ESB Adapter Providers

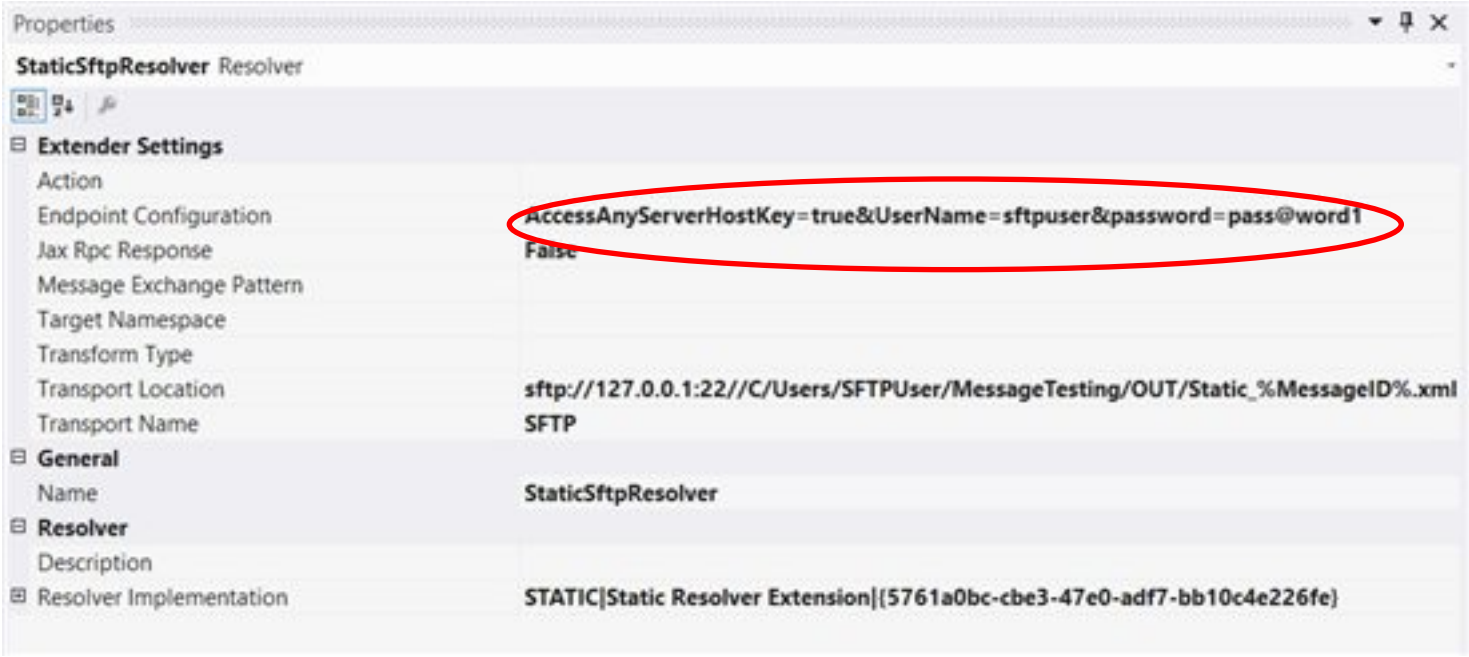
Luigi Pampaloni
BizTalk Solutions Architect
LPYSoft Ltd

What is an ESB Adapter Provider

- In an Itinerary if you want to dynamically call a send port by specifying all the transmission properties (transport type, address etc) as Static or BRE the adapter you call needs to implement an Adapter Provider
- All out of the box BizTalk Adapters are supported by the ESB natively (an Adapter Provider for each is provided by the ESB Toolkit)
- Your custom adapter needs to implement an Adapter Provider if you want to be called from a BRE based Resolver

Static Resolver

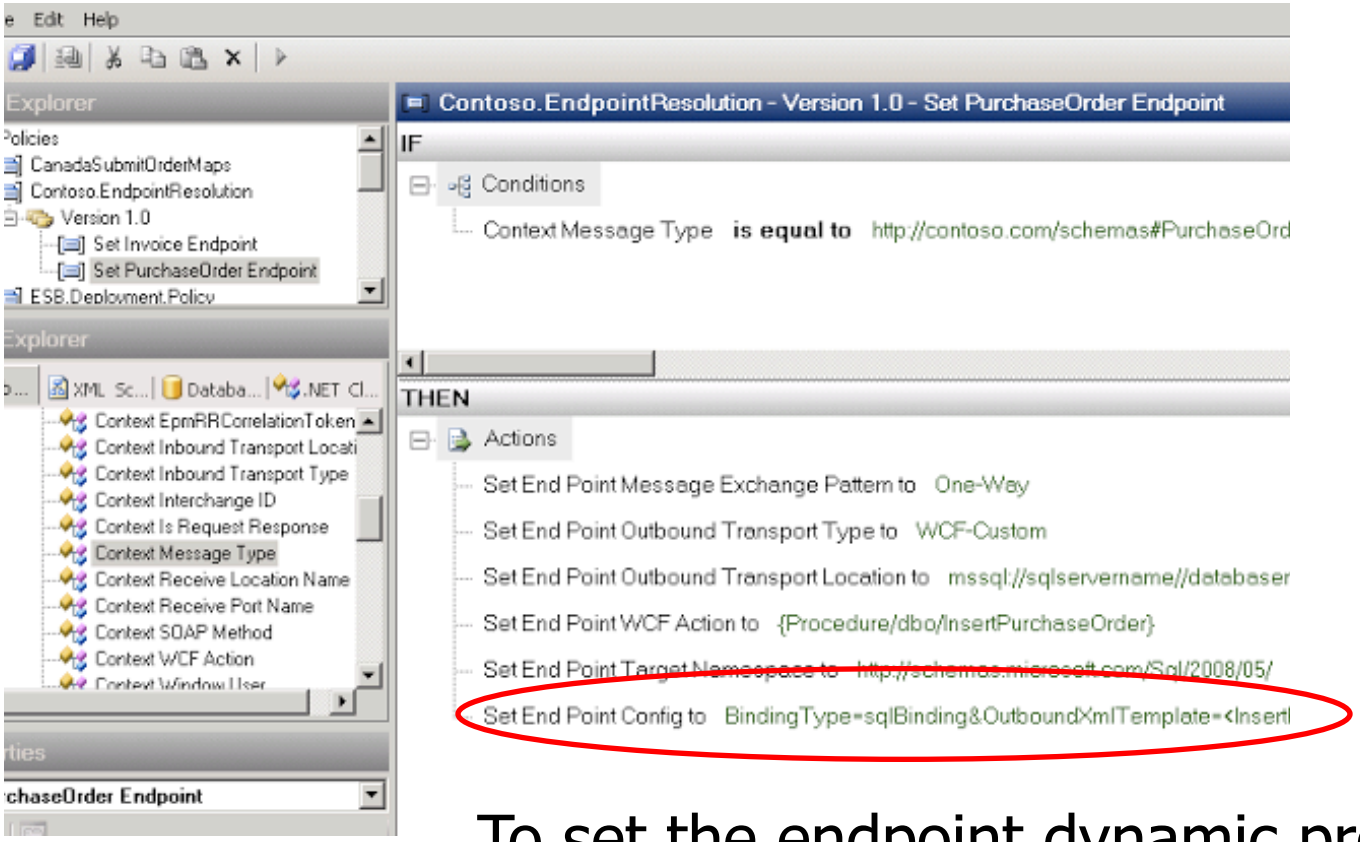
- The Resolver may use a Static port like in the screenshot



To set the endpoint dynamic properties the string in the Static port needs to be converted in actual message context properties

BRE Resolver

- The Resolver may use a BRE like the rule in the screenshot



To set the endpoint dynamic properties the string in the Rule needs to be converted in actual message context properties

How to implement an Adapter Provider

- And Adapter Provider is a .NET component (Class Library)
- Add the following references:
  - Microsoft.BizTalk.GlobalPropertySchemas
  - Microsoft.BizTalk.BizTalk.Pipeline
  - Microsoft.XLANGs.BaseTypes
  - Microsoft.Practices.ESB.Adapter

Add a class with the following code

- Public class deriving from BaseAdapterProvider
  - public class AdapterProvider : BaseAdapterProvider
- Implement a AdapterName property
  - public override string AdapterName
  - {
  - get { return "SimpleHTTP"; }
  - }
- Overrides a SetEndpoint method
  - public override void SetEndpoint(Dictionary<string, string> resolverDictionary, XLANGMessage message)

SetEndpoint

- Calls the base SetEndpoint and retrieve the base transport properties
  - base.SetEndpoint(resolverDictionary, message);
  - string transportLocation = resolverDictionary["Resolver.TransportLocation"];
  - string outboundTransportCLSID = resolverDictionary["Resolver.OutboundTransportCLSID"];
  - string transportType = resolverDictionary["Resolver.TransportType"];
- Sets the context properties
  - message.SetPropertyValue(typeof(BTS.OutboundTransportLocation), transportLocation);
  - message.SetPropertyValue(typeof(BTS.OutboundTransportType), transportType);
  - message.SetPropertyValue(typeof(BTS.OutboundTransportCLSID), outboundTransportCLSID);

SetEndpoint

- Gets the endpoint config
  - string endpointConfig = resolverDictionary["Resolver.EndpointConfig"];

- Parses the string and assigns the properties

```
if (!string.IsNullOrEmpty(endpointConfig))
        {
            // parse delimited endpointconfig and set SFTP specific adapter properties
            // endPointConfig data with this format "Key1=Value1;Key2=Value2;...."
            var config = endpointConfig.Split(';').Select(part => part.Split('=')).ToDictionary(split =>
split[0], split => split[1]);

            // Set the context for the SFTP adapter
            if (config.ContainsKey("Method"))
            {
                message.SetPropertyValue(typeof(SimpleHTTP.Method), config["Method"]);
            }
            if (config.ContainsKey("ContentType"))
            {
                message.SetPropertyValue(typeof(SimpleHTTP.ContentType), config["ContentType"]);
            }
        }
```

SetEndpoint properties parsing

- The implementation just splits by “;” to get key-value pairs then splits by “=” to separate the key from the value
- Simple to read and maintain but your configuration cannot contain ; or = as values (i.e. your password cannot contain = or any base64 encoded string)
- Better implementation is to embed an XML fragment and parse using the DOM

How to deploy the Adapter Provider

- The deployment requires:
  - On all the BizTalk nodes, GAC the Adapter Provider dll
  - Edit the esb.config file to add the following XML fragment under adapterProviders:

```
- <adapterProvider
-   name="SimpleHTTP"
-   type="HTTPAdapterESBProvider.AdapterProvider,
HTTPAdapterESBProvider, Version=1.0.0.0, Culture=neutral,
PublicKeyToken= "
-   moniker="SimpleHTTP" />
```

- Restart the BizTalk Services

How to call the adapter provider

- Now you can call the Off-Ramp on your Itinerary and the dynamic port transmission detail can be stored in a rule

Simple HTTP Adapter Provider

- A sample Adapter Provider source code is attached and implements the Adapter Provider for the Simple HTTPAdapter