Tutorial: Consumir un servicio sincrónico de la PGE

Control de Cambios

Fecha	Versión	Descripción	Autor	Aprobado Por
15/11/10	1.0	Versión inicial	Tecnología – Arquitectura y Normas	
13/08/2014	1.9			

Nombre actual del archivo: TI-PRO-MNL-PGE-Consumo Servicio dotNET-Parte2-v01-09.odt

Objetivo

El objetivo de este tutorial es proveer una guía paso a paso para el desarrollo de un cliente *desktop* de la Plataforma de Gobierno Electrónico (PGE) sobre la plataforma .NET.

Prerrequisitos

Se asume que el usuario conoce las especificaciones WS-Security, WS-Trust, SAML 1.1. Además, se asume que el usuario está familiarizado con el uso de certificados, aplicaciones .NET y Web Services.

Requerimientos del software

La tabla 1 presenta las herramientas y productos de *software* requeridos para desarrollar y ejecutar la Aplicación Cliente.

Producto	Versión
.NET Framework	3.5
Visual Studio Express	2008

Tabla 1 – Requerimientos de Software

Descripción del escenario

La figura 1 presenta el escenario de ejemplo que se utiliza en este tutorial, en el cual intervienen dos organismos: el Banco de Previsión Social (BPS) (Organismo Cliente) y el Ministerio de Salud Pública (MSP) (Organismo Proveedor).

El MSP provee el servicio “Certificado de Nacidos Vivos”. Cuando se registró el servicio en la PGE, se desplegó un Servicio Proxy en ella para que las Aplicaciones Cliente accedieran al servicio a través de él. Además, mediante la configuración de políticas de control de acceso, el MSP autorizó a los usuarios con rol “doctor” de la sección “prestaciones” del BPS a consumir todos los métodos del servicio.

Por otro lado, en el BPS hay una Aplicación Cliente que está siendo utilizada por el usuario Juan que tiene el rol “doctor” en la sección “prestaciones”. La aplicación necesita acceder al servicio del MSP para lo cual, utilizando las credenciales del usuario Juan y a través de una Aplicación Emisora de Tokens interna al BPS, obtiene un *token* de seguridad SAML firmado por el BPS (pasos 1.a y 1.b).

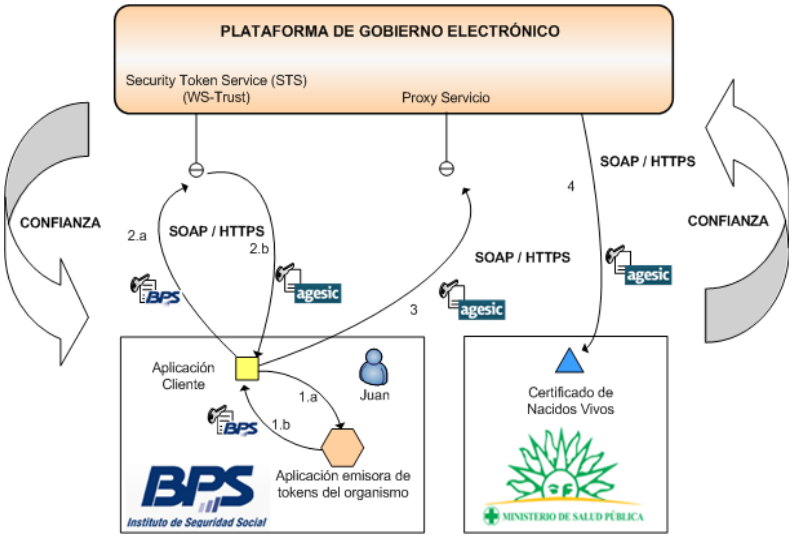


Figura 1: Escenario de uso

Luego con dicho *token* obtiene del STS de la PGE, a través del estándar WS-Trust, otro *token* de seguridad firmado por la plataforma (pasos 2.a y 2.b). Para emitir este *token* la PGE verifica la firma digital del *token* enviado por la aplicación y la existencia del rol "*ou=gerencia de proyectos,o=agesic*".

Por último, la Aplicación Cliente invoca al Servicio del MSP mediante su Servicio Proxy. En la invocación se incluye el *token* firmado por la PGE y se especifican el servicio (Certificado de Nacidos Vivos). Dado que el usuario Juan está autorizado a utilizar el servicio, la invocación se efectúa de forma exitosa.

La tabla 2 especifica algunos de los datos a utilizar en la implementación del escenario.

Dato	**Valor**
Nombre de Usuario	Juan
Rol de Usuario	ou=gerencia de proyectos,o=agesic
Dirección Lógica del Servicio	http://testservicios.pge.red.uy/msp/certificadoCNVE
Método del Servicio	getDatosPersonales
PolicyName[1]	urn:tokensimple
Tipo de *Token*[2]	http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV1.1

Tabla 2 – Datos para la Implementación del Escenario

[1] Es la política de autenticación utilizada por AGESIC para la verificación de solicitudes del cliente. Actualmente el único valor posible es "urn:tokensimple)".

[2] Actualmente la PGE acepta la emisión de *tokens* SAML versión 1.1.

Los datos de negocio a incluir en la invocación, están especificados en la descripción del servicio (WSDL). En esta descripción también se incluye la dirección del Servicio Proxy a donde el cliente debe enviar los mensajes SOAP para invocar al servicio.

Implementación del escenario

En esta sección se describe, paso a paso, la implementación de una Aplicación Cliente .NET de escritorio según el escenario descrito previamente.

La implementación del escenario comprende las siguientes etapas:

- Crear proyecto .NET consola y agregar librerías de apoyo
- Crear una referencia al servicio
- Configurar WS-Addressing
- Configurar la conexión SSL
- Configurar comunicación con el STS de la PGE
- Invocación del Servicio

En las siguientes sub-secciones se describen en detalle cada una de ellas.

Crear proyecto .NET Consola y agregar librerías de apoyo

1. Seleccionar *File* → *New* → *Project* → *Visual C#* → *Console Application* y crear un proyecto con nombre ClienteTutorial, y nombre de la solución Tutorial como se muestra en la figura 1.

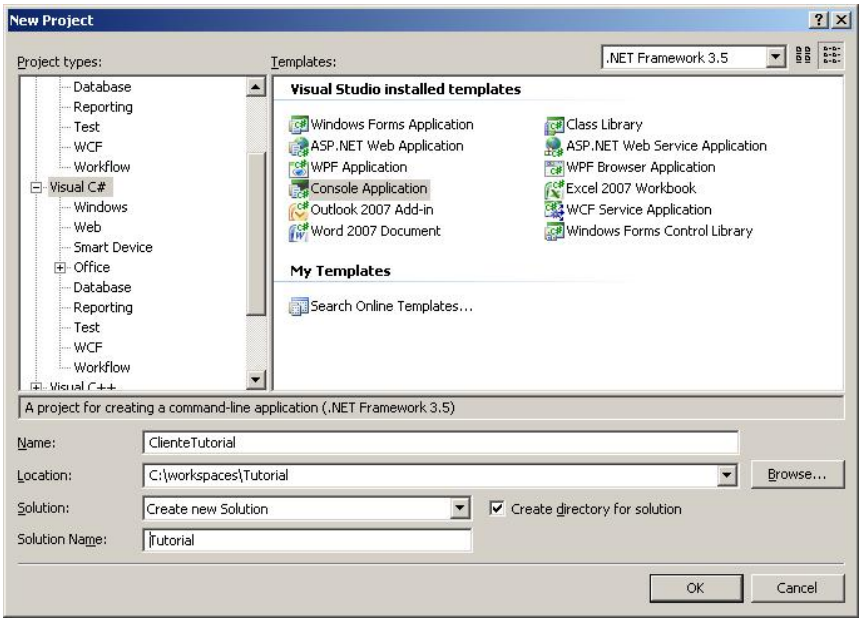


Figura 2: Creación de un proyecto .NET

2. Hacer clic derecho en la solución → *Add* → *Existing Project...* como se muestra en la figura 3.

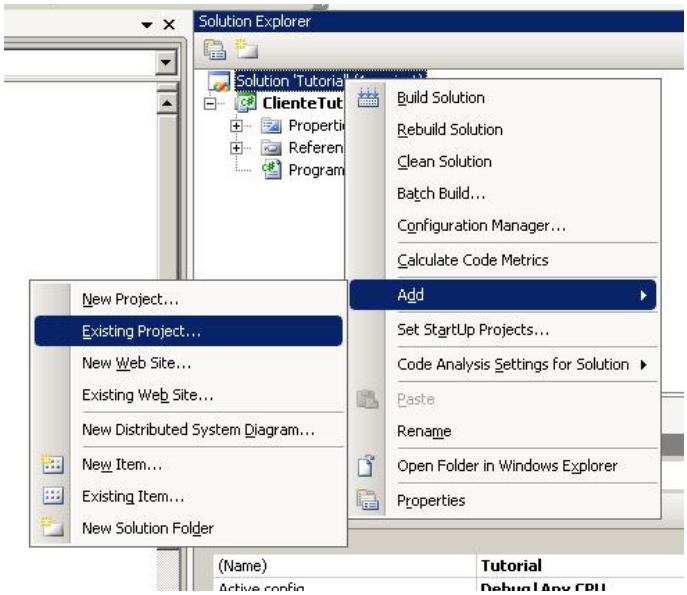


Figura 3: Agregar un proyecto existente

3. Seleccionar la ubicación del proyecto PGE extraido de la carpeta materiales. El resultado esperado debe ser similar al de la figura 4.

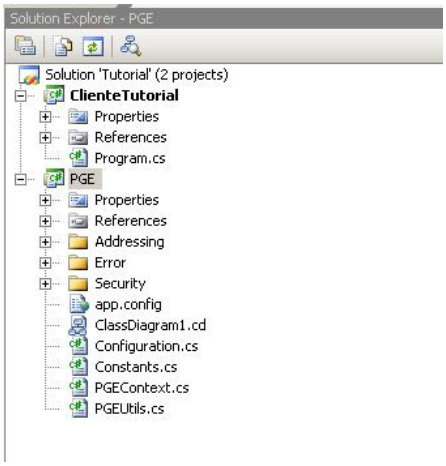


Figura 4: Proyecto agregado

4. Dentro del proyecto ***ClienteTutorial***, hacer clic derecho en *References* → *Add Reference...*, seleccionar la solapa *projects* y luego el proyecto PGE, como se muestra en la figura 5.

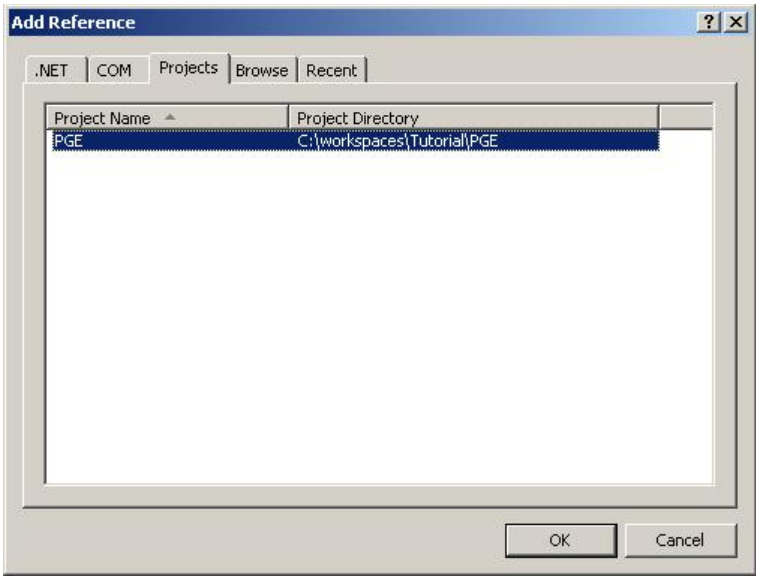


Figura 5: Agregar como referencia el proyecto PGE

Crear Referencia al Servicio

1. Hacer clic derecho en el proyecto ***ClienteTutorial*** y seleccionar *Add Service Reference...*

2. Especificar la dirección del WSDL del servicio como se muestra en la figura 6 y presionar el botón Go. El WSDL a seleccionar es wsdl_base.wsdl.

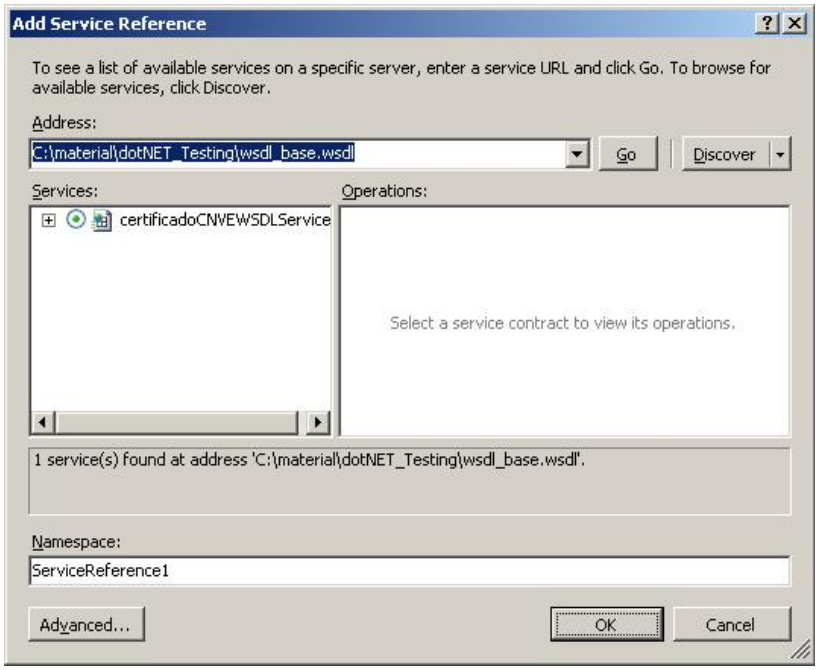


Figura 6: Agregar referencia al Web Service

3. Modificar el namespace a Tutorial y presionar el botón OK. Este paso creará un archivo de configuración del Web Service llamado app.config.

Configurar WS-Addressing

1. Hacer clic derecho en el archivo app.config y seleccionar *Edit WCF Configuration* como se muestra en la figura 7.

Nota: Debido a un bug en Visual Studio 2008, es necesario antes activar la herramienta gráfica de utilizar la opción *Edit WCF Configuration*. Para activarla, es ir al menú superior → *Tools* → *WCF Service Configuration Tool.*

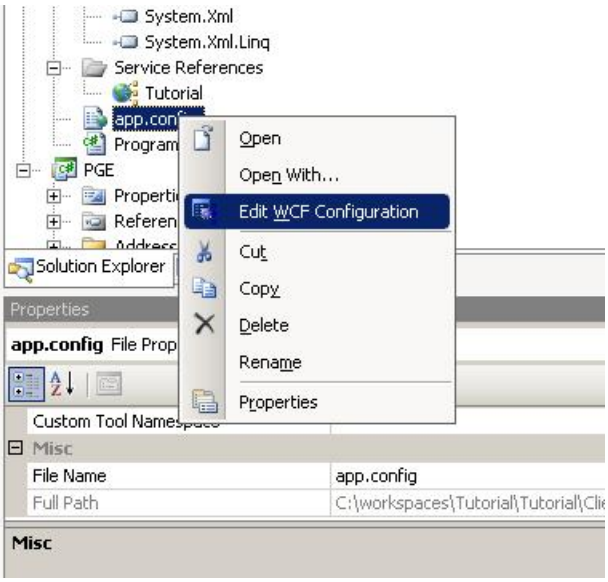


Figura 7: Editar el archivo de configuración del WS

2. Seleccionar *Advanced* → *Endpoint Behavior* → *New Endpoint Behaviour Configuration*

3. Nombrar al nuevo behaviour como PGEBehaviour

4. Presionar el botón Add..., luego client via como se muestra en la figura 8 y luego el botón Add.

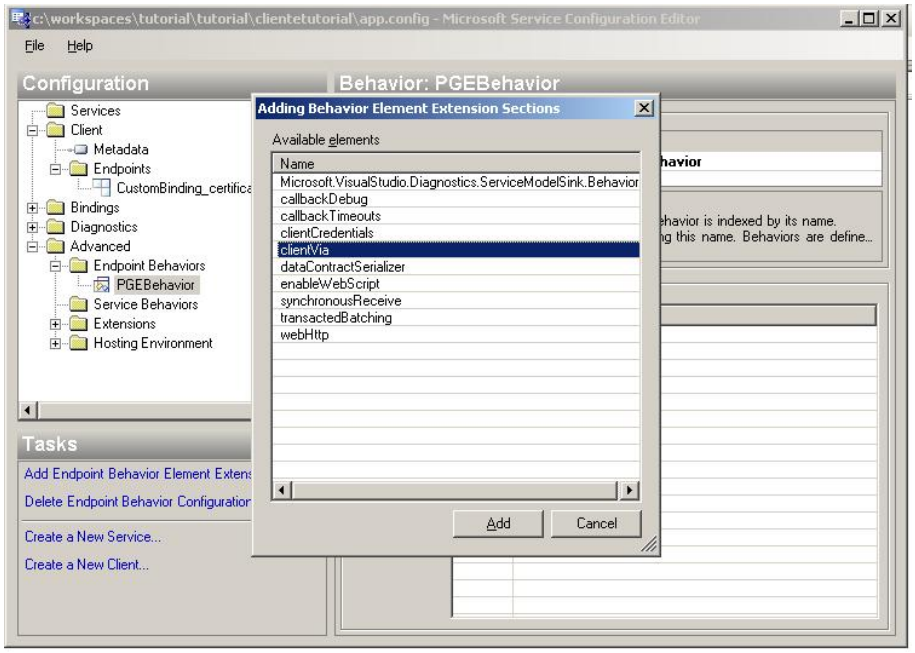


Figura 8: Definir el client via del endpoint

5. Definir *ViaUri* con el valor `https://testservicios.pge.red.uy:6187/ws-cnve/certificadoCNVE`
6. Este valor, representa la dirección física del servicio.

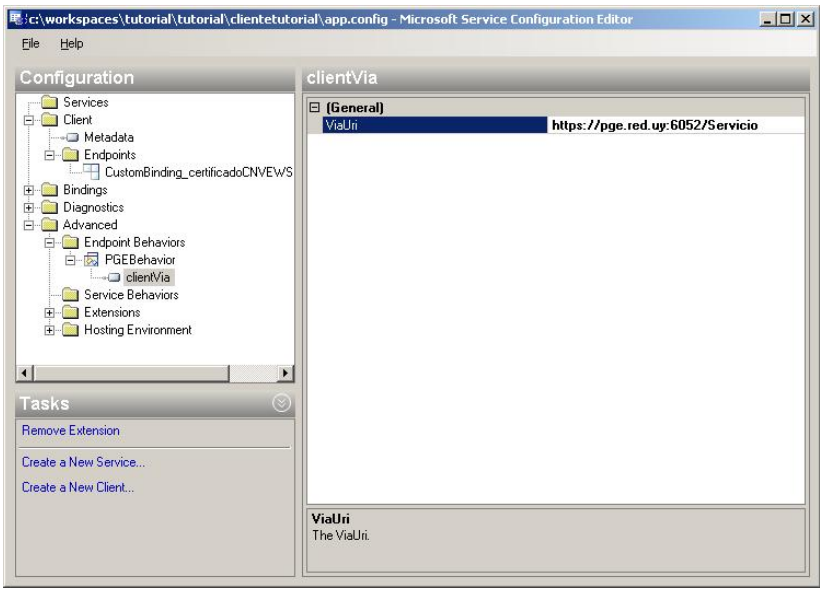
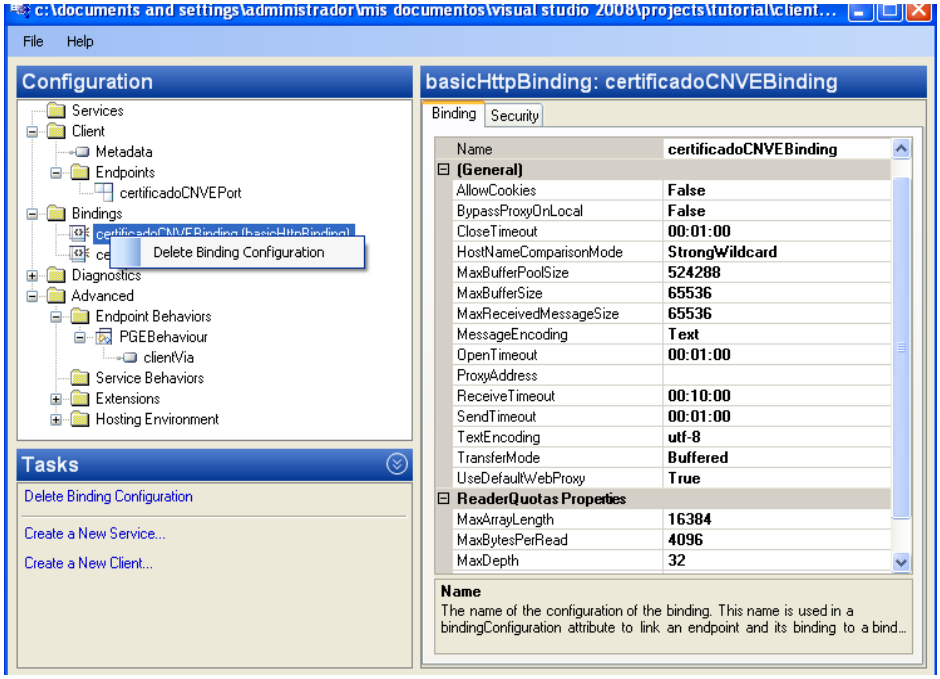


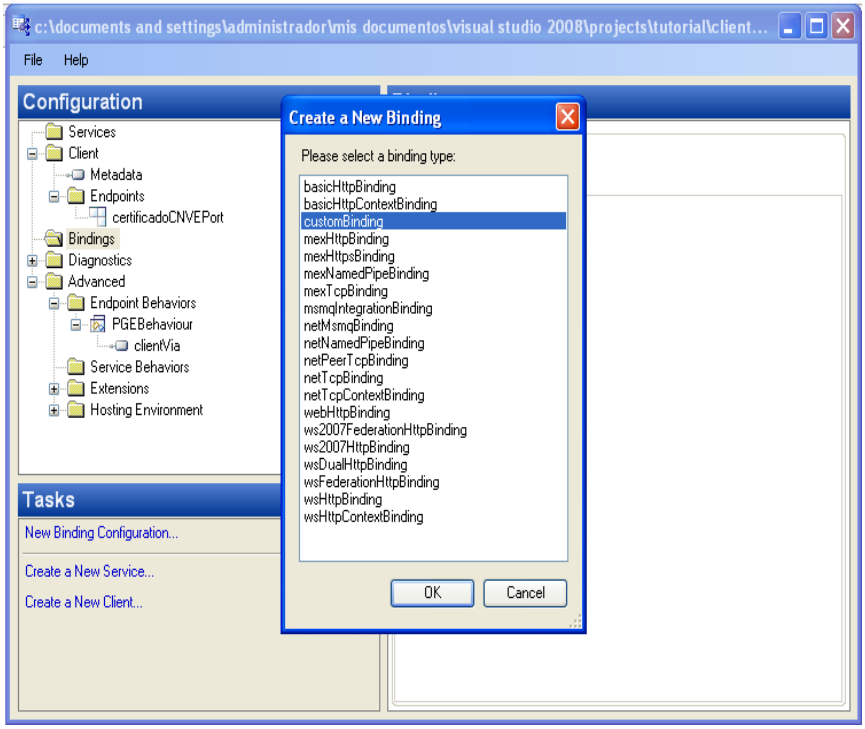
Figura 9: Especificar el valor ViaUri

Configurar la conexión SSL

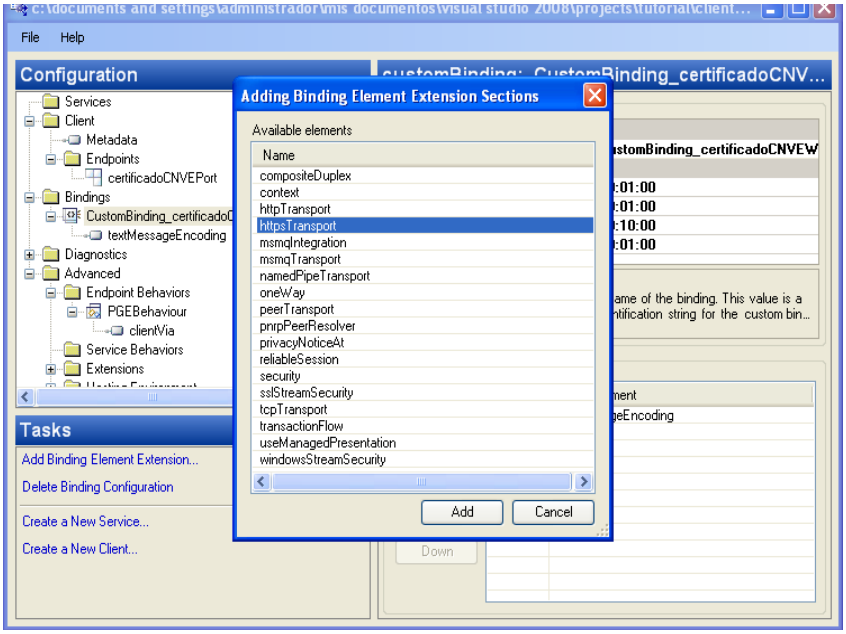
1. En el menu de la izquierda, dentro de Bindings, eliminar cada uno de los bindings que se muestran:



2. Hacer clic derecho en Bindings y seleccionar la opción “New Binding Configuration”. Seleccionar la opción “customBinding” y darle clic en OK.



3. Nombrar el nuevo binding “*CustomBinding_certificadoCNVEWSDLPortType*”
4. *Seleccionar abajo a la derecha el Binding element “httpTransport” y eliminarlo*
5. *Hacer clic en “Add” y seleccionar “httpsTransport”*



6. Dentro de *httpsTransport*, seleccionar como true la opción *RequiredClientCertificate* como *true*, según se muestra en la figura 10.

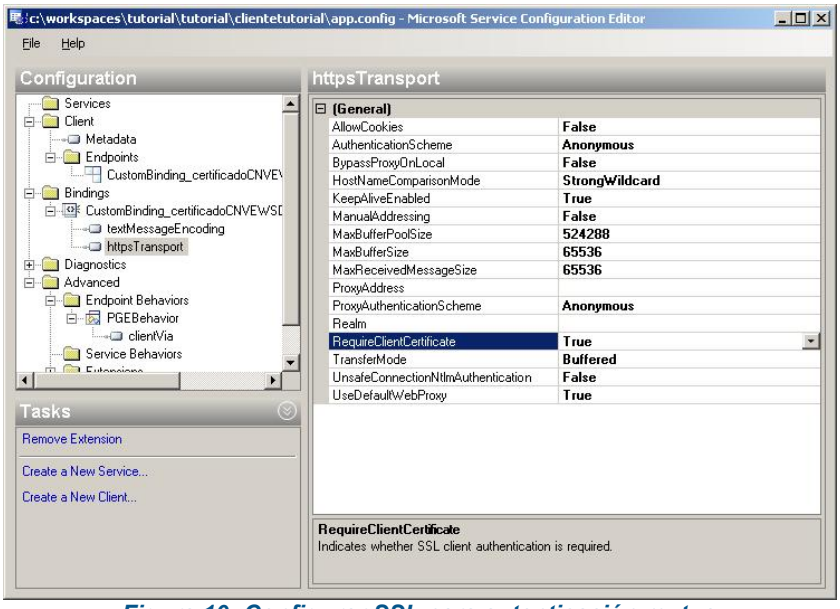


Figura 10: Configurar SSL para autenticación mutua

7. Hacer clic derecho nuevamente en *Bindings* → *CustomBinding_certificadoCNVEWSDLPortType* → *TextMessageEncoding* y verificar que el elemento *MessageVersion* tiene el valor *Soap11WSAddressing10.*

8. Seleccionar *Advanced* → *Endpoint Behaviours* → *PGEBehaviour*, luego clic derecho y la

opción *Add Endpoint Element Extension.*

9. Seleccionar la opción *clientCredentials*

10. Configurar el *clientCredentials* → *ClientCertificate* según muestra la figura 11.

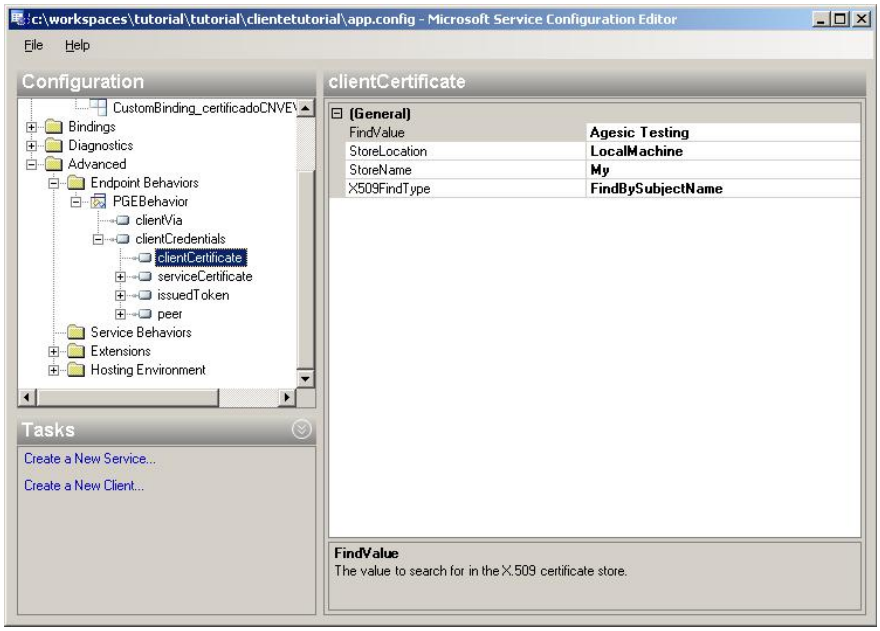


Figura 11: Configurar el certificado del cliente

11. Configurar el clientCredentials → serviceCertificate → Default Certificate según muestra la figura 12.

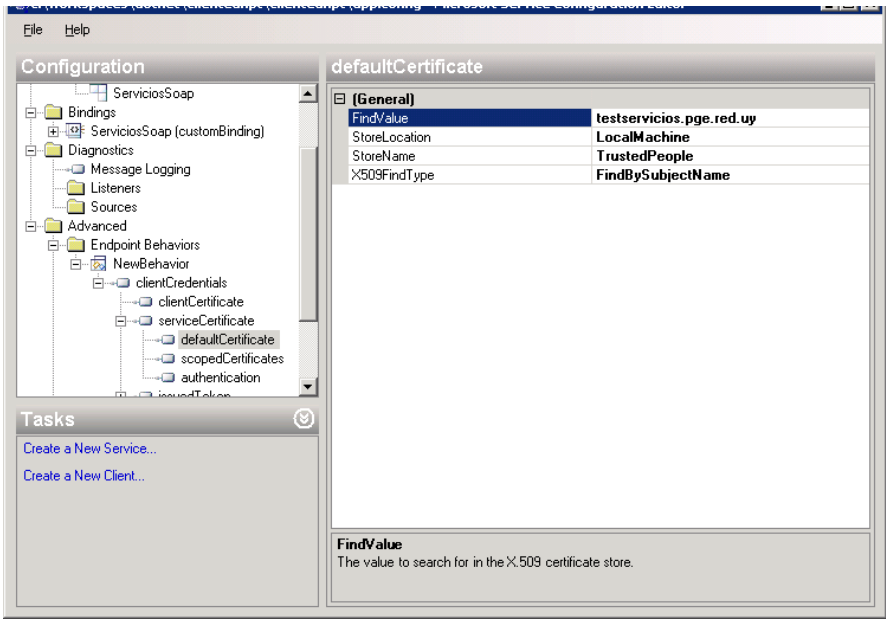


Figura 12: Configurar el certificado de la PGE

12. Configurar el *clientCredentials → serviceCertificate → authentication* según muestra la figura 13.

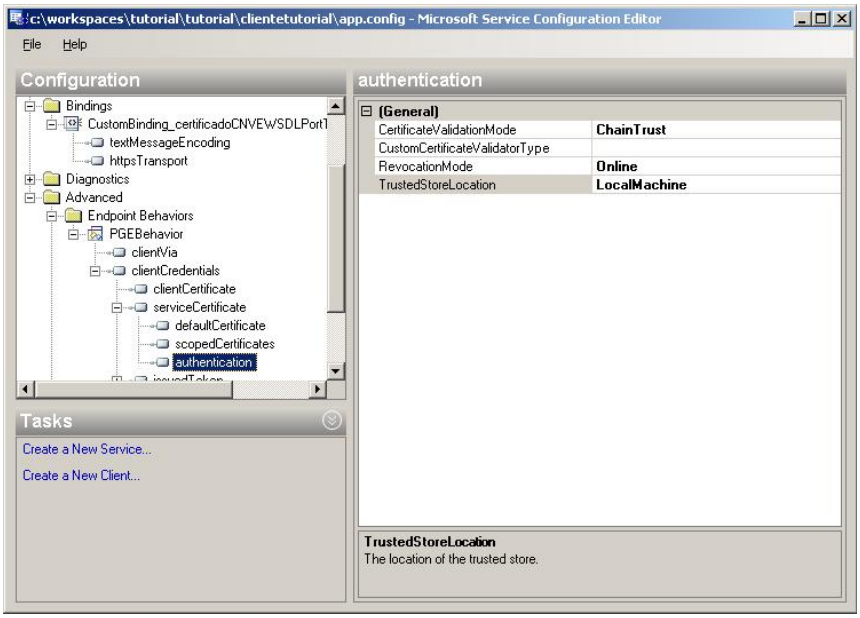
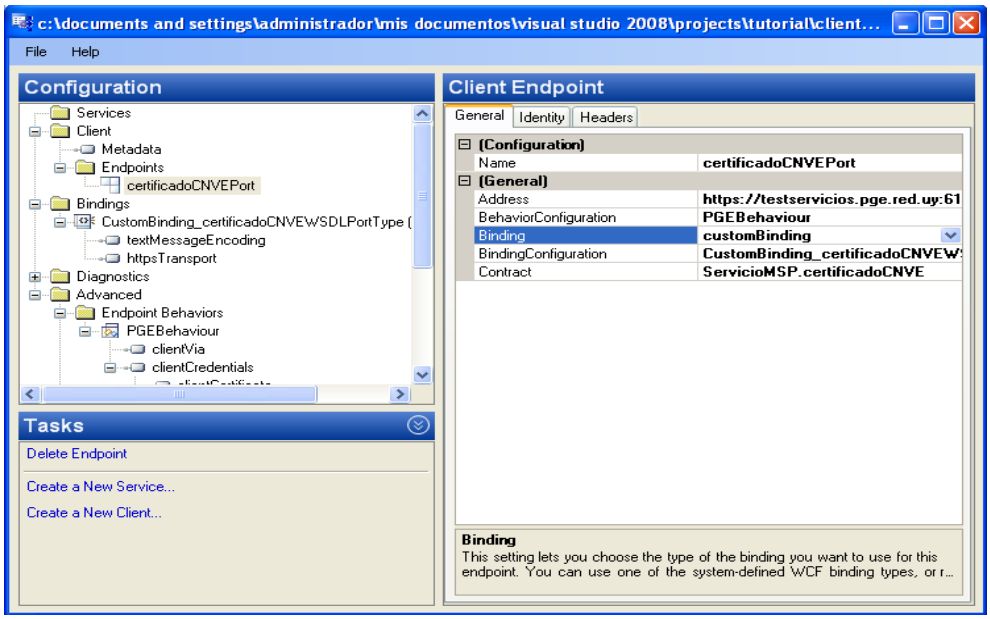


Figura 13: Configurar mecanismo de autenticación de la PGE

13. Asociar el PGE Behaviour creado al cliente. Para ello, seleccionar *Client → Endpoint → CertificadoCNVEPort → BehaviourConfiguration* y seleccionar la opción *PGEBehaviour. A su vez, en la opción "Binding" seleccionar "customBinding" y en "Address" poner "http://testservicios.pge.red.uy/msp/certificadoCNVE"*



14. Seleccionar *File* → *Save* y luego *File* → *Close*.

Configurar comunicación con el STS de la PGE

1. Abrir el archivo app.config y agregar el código en negrita de la figura 14 luego del tag *configuration* y antes del tag *system.serviceModel.*

```
<configuration>
  <configSections>
    <sectionGroup name="pgeConfigSectionGroup">
      <section name="PGEConfigSection" type="AGESIC.PGE.PGEConfigSection, Agesic.PGE,
Culture=neutral, Version=1.0.0.0"/>
    </sectionGroup>
  </configSections>
  <pgeConfigSectionGroup>
    <PGEConfigSection
    SAMLIssuer="AGESIC"
    STSUrl="http://testservicios.pge.red.uy:6001/TrustServer/SecurityTokenService"
    RSTCertificateDN="Agesic Testing"
    STSUrlSSL="https://testservicios.pge.red.uy:6051/TrustServer/SecurityTokenServiceProtected"
    SAMLMaxFrame="15"
    SAMLMinFrame="120"
    RSTCertificateStoreName="My"
    RSTCertificateStoreLocation="LocalMachine" />
  </pgeConfigSectionGroup>
  <system.serviceModel>
  ...
  </system.serviceModel>
</configuration>
```

Figura 14: Configurar opción del STS de la PGE

Con estos agregados, el archivo app.config se debería ver como se muestra a continuación:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
```

```
<configSections>
  <sectionGroup name="pgeConfigSectionGroup">
    <section name="PGEConfigSection" type="AGESIC.PGE.PGEConfigSection, Agesic.PGE, Culture=neutral,
Version=1.0.0.0"/>
  </sectionGroup>
</configSections>
<pgeConfigSectionGroup>
  <PGEConfigSection
  SAMLIssuer="AGESIC"
  STSUrl="http://testservicios.pge.red.uy:6001/TrustServer/SecurityTokenService"
  RSTCertificateDN="Agesic Testing"
  STSUrlSSL="https://testservicios.pge.red.uy:6051/TrustServer/SecurityTokenServiceProtected"
  SAMLMaxFrame="15"
  SAMLMinFrame="120"
  RSTCertificateStoreName="My"
  RSTCertificateStoreLocation="LocalMachine" />
</pgeConfigSectionGroup>
<system.serviceModel>
  <behaviors>
    <endpointBehaviors>
      <behavior name="PGEBehaviour">
        <clientVia viaUri="https://testservicios.pge.red.uy:6187/ws-cnve/certificadoCNVE" />
        <clientCredentials>
          <clientCertificate findValue="Agesic Testing" storeLocation="LocalMachine"
          x509FindType="FindBySubjectName" />
          <serviceCertificate>
            <defaultCertificate findValue="testservicios.pge.red.uy" storeLocation="LocalMachine"
            storeName="TrustedPeople" x509FindType="FindBySubjectName" />
            <authentication trustedStoreLocation="LocalMachine" />
          </serviceCertificate>
        </clientCredentials>
      </behavior>
    </endpointBehaviors>
  </behaviors>
  <bindings>
    <customBinding>
      <binding name="CustomBinding_certificadoCNVEWSDLPortType">
        <textMessageEncoding maxReadPoolSize="64" maxWritePoolSize="16"
        messageVersion="Soap11WSAddressing10" writeEncoding="utf-8">
          <readerQuotas maxDepth="32" maxStringContentLength="8192" maxArrayLength="16384"
          maxBytesPerRead="4096" maxNameTableCharCount="16384" />
        </textMessageEncoding>
        <httpsTransport requireClientCertificate="true" />
      </binding>
    </customBinding>
  </bindings>
  <client>
    <endpoint address="http://testservicios.pge.red.uy/msp/certificadoCNVE" behaviorConfiguration="PGEBehaviour"
    binding="customBinding" bindingConfiguration="CustomBinding_certificadoCNVEWSDLPortType"
    contract="Tutorial.certificadoCNVE"
    name="CustomBinding_certificadoCNVEWSDLPortType" />
  </client>
</system.serviceModel>
</configuration>
```

Figura 15: Archivo app.config

Invocación del Servicio

1. Modificar el código de la clase Program para que quede similar a la figura Error: no se encontró el origen de la referencia.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using AGESIC.PGE;
using ClienteTutorial.Tutorial;

namespace ClienteTutorial
{
    public class Program
    {
        [STAThread]
        static void Main(string[] args)
        {
            PGEContext<certificadoCNVEClient> contexto =
                PGEContext<certificadoCNVEClient>.CreatePGEContext(
                    "Juan",
                    "ou=gerencia de proyectos,o=agesic",
                    "urn:tokensimple");

            getDatosPersonales dp = new getDatosPersonales();
            dp.datosPersonalesInput = new IdentificacionPersonaEntrada();
            dp.datosPersonalesInput.numeroDocumento = "1";
            dp.datosPersonalesInput.tipoDocumento = "1";

            getDatosPersonalesResponse resp = contexto.Client.getDatosPersonales(dp);
            Console.WriteLine("Servicio consumido correctamente");
            Console.ReadLine();
        }
    }
}
```

Figura 16: Crear Cliente PGE

2. Ejecutar el cliente haciendo clic derecho en el proyecto ClienteTutorial → *Debug* → *Start new instance.*

> **Importante:** Antes de correr el ejemplo asegúrese que la hora del servidor se encuentra sincronizada con la hora actual (incluyendo segundos). Si la hora se encuentra adelantada, ocurrirá un error en la ejecución.
>
> Ustede puede sincronizar la hora con el servidor NTP de la Plataforma: ntp.pge.red.uy