Integrated Capture Points Guide

Web Services Capture Point—Generate a .NET Client

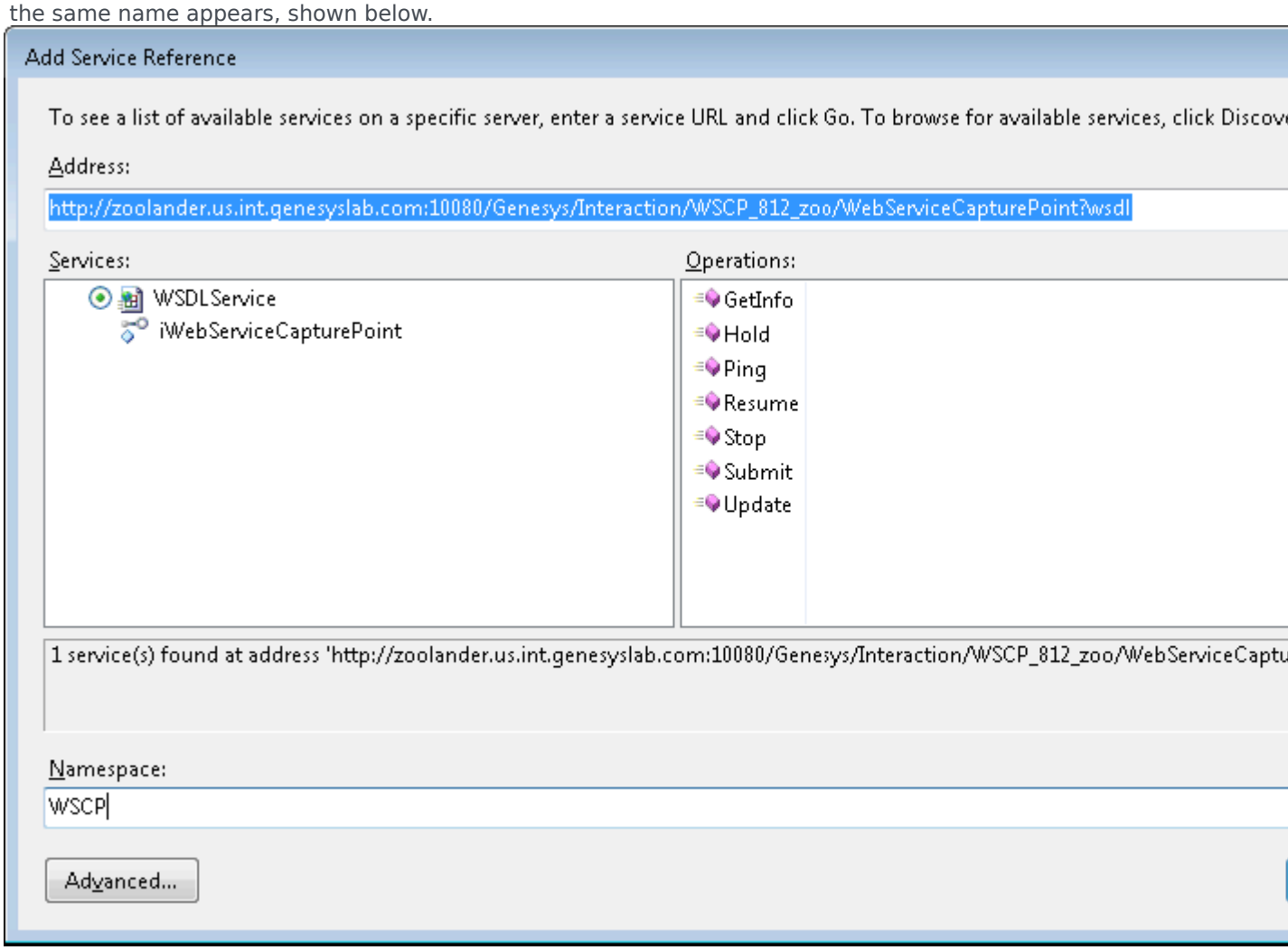
5/11/2025

Web Services Capture Point—Generate a .NET Client

This page provides an example of generating a .NET Client. See the list of tools used to generate clients in this document.

Start

1. Open Visual Studio 2010 and create a C# Win32 console application.
2. In Solution Explorer right-click `References` and choose `Add Service Reference.` The dialog box of the same name appears, shown below.



Add Service Reference

3. Enter the WSDL URL of the Web Service Capture Point.

4. Enter the service namespace (for example, `WSCP`):
5. Click Go.

   Provided Interaction Server is running and the WSDL URL is specified correctly, `WebServiceCapturePoint` should appear in the `Services` list.
6. Click `OK` to generate the service client.
7. To test the service, open the `Program.cs` file and insert the following code in the `main` method:

```
WSCP.iWebServiceCapturePointClient client = new WSCP.iWebServiceCapturePointClient();
// This is an optional step to reconfigure the client to use different endpoint.
// It's usually done using configuration setting for the application
//client.Endpoint.Address = new System.ServiceModel.EndpointAddress(
//    "http://localhost/Genesys/Interaction/MyCP/WebServiceCapturePoint");
// Create a key-value list of extensions and specify the signature,
// so we can recognize the request in Interaction Server log
var extension = new WSCP.KVList();
extension.Add(new WSCP.KVPair() { key = "signature",
value = new WSCP.KVPairValue() { ValueString = ".Net WSCP test client" } });
// We expect ping info back in Ping response
WSCP.KVList userdata = null;
WSCP.KVList pinginfo = null;
try
{
    // Ping the server and get some statistics back
    client.Ping(out userdata, out pinginfo, ref extension);
    Console.Out.WriteLine(trace_list(pinginfo));
}
catch (FaultException<FaultMessage> ex)
{
    // process WSCP specific error code
  Console.Out.WriteLine("Error {0}: {1}",
ex.Detail.ErrorCode, ex.Detail.ErrorDescription);
}
catch (Exception ex)
{
    Console.Out.WriteLine(ex.ToString());
}
```

8. Add the method `trace_list` to your program to output the server response:

```
static string trace_list(WSCP.KVList list, string indent = "")
{
    StringBuilder result = new StringBuilder();
    list.ForEach((item) =>
    {
        result.Append(indent);
        result.Append(item.key);
        if(null != item.value.ValueString)
        {
            result.Append(" [string] = ");
            result.Append(item.value.ValueString);
            result.Append('\n');
        }
        else if (null != item.value.ValueList)
        {
            result.Append(" [list] = \n");
            result.Append(trace_list(item.value.ValueList, indent + "    "));
        }
        else
        {
```

```
                result.Append(" [int] = ");
                result.Append(item.value.ValueInt.ToString());
                result.Append('\n');
            }
        });
        return result.ToString();
    }
```

This simple test prints Interaction Server statistics into a console window. You can then discover the service methods using autocompletion and the object browser.

End