CASA Editor UI Specification

Author: Tientien Li

$Revision: 1.00 $
$Date: 01/04/2007 15:01:51 $

Note: An earlier 0.5 version of this document was published in CASA design use cases[1] which was developed and reviewed by CompApp I-Team members between 09/06 and 11/06. The five use cases described in this document are copied from the earlier document and reformatted according to the Netbeans UI Spec Template[2].

1. Use Cases and Scenarios

1.1 Add missing concrete WSDL elements

A bpel project can be developed using only abstract WSDL elements. However to deploy such a project, the user has to add concrete WSDL elements specifying ports and bindings. In this use case, the user uses the CASA editor to supplied concrete WSDL elements needed for the deployment in the composite application.

Scenario:

- Select the composite application project.
- Invoke project **Context Menu** item **Edit Project.** (see Figure 1)
- A **CASA Editor Window** opens showing the **Design View** of the composite application project configuration. (see Figure 3)
- In the **JBI Modules** area, there is a BPEL **Service Unit** with a **Provide endpoint**.
- Select the **SOAP** palette item from the **WSDL Binding** category on the **CASA Palette Panel**. (see Figure 4)
- Drag-n-drop it onto the **WSDL Ports** area of the **Design View** window.
- A **casaport1** WSDL port is created on the **WSDL Ports** area.
- Click on the **Consume endpoint** of **casaport1** and drag the mouse over to the **Provide endpoint** of the Bpel **Service Unit** to create a connection.
- Expand the **casaport1** node on the **CASA Navigator Panel**. (see Figure 5)
- Select WSDL port or binding nodes to edit their attributes in the **CASA Property Panel**. (see Figure 6)
- Click on the "x" mark on the tab of **CASA Editor Window** to close it.
- Select **Save** button to save the changes. (see Figure 7)
- Build and deploy the composite application.

1.2 Replace concrete WSDL elements

The CASA editor can be used to change the deployment configuration of a composite application. In this use case, a service provided by a bpel project configured to use an

HTTP SOAP binding port is modified to provide the same service using a FILE binding port instead. The change is made in the composite application project using CASA without touching the bpel project.

Scenario:

- Select the composite application project.
- Invoke project **Context Menu** item **Edit Project.**. (see Figure 1)
- A **CASA Editor Window** opens showing the **Design View** of the composite application project configuration. (see Figure 3)
- In the **JBI Modules** area, there is a BPEL **Service Unit** with a **Provide endpoint** connected to the **Consume endpoint** of **port1** on the **WSDL Ports** area.
- Click on the connection to select it.
- Click on the DELTE key to delete the selected connection.
- Select the **FILE** palette item from the **WSDL Binding** category on the **CASA Palette Panel**. (see Figure 4)
- Drag and drop it onto the **WSDL Ports** area of the **Design View** window.
- A **casaport1** WSDL port is created on the **WSDL Ports** area.
- Click on the **Consume endpoint** of **casaport1** and drag the mouse over to the **Provide endpoint** of the Bpel **Service Unit** to create a connection.
- Expand the **casaport1** node on the **CASA Navigator Panel**. (see Figure 5)
- Select WSDL port or binding nodes to edit their attributes in the **CASA Property Panel**. (see Figure 6)
- Click on the "x" mark on the tab of **CASA Editor Window** to close it.
- Select **Save** button to save the changes. (see Figure 7)
- Build and deploy the composite application.

1.3 Correct auto-generated deployment configuration

The deployment configuration auto-generated by the composite application project build process may not be correct. In this use case, the user uses the CASA editor to make corrections to the auto-generated configuration. The changes user made in CASA will be persisted in subsequent project builds.

Scenario:

- Select the composite application project.
- Invoke project **Context Menu** item **Edit Project.**. (see Figure 1)
- A **CASA Editor Window** opens showing the **Design View** of the composite application project configuration. (see Figure 3)
- In the **JBI Modules** area, there is a BPEL **Service Unit** with a **Provide endpoint** connected to the **Consume endpoint** of a SOAP port, **casaport1**, on the **WSDL Ports** area. However, the correct configuration is to connect the service to another SOAP port, **port1**.
- Click on the connection to select it.
- Click on the DELTE key to delete the selected connection.

- Click on the **Consume endpoint** of **port1** and drag the mouse over to the **Provide endpoint** of the Bpel **Service Unit** to create a connection.
- Expand the **port1** node on the **CASA Navigator Panel**. (see Figure 5)
- Select WSDL port or binding nodes to edit their attributes in the **CASA Property Panel**. (see Figure 6)
- Click on the "x" mark on the tab of **CASA Editor Window** to close it.
- Select **Save** button to save the changes. (see Figure 7)
- Build and deploy the composite application.

1.4 Connect to endpoints of an external service unit

A service assembly can have connections across its boundary to endpoints of external service units. In this use case, the user uses the CASA editor to make a connection to an endpoint of an external service unit outside of the composite application. The CASA editor provides support for specifying endpoints of external service units developed outside of the Netbeans enterprise pack.

Scenario:

- Select the composite application project containing the bpel project.
- Invoke project **Context Menu** item **Edit Project.**. (see Figure 1)
- A **CASA Editor Window** opens showing the **Design View** of the composite application project configuration. (see Figure 3)
- In the **JBI Modules** area, there is a BPEL **Service Unit** with a **Consume endpoint**. The use needs to connect this endpoint to an endpoint of an external service unit.

Sub-case 1: the external service unit is another bpel project

- Select the external bpel project and drag-n-drop it onto the **Ext. Modules** area of the **Design View**.
- In the **Ext. Modules** area, it shows a BPEL **Service Unit** with a **Provide endpoint**.
- Click on the **Consume endpoint** and drag the mouse over to the **Provide endpoint** to create a connection.
- Click on the "x" mark on the tab of **CASA Editor Window** to close it.
- Select **Save** button to save the changes. (see Figure 7)
- Build and deploy the composite application.

Sub-case 2: the external service unit does not have a Netbeans project

- Select the **External SU** palette item of the **Service Unit** category on the **CASA Palette Panel**. (see Figure 4)
- Drag and drop it onto the **Ext. Modules** area of the **Design View** window.
- A **Service Unit** is created on the **Ext. Modules** area.

- Select the **Provide Endpoint** palette item from the **Endpoint** category on the **CASA Palette Panel**. (see Figure 4)
- Drag-n-drop it onto the **Service Unit** in the **Ext. Modules** area.
- A **Provide Endpoint** is created for the external **Service Unit**.
- Expand the external **Service Unit** node on the **CASA Navigator Panel**. (see Figure 5)
- Select the Provide nodes to edit its attributes in the **CASA Property Panel**. (see Figure 6)
- In the **Design View**, click on the **Consume endpoint** and drag the mouse over to the **Provide endpoint** to create a **connection**.
- Click on the "x" mark on the tab of **CASA Editor Window** to close it.
- Select **Save** button to save the changes. (see Figure 7)
- Build and deploy the composite application.

1.5 Attach WS policy definitions

In a composite application, a WSDL port can be attached with WS and other policy definitions that add requirements and/or preferences to the web service provider and/or consumers. In this use case, the user uses the CASA editor to attach a WS policy defined by the Web Services Interoperability Technology (WSIT) module in the composite application. The policy definition is persisted in the deployment configuration, not the underlying bpel project implementing the web service.

Scenario:

- Select the composite application project.
- Invoke project **Context Menu** item **Edit Project.**. (see Figure 1)
- A **CASA Editor Window** opens showing the **Design View** of the composite application project configuration. (see Figure 3)
- In the **JBI Modules** area, there is a BPEL **Service Unit** with a **Provide endpoint** connected to the **Consume endpoint** of a SOAP port on the **WSDL Ports** area.
- Right click on the SOAP port on the **WSDL Ports** area and invoke the **Attach Policy** menu item. (see Figure 8)
- The **WSIT Policy Definition Panel** from the WSIT module is opened. (see Figure 9)
- Edit the WS policy definition attributes.
- Click on the OK button to save the policy definition.
- Click on the "x" mark on the tab of **CASA Editor Window** to close it.
- Select **Save** button to save the changes. (see Figure 7)
- Build and deploy the composite application.

2. Specification

2.1 Project Context Menu – Edit Project

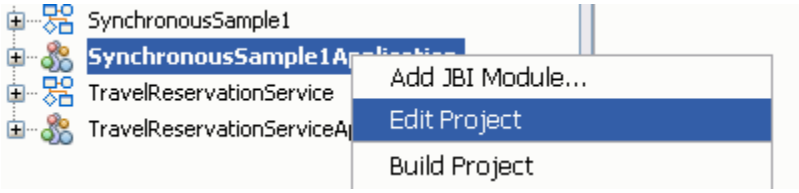

Figure 1: Composite Application project Context Menu, Edit Project menu item

2.2 CASA Editor Window - Source View

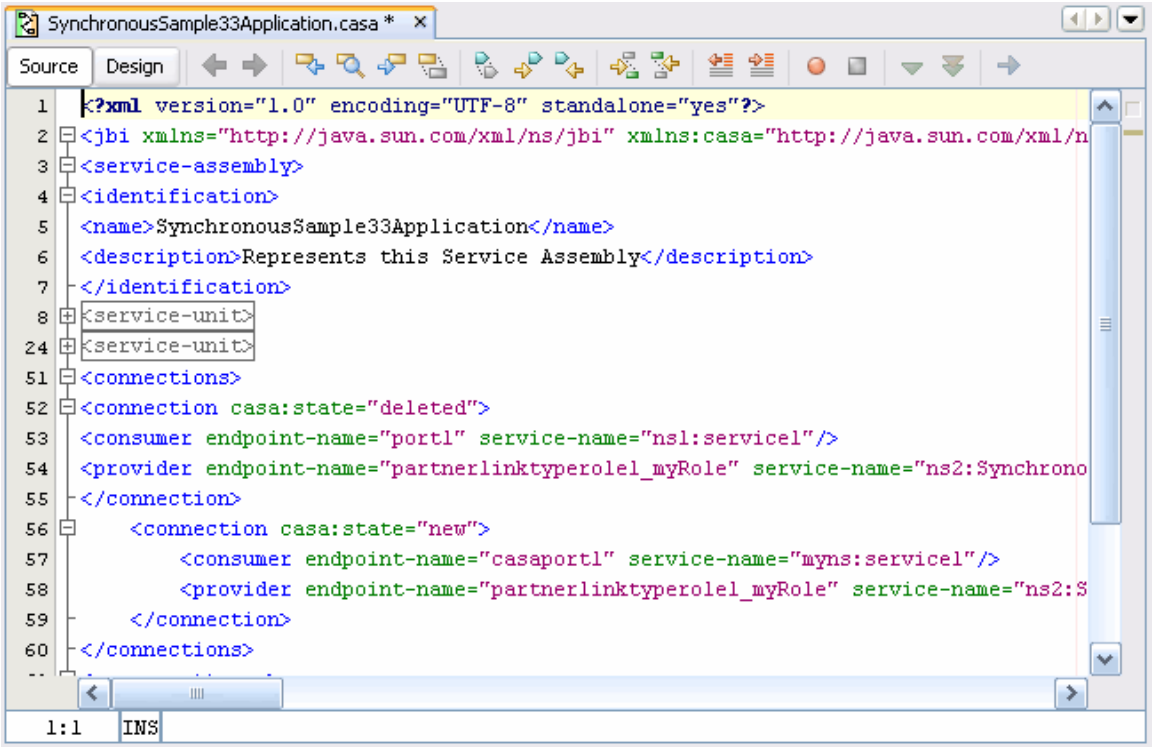

Figure 2: The Source View of the CASA editor window

Components:

- **Source** button - if selected, the XML source of the composite application deployment configuration file is displayed.
- **Design** button - if selected, the graph view of the composite application deployment configuration file is displayed.
- **Toolbar** buttons – these are standard Netbeans XML editor toolbar buttons.

2.3 CASA Editor Window - Design View

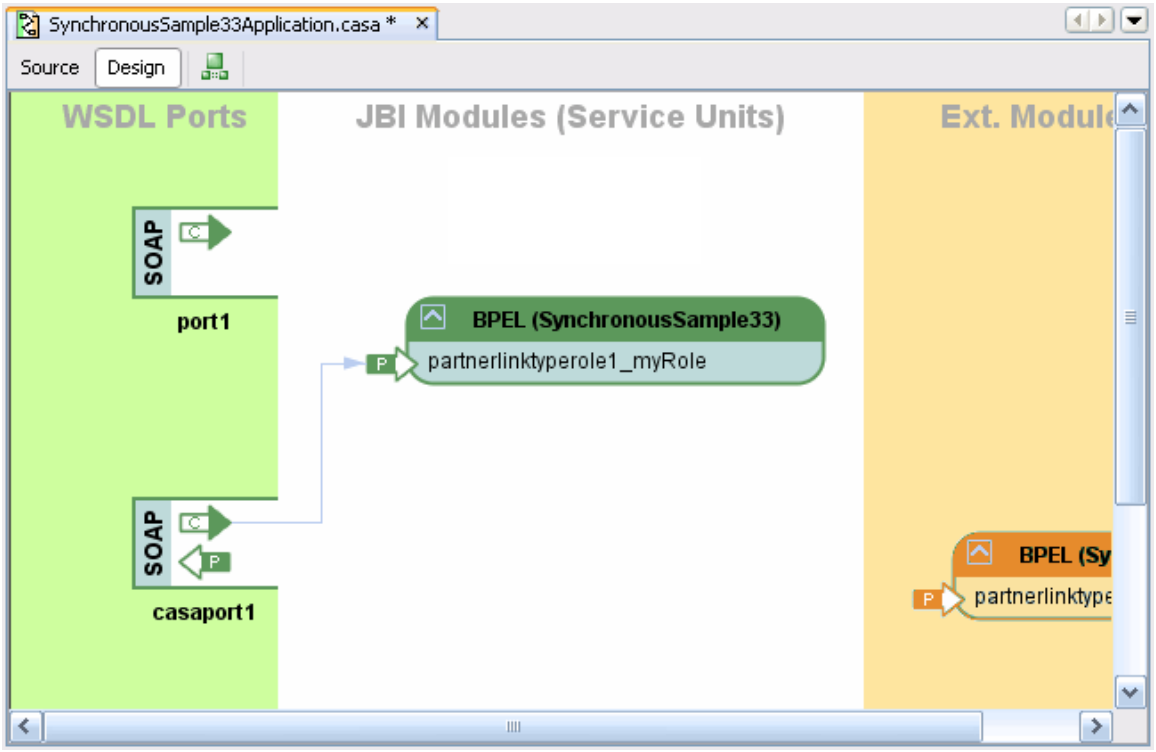

Figure 3: The Design View of the CASA editor window. The content shown contains two WSDL ports, a connection, a BPEL service unit in the JBI Modules area, and an external service unit in the Ext. Modules area.

Components:

- **WSDL Ports** area – the green area on the left is for placing WSDL ports.
- **JBI Modules** area – the middle white area contains JBI modules, i.e., service units of the composite application.
- **Ext. Modules** area – the orange area on the right is for external service units that are not packaged within the composite application.
- **WSDL port** icon – each representing a WSDL port with labels showing its binding type (left) and port name (bottom).
- **Consume endpoint** icon – the arrows icon with a "C" represents a consume endpoint as defined by a JBI service unit.
- **Provide endpoint** icon – the arrows icon with a "P" represents a provide endpoint as defined by a JBI service unit.
- **Service Unit** icon – the rounded rectangle box represents a service unit.
- **Service Unit** title bar – It contains a Open/Close toggle box, a JBI service engine type label, and a service unit name label.

- **Service Unit** endpoint list – The area below title bar is a list of endpoints within the service unit. A service unit may have 0 or more consume and provide endpoints.

2.4 CASA Palette Panel

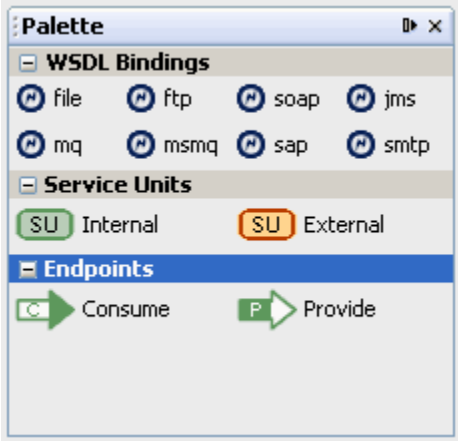

Figure 4: CASA Palette Panel

Components:

- **WSDL Binding** palette category – Each palette item in this category represents a WSDL binding type, e.g., SOAP, FILE, JMS, SMTP, etc.
- **Service Units** palette category - Each palette item in this category represents a service unit type, either internal or external.
- **Endpoints** palette category – Each palette item in this category represents a service unit endpoint type, either consume or provide.

2.5 CASA Navigator Panel

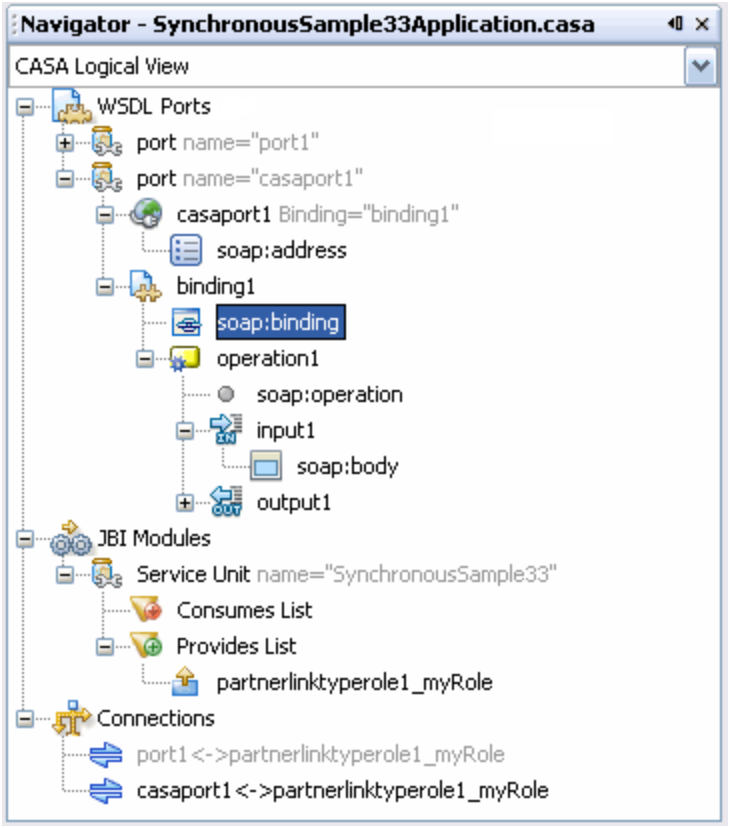

Figure 4: CASA Palette Panel

Components:

- **WSDL Ports** node – The root node of all WSDL ports shown in the CASA editor **Design View**.
- **port** node – The port node represents a WSDL port. It has two children nodes: a port element and a binding element from the associated WSDL file.
- **JBI Modules** node – The root node of all JBI service units shown in the CASA editor **Design View**.
- **service unit** node – The service unit node represents a JBI service unit. It has two children nodes: a consume endpoint list and a provide endpoint list.
- **Connections** node – The root node of all connections shown in the CASA editor **Design View**.
- **connection** node – The connection node represents a connection between two endpoints.

2.6 CASA Properties Panel

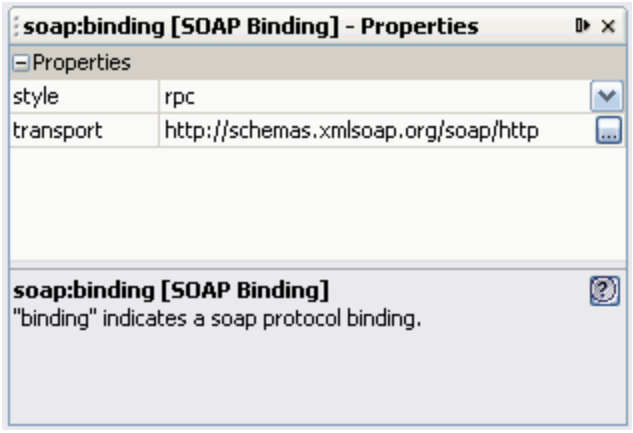

Figure 6: CASA Properties Panel. The content shown is the XML attributes of the "soap:binding" element selected in Figure 5.

<soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>

Components:

- **Properties** section – This is a grouping of attributes within an XML element.
- **Attribute** name – The left column is the name of the attribute.
- **Attribute** value – The right column is the attribute value. It has an associated property editor for editing attribute value.

2.7 Save Dialog

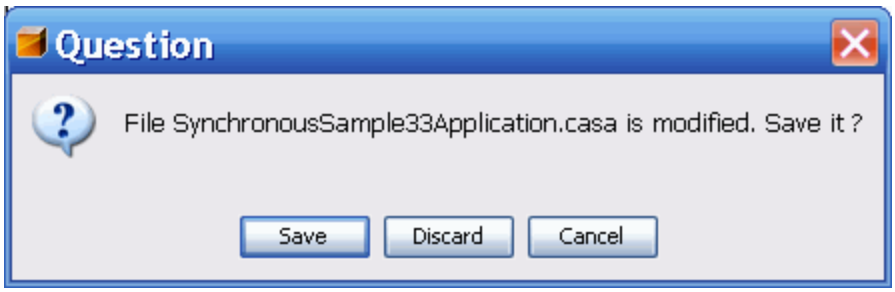

Figure 7: Save Dialog

Components:

- **Save** button – Save modifications and close the editor.
- **Discard** button – Discard modifications and close the editor.
- **Cancel** button – Cancel the close action.

2.8 WSDL Port Context Menu – Attach Policy

Figure 8: WSDL Port Context Menu, Attach Policy menu item

2.9 WSIT Policy Definition Panel

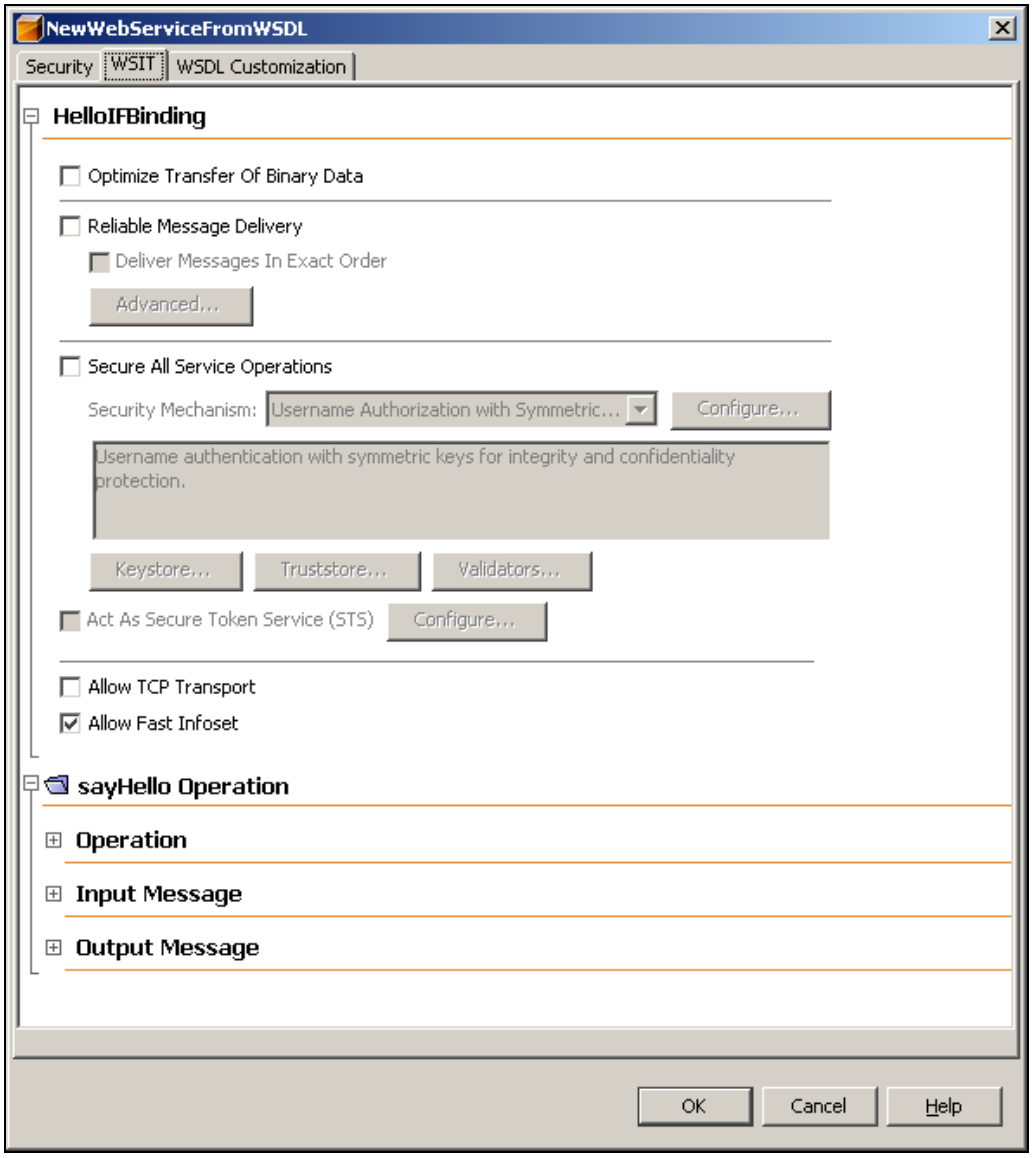

Figure 9: WSIT Policy Definition Panel (This UI is generated by the WSIT module)

3. References

1. CASA Design Use Cases,
   http://alaska.stc.com:10000/alaska/attach/CASAEditor/casadesign.pdf

2. Netbeans UI Spec Template,
   http://ui.netbeans.org/docs/hi/uireview/ui-spec-template.html