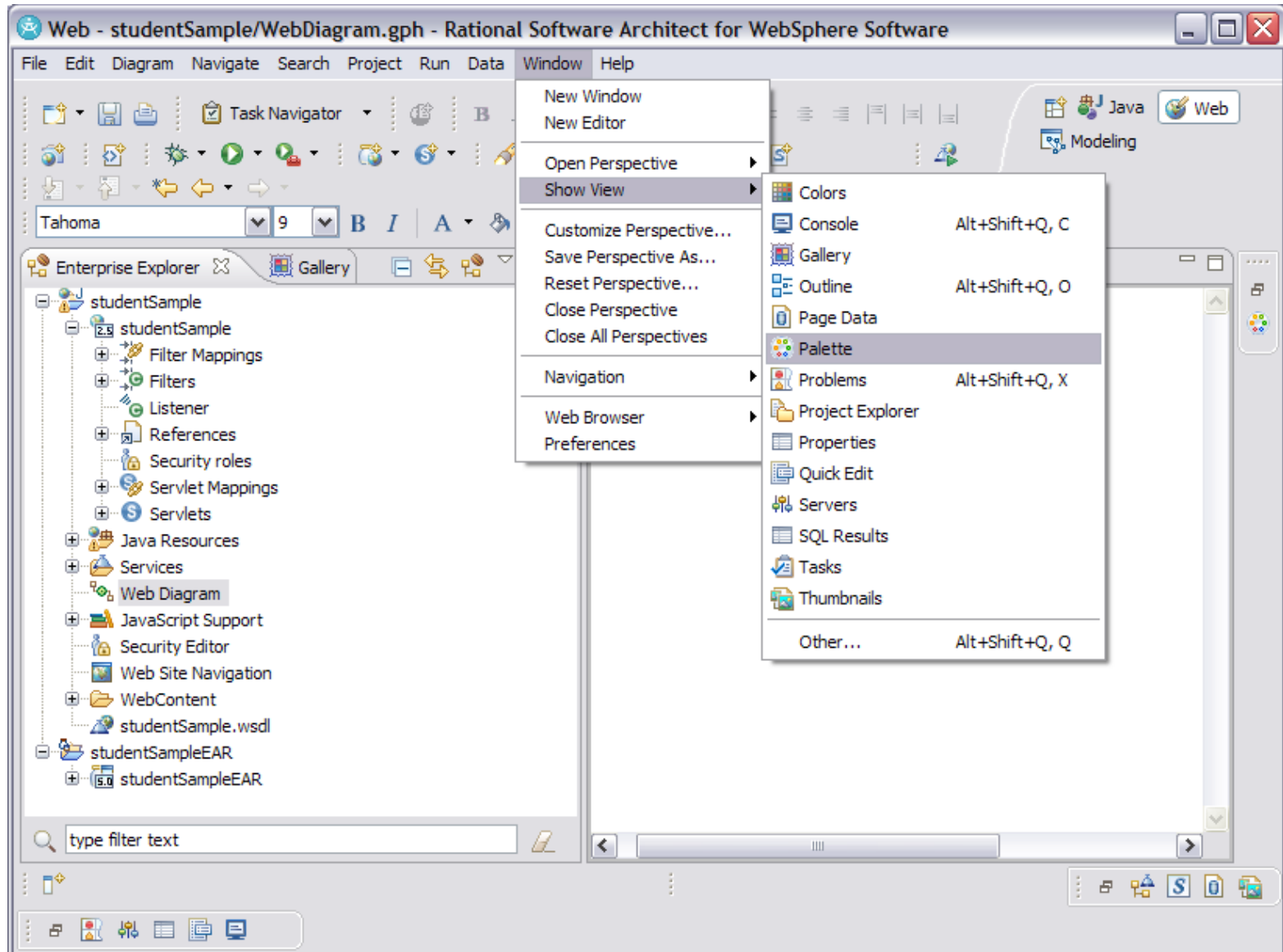


<Update sample application with remove service call>

# Open Web Diagram Editor

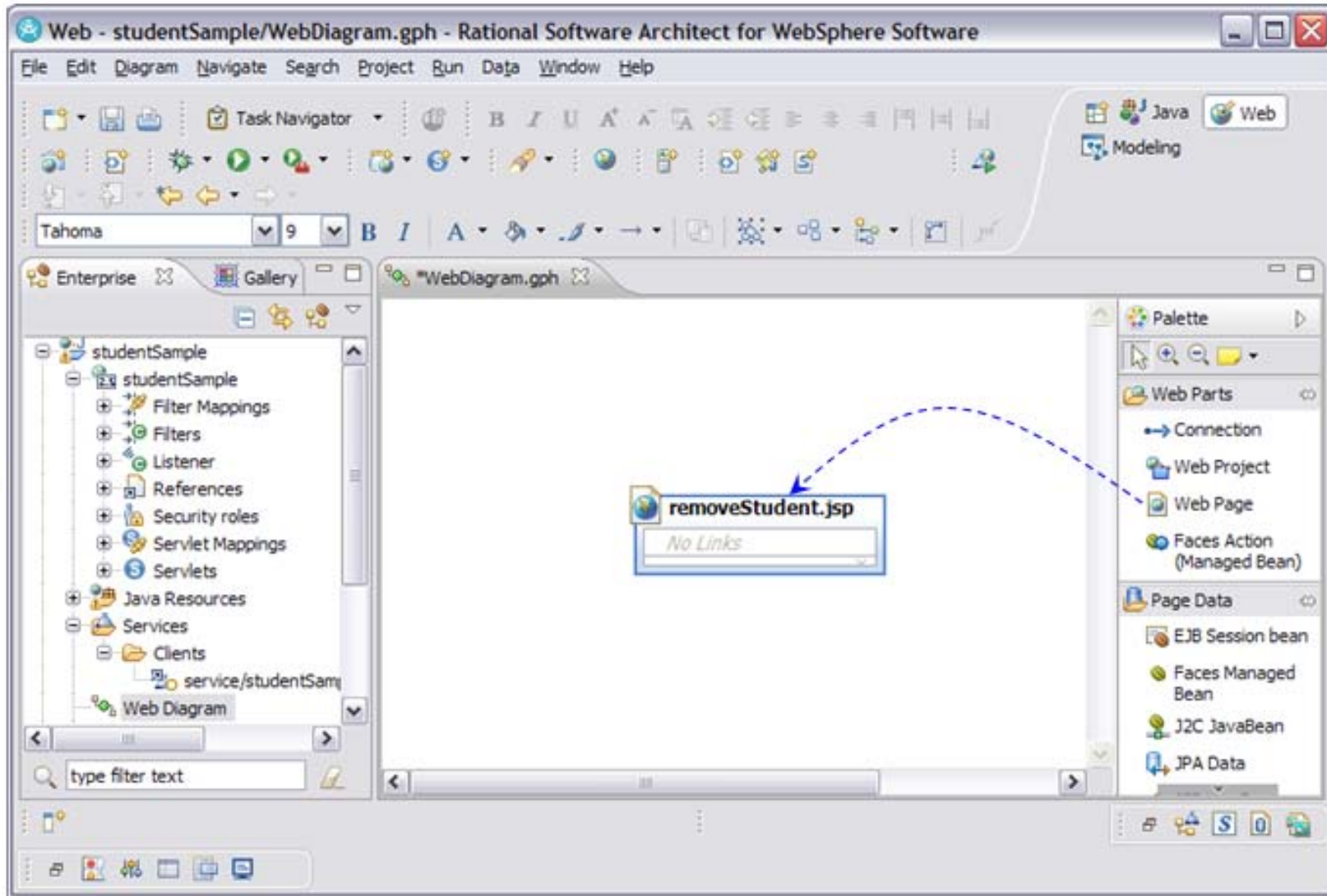


# Open the Palette of web tools

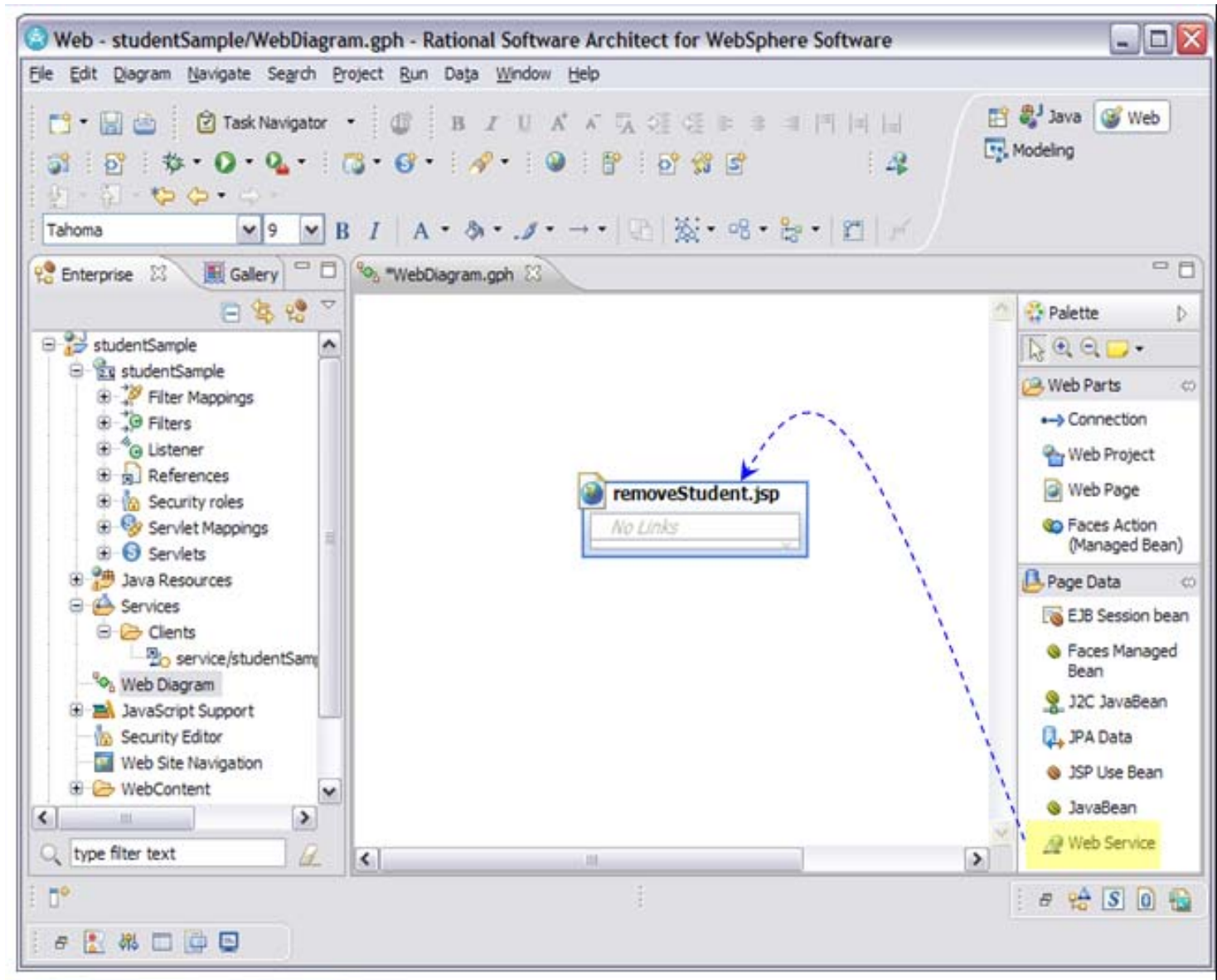


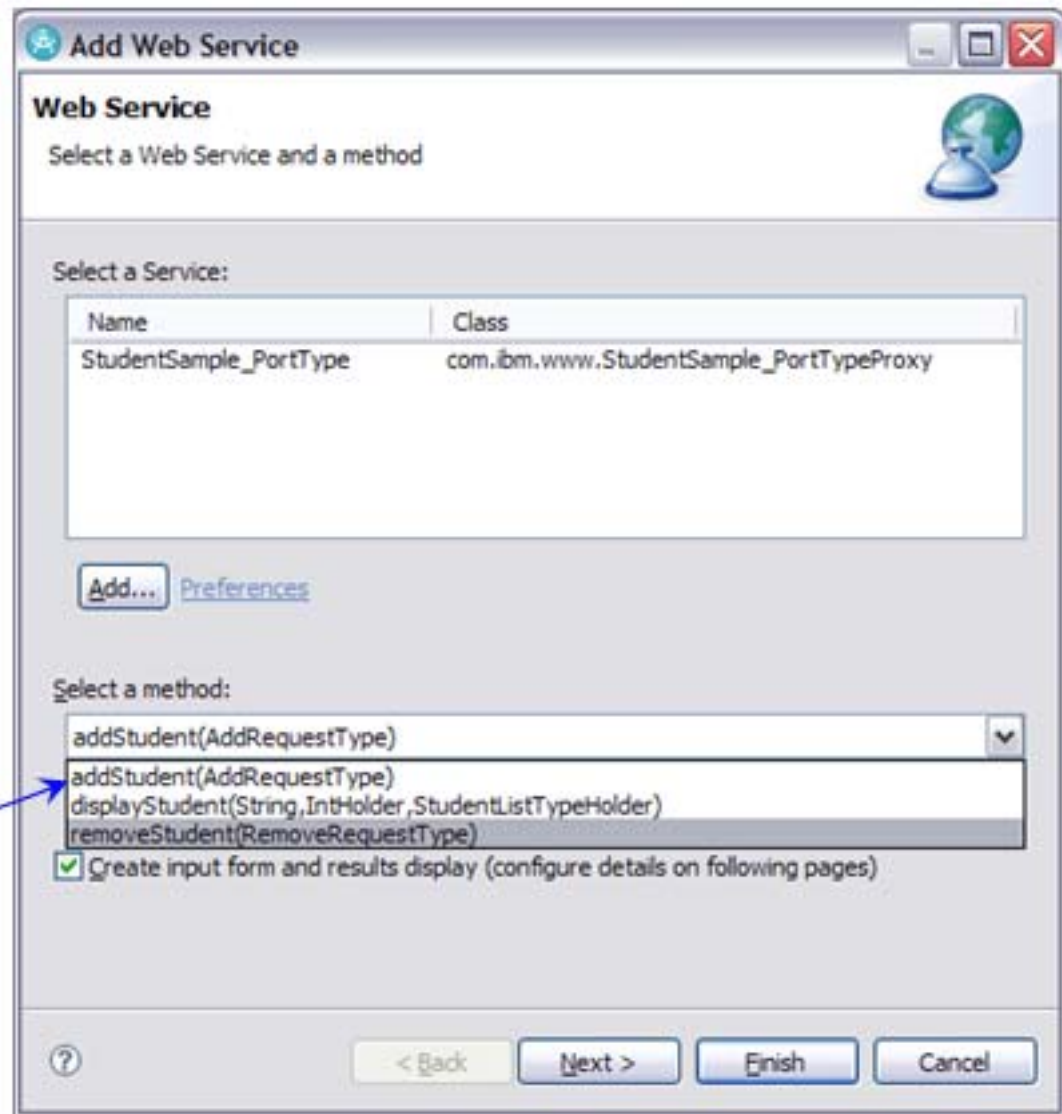


Drag and drop a new web page into the web diagram area



Drag  
and  
drop a  
web  
service  
onto  
the new  
page





3 methods in this  
service - select  
removeStudent

## Select fields to be displayed on the new page

**Add Web Service**

**Input Form**  
Configure the data controls for the input form

Fields to display:

Field Name	Label	Control Type
<input checked="" type="checkbox"/> removeRequest.student.fir:	FirstName:	Input Text
<input checked="" type="checkbox"/> removeRequest.student.las	LastName:	Input Text
<input checked="" type="checkbox"/> removeRequest.student.co	CompanyName:	Input Text

[All](#) [None](#) [Options...](#) [Configure Control Types](#)

[? < Back](#) [Next >](#) [Finish](#) [Cancel](#)

**Add Web Service**

**Results Form**  
Configure how the results will be displayed

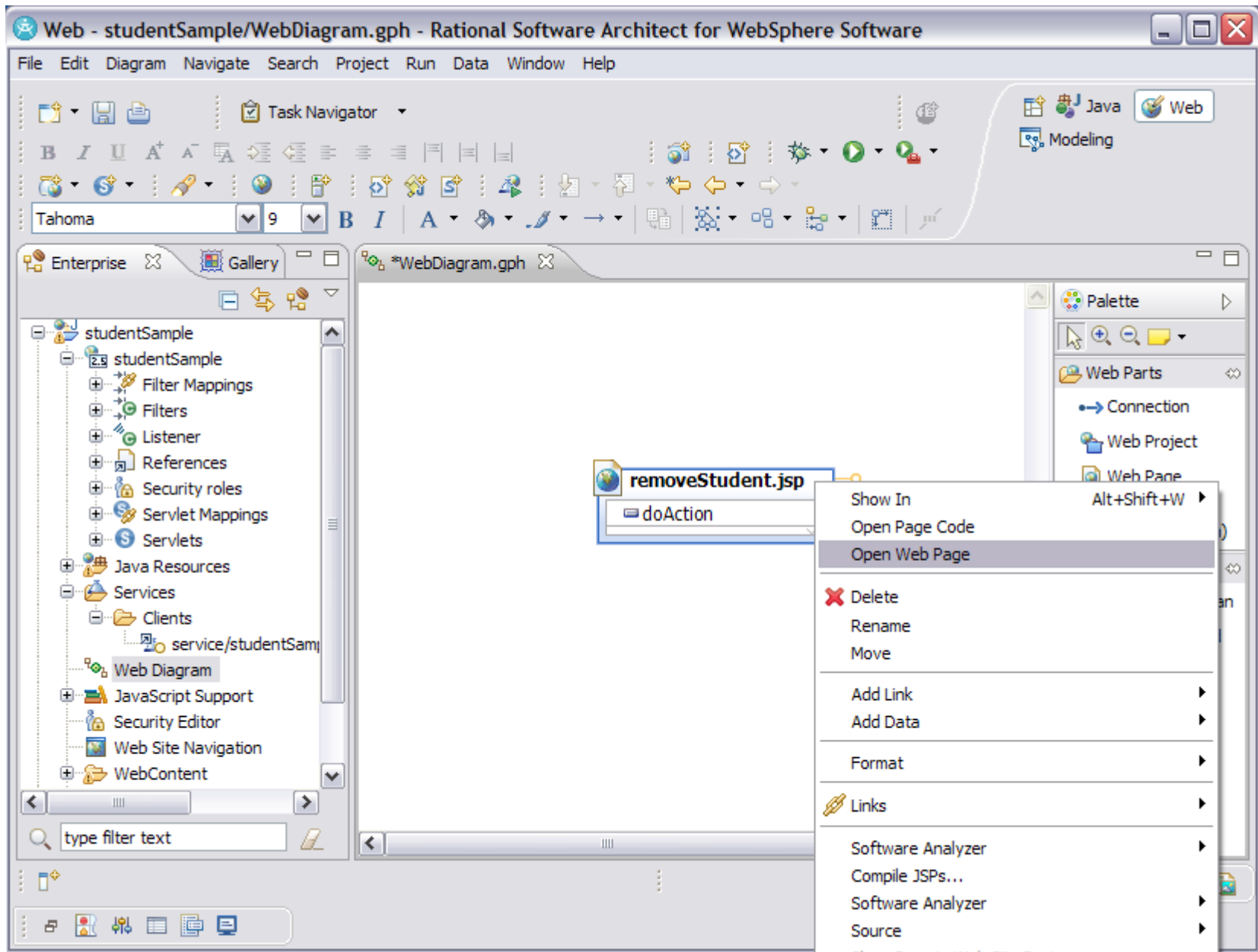
Fields to display:

Field Name	Label	Control Type
<input checked="" type="checkbox"/> operationCode (int)	OperationCode:	Display Text
<input checked="" type="checkbox"/> studentList (com.ibm.www..?	StudentList:	Data Table w

[All](#) [None](#) [Options...](#) [Configure Control Types](#)

[? < Back](#) [Next >](#) [Finish](#) [Cancel](#)





Task Navigator

None

Java Web Modeling

Enterprise Gallery

- studentSample
  - studentSample
    - Filter Mappings
    - Filters
    - Listener
    - References
    - Security roles
    - Servlet Mappings
    - Servlets
  - Java Resources
  - Services
    - Clients
      - service/studentSample
    - Web Diagram
  - JavaScript Support
  - Security Editor
  - Web Site Navigation
  - WebContent
  - studentSample.wsdl

type filter text

\*WebDiagram.gph removeStudent.jsp

removeStudent.jsp - removeStudent

body Standard

removeStudent.jsp - removeStudent

FirstName: {firstName} abc  
LastName: {lastName} abc  
CompanyName: {companyName} abc

{Error Messages} ↓

Submit

OperationCode: {operationCode} abc

StudentList:

FirstName abc	LastName abc	CompanyName abc
{firstName} abc	{lastName} abc	{companyName} abc

Design Source Split Preview

# Need to make a couple of updates to generated code...

...src/services/StudentSample\_PortType\_RemoveStudent.java

- Update request to create complex object

```
public AddRequestType getAddRequest() {
    if (removeRequest == null) {
        removeRequest = new RemoveRequestType();
        //TODO This object has properties which may not have been initialized. Add
        initialization code here if needed.
        removeRequest.setStudent(new StudentType());
    }
    return addRequest;
}
```

- Add service endpoint information

```
public String doAction() {
    getParamBean();
    getService().setEndpoint("http://9.57.13.195/studentSample");
    if (paramBean != null) {
        try {
            resultBean = getService().removeStudent(
                paramBean.getRemoveRequest());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    return "";
}
```

Update managed bean to be session scope - allows for data and control to be passed from one page to another

The screenshot shows the Rational Software Architect interface for editing a ManagedBean in a faces-config.xml file. The main window is titled "Web - studentSample/WebContent/WEB-INF/faces-config.xml - Rational Software Architect for WebSphere Software".

The left sidebar shows a project tree with folders like WebContent, META-INF, theme, WEB-INF, lib, wsd, and files like faces-config.xml, ibm-web-bnd.xml, and ibm-web-ext.xml. The "faces-config.xml" file is selected.

The main workspace is titled "ManagedBean" and contains the following sections:

- Managed Bean Elements:** A tree view showing the configuration structure. Under "session", there are three "studentSample\_Por" entries. Under "request", there are "pc\_AddStudent", "pc\_RemoveStudent", and "pc\_DisplayStudentL". Under "application", there is "none".
- Managed Bean:** This section describes the general configuration of the managed bean.
  - Managed Bean name\*: studentSample\_PortType\_RemoveStudent
  - Managed Bean class\*: services.StudentSample\_PortType (with a "Browse..." button)
  - Managed Bean scope\*: session (selected from a dropdown menu)
- Initialization:** This section allows for initializing the managed bean's properties.
  - Managed Bean class type:  General class  Map  List
  - A table with columns "Name", "Class", and "Value" is present, with "Add...", "Edit...", and "Remove" buttons to its right.

At the bottom of the workspace, there are tabs for "Overview", "Navigation Rule", "ManagedBean", "Component", "Others", and "Source". The "ManagedBean" tab is active.

The status bar at the bottom right shows "Save Domain and Na...t references" and several utility icons.



<Run application to show removeStudent>

<Create action button on remove page to call display page>

# From design view of displayStudentList.jsp - drag and drop button



The screenshot displays the Rational Software Architect for WebSphere Software interface. The main window shows the design view of the `displayStudentList.jsp` file. The design canvas contains a form with the following elements:

- Input fields for `firstName`, `LastName`, and `Company Name`.
- An `Submit` button.
- A `StudentList` component containing three columns: `FirstName`, `LastName`, and `Company Name`.

A blue dashed arrow indicates the drag-and-drop action of the `Submit` button from the Properties panel to the design canvas. The Properties panel shows the configuration for the `hx:commandExButton` component:

- Component: `hx:commandExButton`
- ID: `button1`
- Action or outcome: (empty)
- Style: Props: (empty)
- Classes: `commandExButt...`

The Properties panel also includes sections for Display options, Parameter, Accessibility, and Styles. The right-hand side of the interface features a Palette with various UI components, including `Button - Command`, which is highlighted.

# Edit to update label and define an action rule

The screenshot displays the IBM WebSphere IDE interface. The main workspace shows a web page design for 'removeStudent.jsp' with a 'Submit' button. A dialog box titled 'Edit Navigation Rule' is open, allowing configuration of the button's action. The dialog includes the following sections:

- Go to the page:** Page: /displayStudentList.jsp
- When the action returns the outcome:**  The outcome named: Display
- This rule is used for:**  This page only
- This rule is used by:**  This action only: Display
- When following this rule:**  Use request forwarding (parameters work automatically)

The 'Properties' view at the bottom shows the selected button's configuration:

- id:** button1
- Action or outcome:** Display
- Type:**  Submit

Below the 'Action or outcome' field, a table lists rules on the page:

When Outcome Is	Go To Page	If Action Is	
Display	/displayStudent...	Display	<input type="checkbox"/> Add Rule...
<input type="checkbox"/> Rules on this page			

Buttons for 'Remove Rule' and 'Edit Rule...' are also visible.

<Run application to show display button from remove page>

< Using data from display page as input to remove page >



# Update displayStudentList.jsp to add remove button ...

The screenshot displays the Rational Software Architect interface for editing the `displayStudentList.jsp` file. The design view shows a form with the following elements:

- A text input field for `lastName` with a label `LastName`.
- An `{Error Messages}` section containing a `Submit` button.
- A table with three columns: `FirstName`, `LastName`, and `CompanyName`.
- Below the table, there are three text input fields with labels `{firstName}`, `{lastName}`, and `{companyName}`.
- A `Submit` button at the bottom left of the form area.

The right-hand palette shows various UI components, with `Button - Command` selected. A blue dashed arrow points from this component to the `Submit` button in the design view. The console at the bottom shows the component type `hx:commandExButton` and its properties, including the button label `Submit`.

# Add custom action for remove button ...

The screenshot shows the IBM WebSphere IDE interface. The main window displays a web page design for 'displayStudentList.jsp'. The design includes a form with input fields for 'FirstName' and 'LastName', a 'Submit' button, and a 'Remove' button. A 'Faces Action selection' dialog box is open, allowing the user to choose an action for the 'Remove' button. The dialog lists several actions, with 'doRemove (services.StudentSample\_PortType\_D)' selected. The IDE's left sidebar shows a project tree with 'studentSample' and various libraries. The bottom of the IDE shows a 'Servers' panel and a 'Properties' panel for the selected 'hxcommandExButton' component.



# Some Lessons Learned

- **Very easy to use for simple interfaces**
  - Do not need in-depth TPF skills
  - For more complicated types will need more Java and web services skills
- **Generated message guaranteed to be valid WSDL/XML**
  - Web service provider can focus on main-path vs. error path
- **Multiple generation options**
  - JAX-RPC vs. JAX-WS
  - Use one format consistently, otherwise different generated code
  - WSDL may dictate what runtime is needed
- **Tooling limitations**
  - Types limited to Java types - e.g. unsigned integers, xsd:any
  - XML options – e.g. choice not supported
  - Versioning – e.g. changing datatypes or removing elements
- **Complex type conflicts**
  - Using multiple WSDLs may see conflicts, if use same type name with same namespace

# Suggested Best Practices

- **Establish and use a namespace hierarchy**
  - Group related services within same namespace
  - Avoids conflicts
- **Keep messages well structured**
  - Easier to maintain and reuse definitions
- **Define and use set of common types**
  - Easier to flow output from one message as input to another
- **A good design for message interface eliminates issues/problems creating client application**

# References

- **Rational Software Architect trial / evaluation**
  - <http://www.ibm.com/developerworks/downloads/r/architect/>
  - Include the optional WebSphere Application Server component to be able to define a server within RSA environment
- **Open Travel Alliance (OTA) 2.0 Work Group**
  - [http://wiki.opentravel.org/index.php/Public:OpenTravel\\_Projects](http://wiki.opentravel.org/index.php/Public:OpenTravel_Projects)
  - Access to the project team page requires a login but introduction presentation is available to public

# Acknowledgements

Many Thanks to teammates ...

Dan Gritter

Junior Rincon

Kishore Nagareddy

Matt Gritter

Thank You !

# Trademarks

- **IBM® Rational® WebSphere®** are registered trademarks of International Business Machines Corporation in the United States, other countries, or both.
- **Java** and all **Java-based** trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
- **Microsoft, Windows, Windows NT,** and the **Windows logo** are trademarks of Microsoft Corporation in the United States, other countries, or both.
- **Notes**
- **Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.**
- **All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.**
- **This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.**
- **All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.**
- **Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.**
- **Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.**
- **This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.**