Proposal for Bug 3599

URI References for WSDL 1.0 and other Message Definition Mechanisms

Section 3.4 of the PolicyAttachment document defines an external attachment mechanism that allows policies to be associated with a policy subject independent of that subject's definition and/or representation through the use of a `wsp:PolicyAttachment` element. The PolicyAttachment element has a child element called `wsp:AppliesTo` which takes a DomainExpression as argument. The DomainExpression refers to the Policy Subject.

Section 4 of the PolicyAttachment document discusses attaching policies to WSDL 1.1 by attaching policies directly to WSDL 1.1 components. Thus, attaching policies to WSDL 1.1 components is considered to be important. However, in cases where the WSDL cannot or should not be modified, there is no way to use the external attachment mechanism to attach policies to WSDL 1.1 components because there is no way to refer to individual WSDL 1.1 components. References to individual WSDL 1.1 components is required to specify different policy subjects.

This proposal addresses this deficiency by proposing a mechanism for referring to WSDL 1.1 components using URI references. For Web Services that do not use SOAP and WSDL, it also defines mechanisms for referring to HTTP messages and JMS messages.

For policies associated with WSDL components using the external attachment mechanism, the effective policy is calculated using the algorithms described in Section 4.1 of the PolicyAttachment spec for directly attached policies.

For HTTP Service and JMS Service there is no need to calculate effective policy.
For HTTP, policies can only be attached to an endpoint. For JMS they can be attached only to a message queue.

Referring to WSDL 1.1 Components

WSDL 2.0 discusses how URIs can be created to refer to individual WSDL components. For WSDL 1.1. we suggest a mechanism similar to the WSDL 2.0 mechanism for generating URIs. Essentially, a URI identifies the WSDL component and a fragment identifier identifies the particular definition.

Consider a WSDL 1.1 file that is available at http://samples.otn.com.LoanFlow.
Now consider that you want to associate a policy with a portType defined in this file. First you construct a URI reference for the portType such as http://samples.otn.com.LoanFlow#wsdl.PortType(LoanFlowPortType) by adding the fragment identifier “wsdl.PortType(LoanFlowPortType)” to the URL for the file where “LoanFlowPortType” is the name of the port type. Then you use this URI reference as a DomainExpression in a PolicyAttachment element.

In the same way, URI References can be constructed for other types of WSDL 1.1 components by adding a fragment idenand used as a DomainExpression in a PolicyAttachment.

The following sections specify in detail how the fragment identifier is constructed for individual WSDL 1,1 components:

```
1.    The PortType Component
wsdl.portType(portType)
Where "portType" is the local name of the {name} property of the
PortType component.

2.    The PortType Operation Component
wsdl.portTypeOperation(portType/operation)
Where "portType" is the local name of the {name} property of the parent
PortType component and "operation" is the local name of the {name}
property of the PortType Operation component.

3.    The PortType Operation Input/Output Message Component
Where wsdl.portTypeOperation[Input|Output](portType/operation)
"portType" is the local name of the {name} property of the grandparent
PortType component and "operation" is the local name of the {name}
property of the parent PortType Operation component.

4.    The PortType Operation Fault Message Component
Where wsdl.portTypeOperationFault(portType/operation/fault)
"portType" is the local name of the {name} property of the grandparent
PortType component and "operation" is the local name of the {name}
property of the parent PortType Operation component and
"fault" is the {name} property of the Operation Fault component.

5.    The Binding Component
wsdl.binding(binding)
Where "binding" is the local name of the {name} property of the Binding
component.

6.    The Binding Operation Component
Where wsdl.bindingOperation(binding/operation)
"binding" is the local name of the {name} property of the parent
Binding component and "operation" is the {name} property of the
PortType Operation component referred to by the {portType operation}
property of the Binding Operation component.

7.    The Binding Operation Message Input/Output Component
wsdl.bindingOperation[Input|Output](binding/operation)
Where "binding" is the local name of the {name} property of the
grandparent Binding component and "operation" is the {name} property of
the PortType Operation component referred to by the {portType
operation} property of the parent Binding Operation component.

8.    The Binding Operation Fault Message Component
wsdl.bindingOperationFault(binding/operation/fault)
Where "binding" is the local name of the {name} property of the
grandparent Binding component and "operation" is the {name} property of
the PortType Operation component referred to by the {portType
operation} property of the parent Binding Operation component and
"fault" is the {name} property of the Operation Fault component.
```

9. The Service Component
wsdl.service(service)
Where “service” is the local name of the {name} property of the Service component.

10. The Endpoint Component
wsdl.endpoint(service/port)
Where “service” is the local name of the {name} property of the parent Service component and “port” is the {name} property of the Port component.

11. The Endpoint Operation Component
wsdl.endpointOperation(service/port)
Where “service” is the local name of the {name} property of the parent Service component and “port” is the {name} property of the Port component.

12. The Endpoint Input/Output Component
wsdl.endpointOperation[Input|Output](service/port)
Where “service” is the local name of the {name} property of the parent Service component and “port” is the {name} property of the Port component.

13. The Endpoint Fault Component
wsdl.endpointOperation[Fault](service/port)
Where “service” is the local name of the {name} property of the parent Service component and “port” is the {name} property of the Port component.

14. Extension Components
WSDL 1.1 is extensible and it is possible for an extension to define new components types. The scheme for extension components is: wsdl.extension(namespace, identifier) where “namespace” is the namespace URI that identifies the extension, e.g. for the WSDL 1.1 SOAP 1.1 Binding the namespace is "http://schemas.xmlsoap.org/wsdl/soap/" and for SOAP 1.2 Binding "http://www.w3.org/2005/08/wsdl/soap". identifier is defined by the extension using a syntax specific to the extension. The owner of the extension must define any components contributed by the extension and the syntax for identifying them.

Referring to HTTP and JMS Components

HTTP Service:
a. Endpoint: .
i.e. http://example.com/LoanFlow

b. Endpoint Pattern:
This identifies all HTTP Services at a specific url.
i.e. http://example.com/*

c. Endpoint Pattern:
This identifies all HTTP Services within a specific webapp.

JMS Service: jms://host:port/factoryJndiName/destJndiName
i.e. jms://example.com:7115/quoteQCF/LoanFlowQueue