PEPPOL
PAN-EUROPEAN PUBLIC PROCUREMENT ONLINE

cip competitiveness and innovation framework programme 2007-2013

Documentation

Project Acronym: PEPPOL
Grant Agreement number: 224974
Project Title: Pan-European Public Procurement Online

PEPPOL Transport Infrastructure

.NET Sample Implementation Developer Documentation, Installation and Configuration Manual

Version: 2.1.3
Status: In Use

Editors:
Oscar Jara and Carlos Quiroz, Difi/Alfa1lab

Project co-funded by the European Commission within the ICT Policy Support Programme		
Dissemination Level		
P	Public	X
C	Confidential, only for members of the consortium and the Commission Services	

Revision History

Version	Date	Editor	Org	Description
2.00	15.11.2011	Oscar Jara / Carlos Quiroz	Alfa1lab	Documentation for the first version of the PEPPOL START Sample Implementation. Versioned 2.00 to match Metro Java versioning.
2.10	13.01.2012	Oscar Jara / Carlos Quiroz	Alfa1lab	New updated release of the PEPPOL START Sample Implementation, see release notes.
2.1.1	01.02.2012	Oscar Jara / Jorge Reátegui	Alfa1lab	Updated SVN repository location for the Sample Implementation source code to reflect new PEPPOL EIA structure.
2.1.2	14.03.2012	Oscar Jara / Carlos Quiroz	Alfa1lab	New updated release of the PEPPOL START Sample Implementation, see release notes.
2.1.3	04.04.2012	Oscar Jara / Carlos Quiroz	Alfa1lab	New updated release of the PEPPOL START Sample Implementation, see release notes.

Statement of originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Statement of copyright

Contributors

Organizations

Difi (Direktoratet for forvaltning og IKT), Norway, www.difi.no
Alfa1lab, Denmark, www.alfa1lab.com

Persons

Oscar Jara, Difi/Alfa1lab
Carlos Quiroz, Difi/Alfa1lab
Jan Victoir, Difi/Alfa1lab
George Reátegui, Difi/Alfa1lab
Kenneth Bengtsson, Difi/Alfa1lab

Table of Contents

1 Release Notes

1.1 Changes

The following changes have been made since version 2.0.0

- Implemented method to send secure message through “UnitTests” project.

The following changes have been made since version 2.1.0

- “PingMessage” method was added at client side.
- Implemented validation for SMP certificate was added in the whole process.
- Implemented application for lookup was added in the project for helping purposes at "resources\utilities\PEPPOL-Lookup" folder.

The following changes have been made since version 2.1.1

- Simplified START client sending mechanism and removed dependencies from START server.
- Simplified structure of properties file.
- Changed service name to "accessPointService" to align with WSDL 2.0 example service name.

The following changes have been made since version 2.1.2

- The redirection, as stated in the SMP has been implemented in the STARTLibrary.
- Validations for certificate UID and restrictions for multiple redirections were added as stated in specifications.
- Participant lookup logic was upgraded with redirection features.

1.2 Bug fixes

The following changes have been made since version 2.1.0

- Implemented PEPPOL standard messages for error handling according the START specifications.

2 Introduction

2.1 Objective and Scope

This document provides all the information to explain in detail the configuration, installation and how to deploy .NET Access Point on an IIS server to demonstrate PEPPOL’s transport infrastructure.

2.2 Audience

The audience for this document is organizations in need for a short introduction to the PEPPOL BIS. These may include the following PEPPOL Stakeholders:

- PEPPOL Community Governance
- Contracting Authorities
- Economic Operators
- ICT Providers
- Service Providers

More specific it is the following roles:

- Business Experts
- ICT Architects
- ICT Developers
- ICT Governing participants

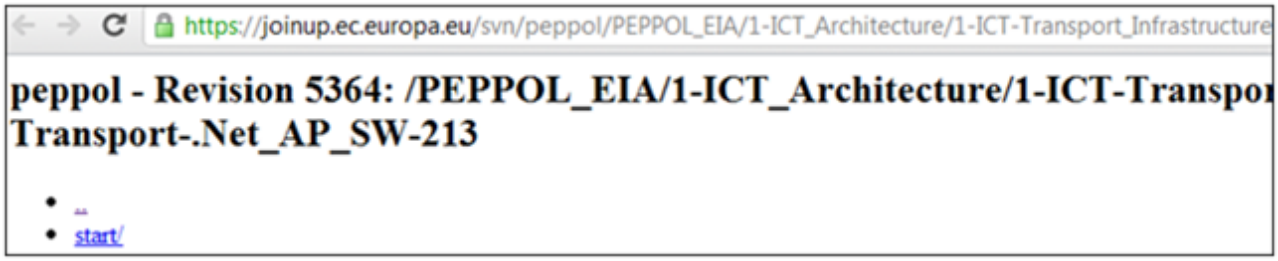
2.3 Developers step by step guide and configurations

2.3.1 Preparing the Environment

This guide is based on Windows 7 (Please, adapt it to other O.S. if needed)
Requirements:
- Microsoft .NET Framework 3.5
- Microsoft Visual Studio 2008 (Team System or other version)
Open <project path> \project\START\START.sln (Project solution)

2.3.2 Source code Download

Download the source code from the PEPPOL svn at:
https://joinup.ec.europa.eu/svn/peppol/PEPPOL_EIA/1-ICT_Architecture/1-ICT-Transport_Infrastructure/14-ICT-Services-Components/ICT-Transport-.Net_AP_SW-213/



2.3.3 Configurations

2.3.3.1 How to create keystores and truststores

The following table provides an overview, where to use what certificates/key pairs for configuring the PEPPOL OSS components:

	PKI Keystore	PKI Truststore	SSL Keystore	SSL Truststore
Access Point	Your AP Key Pair	Root Certificate Generic AP CA Certificate		
Access Point Hosting Server			Any Key Pair	Any matching Certificate
SMP	Your SMP Key Pair	Root Certificate Generic SMP CA Certificate		
SML client	Your SMP Key Pair	Root Certificate Generic SMP CA Certificate		Thawte Root Certificate
SML server*	Your SML Key Pair	Root Certificate Generic SMP CA Certificate	Any valid SSL Key Pair	

When a Key Pair is referenced, it means **Certificate + Private Key** together.

If Certificate is referenced, it means **Public Key** only

* The SML server is centrally operated, so normally you shouldn't care about this

For details on the PEPPOL certificate setup see **this document**.

Alternatively the following table shows which key to use in which component:

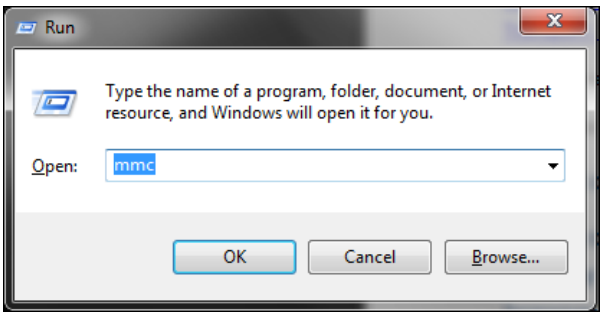
	AP Private Key	AP Certificate	AP CA Certificate	SMP Private Key	SMP Certificate	SML Private Key	SML Certificate	SMP CA Certificate	Root Certificate	SSL Private Key	SSL Certificate
AP Keystore	X	X									
AP Truststore			X						X		
AP WebApp Host										X[1]	X[1]
SMP Keystore				X	X						
SMP Truststore								X	X		
SML Client Keystore				X	X						
SML Client Truststore								X	X		X[2]
SML Server Keystore						X	X			X[2]	
SML Server Truststore								X	X		

2.3.3.2 Certificates configuration in your computer or server store

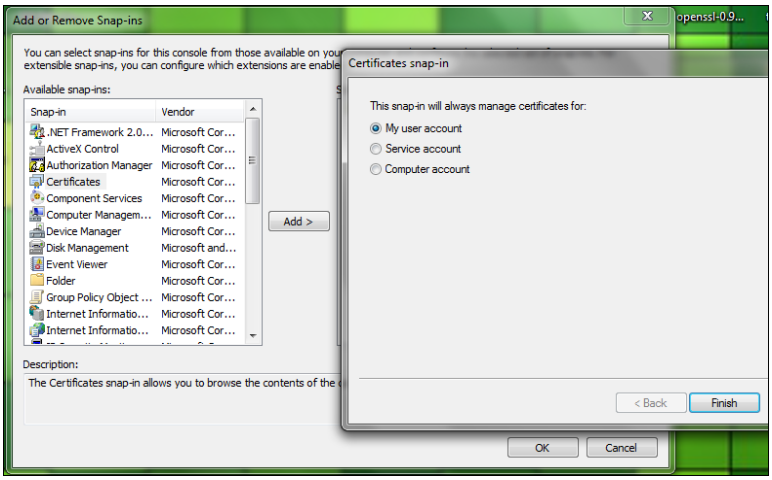
Download PEPPOL pilot CA certificates from the svn:
https://joinup.ec.europa.eu/svn/peppol/PEPPOL_EIA/1-ICT_Architecture/1-ICT-Transport_Infrastructure/16-ICT-Implementations/PKI/PEPPOL%20Pilot%20CA%20certificates.zip

Install them, follow these steps:

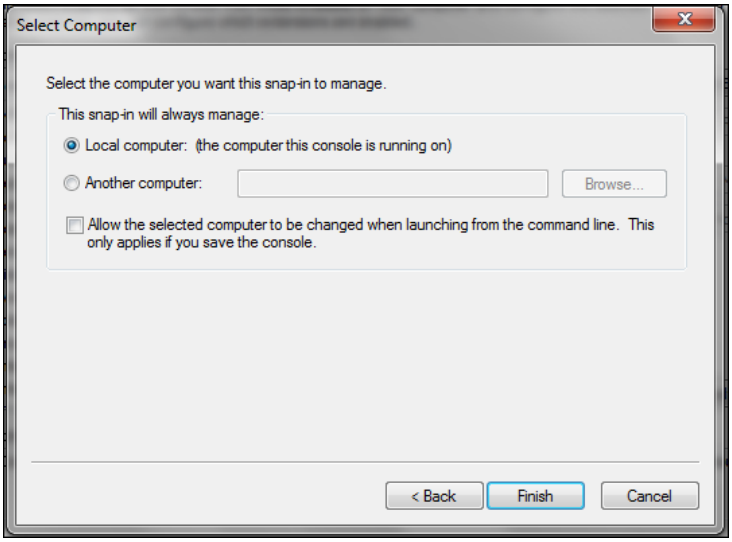
1. Go to run and type “mmc”.



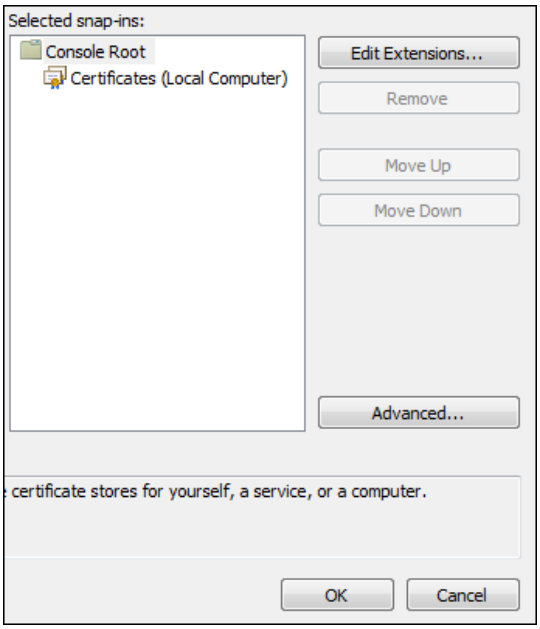
2. Go to File/Add or Remove Snaps-in, select “Certificates”, click on “Add” and a prompt will appear.



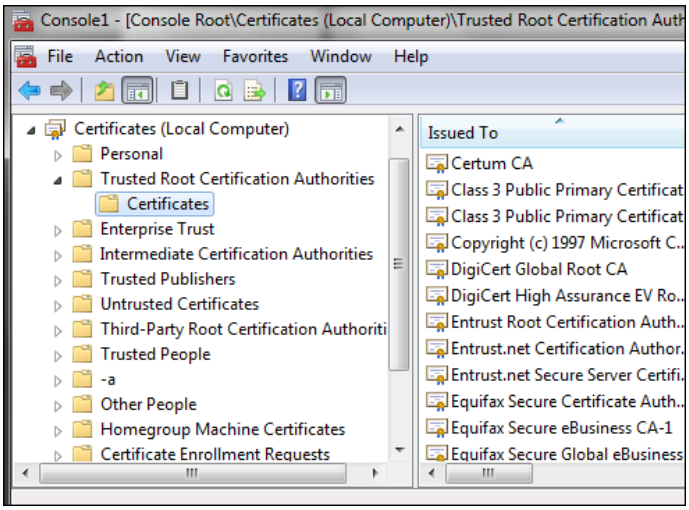
3. Select “Computer account” and go “Next” now select “Local Computer” and click “Finish”.



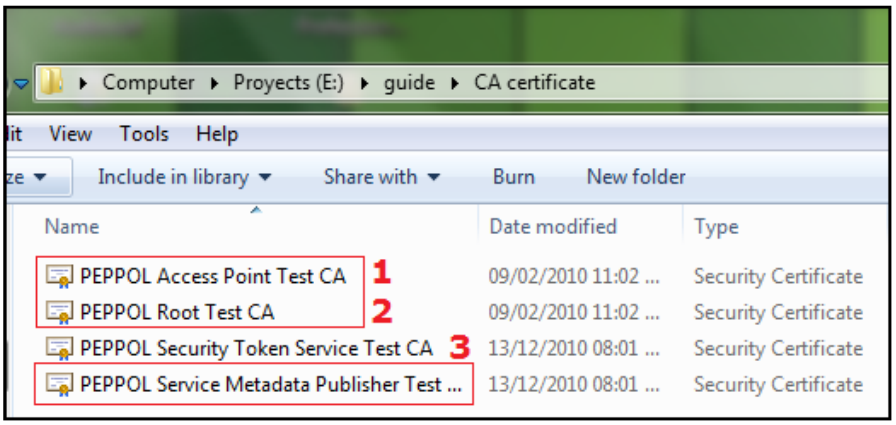
4. Certificates will be added to the snaps-in, now click “Ok”.



5. Go to “Trusted Root Certification Authorities”.

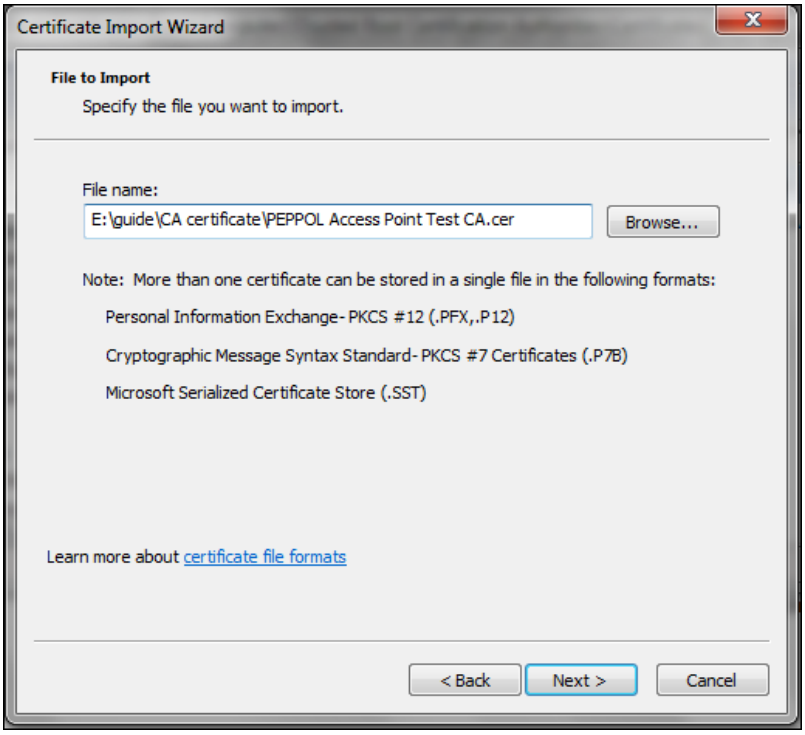


6. In this step, extract the zip file with certificates, you will need only these certificates:

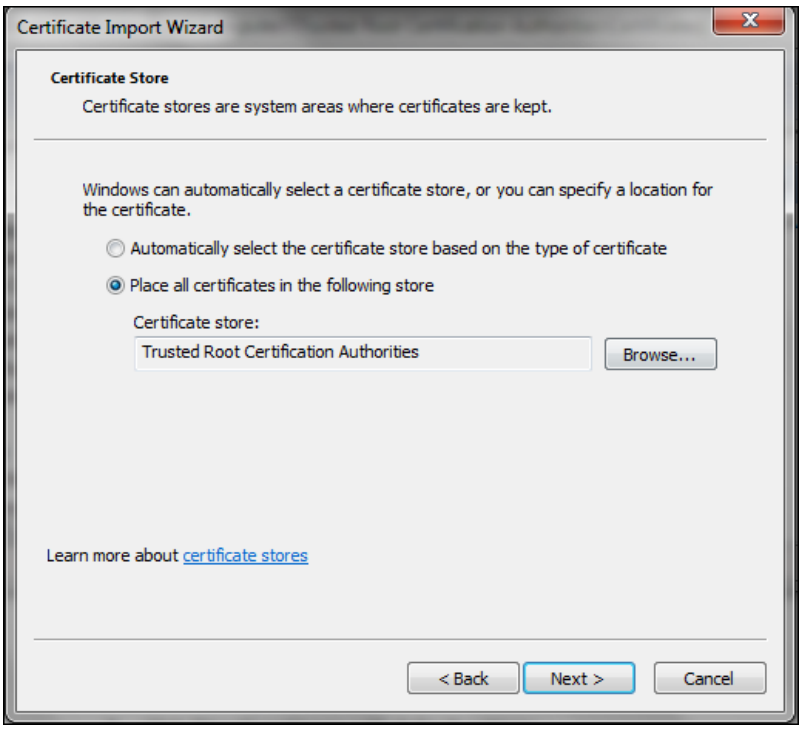


7. Go back to certificates store and right click on “Certificates” go to “All tasks/Import” and a prompt will appear.

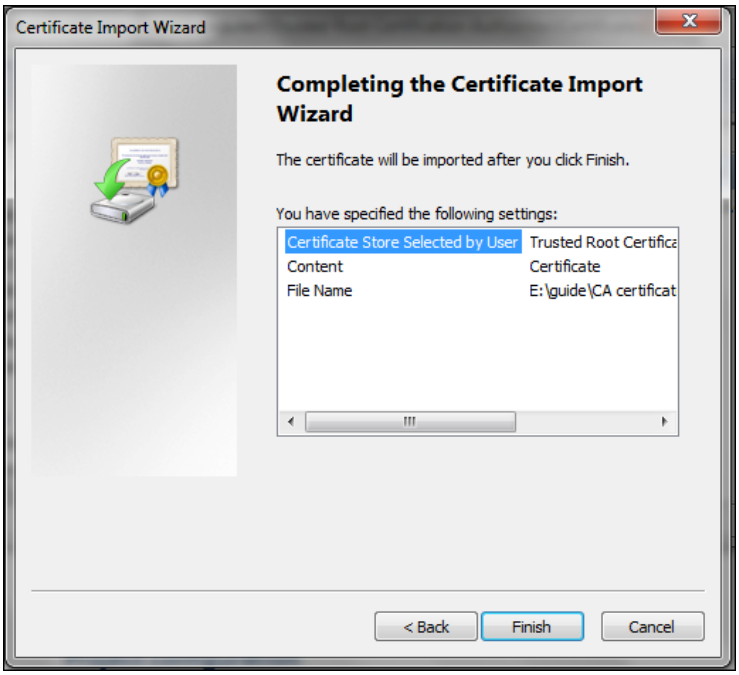
8. Click “Next” and type or browse the path where you put the PEPPOL certificates and select the first certificate.



9. Click “Next” and keep with default options.

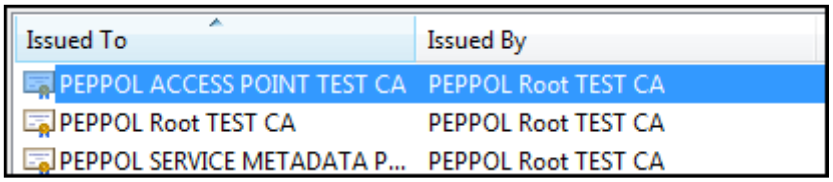


10. Click “Next” again and “Finish”.



11. I f everything is correct, a success message will appear.

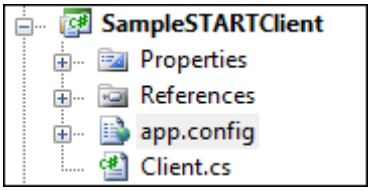
12. Repeat the steps for the next 2 certificates that lasts and finally you will get something like this in your certificate store:



2.3.3.3 Certificates configuration in the project

a) CA Certificates configuration

1. Go to the “SampleSTARTClient” project and open “app.config” file.

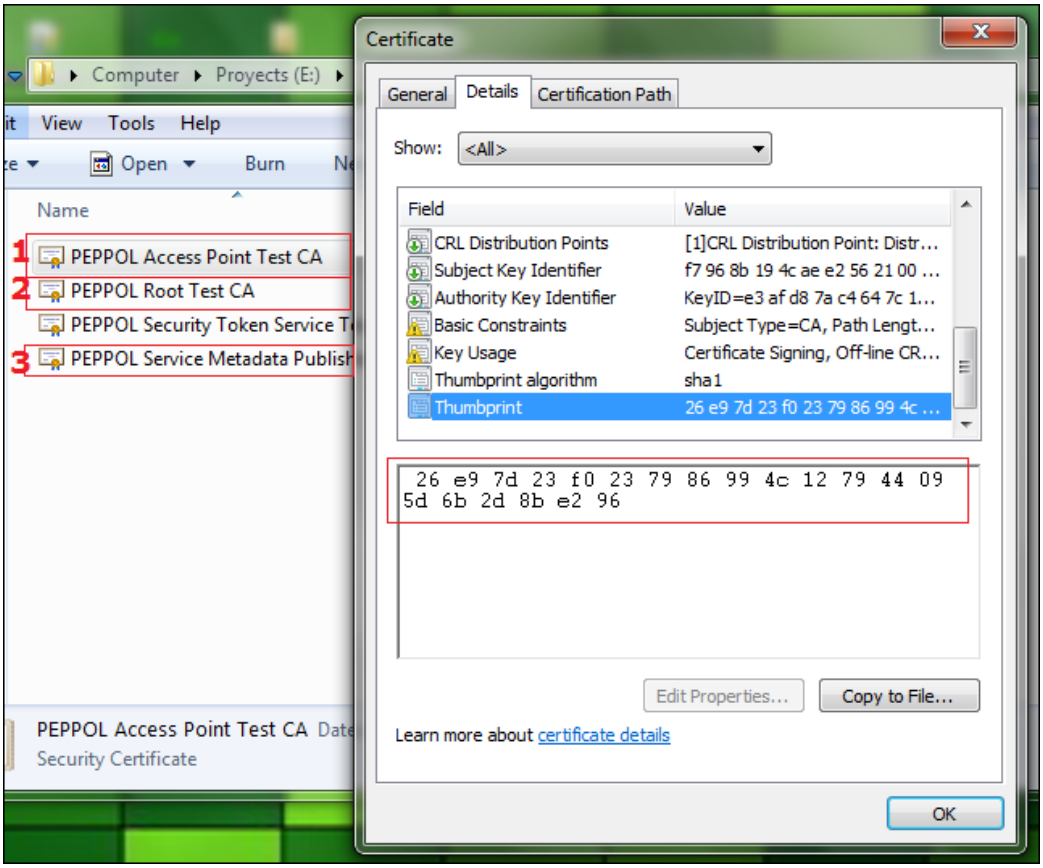


2. Go to lines **92-95** and setup the certificates thumbprints of the 3 certificates you setup in your certificate store.

```
86   <peppol.certificates>
87     <validation>
88       <!-- Thumbprints for:
89             - PEPPOL Root TEST CA
90             - PEPPOL SERVICE METADATA PUBLISHER TEST CA
91             - PEPPOL ACCESS POINT TEST CA -->
92       <add name="MyRootCertificates"
93            rootCACertificateThumbprint=""
94            intermediateSmpCACertificateThumbprint=""
95            intermediateAcessPointCACertificateThumbprint="" />
```

How to obtain certificates thumbprints?

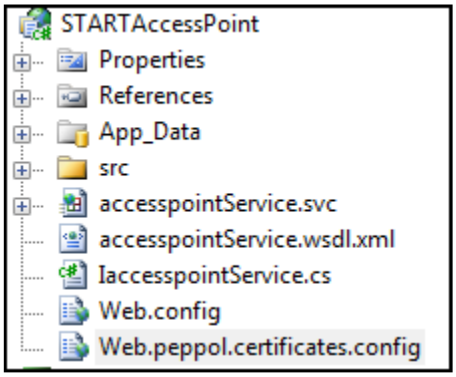
Go to the path you extracted the certificates in the previous part and open one certificate, go to "Details" tab, scroll down and copy "Thumbprint". Repeat this step for the next 2 certificates that lasts. (See image below)



3. Finally you will have something like this:

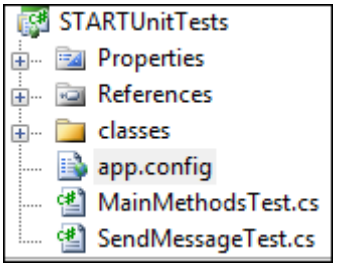
```
<peppol.certificates>
  <validation>
    <!-- Thumbprints for:
          - PEPPOL Root TEST CA
          - PEPPOL SERVICE METADATA PUBLISHER TEST CA
          - PEPPOL ACCESS POINT TEST CA -->
    <add name="MyRootCertificates"
         rootCACertificateThumbprint="55DDC8CE5D35F79271A1DC62830602BAA1B2A0D1"
         intermediateSmpCACertificateThumbprint="D2DDCB2F24B6B840005525346C23197EB32466F0"
         intermediateAcessPointCACertificateThumbprint="26E97D23F0237986994C127944095D6B2D8BE296" />
```

4. Configure the same parameters in the “STARTAccessPoint” project. Open “Web.peppol.certificates.config” file:



5. Go to lines **8-11** and copy/paste the thumbprints of the certificates again.

6. Configure the same parameters in the “STARTUnitTests” project. Open “app.config” file:



7. Go to lines **88-91** and copy/paste the thumbprints of the certificates again.

b) Client and Service Certificates configuration

We just can give you some information regarding how to create keystore (Client certificate) and truststore (Service certificate) certificate files. Please contact a PEPPOL representative to get more information and a certificate key according to you.

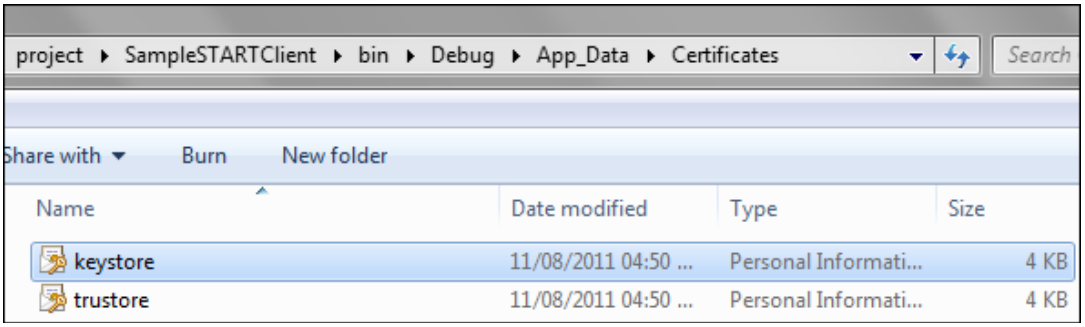
- The keystore for .NET contains two certificates (START-AP and Intermediate certificates) and one private key.
- The truststore for .NET contains two certificates (START-AP and Root certificates) and one private key.

- The certificates must be in ".pfx" format for .NET AccessPoint project.

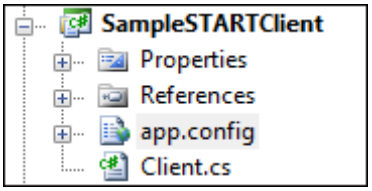
1. Once you have your "keystore.pfx" and "truststore.pfx" files go to:

- **"<project path>\project\SampleSTARTClient\bin\Debug\App_Data\Certificates"**
- **"<project path>\START\STARTAccessPoint\App_Data\Certificates"**

And copy your "keystore.pfx" and "truststore.pfx" files, e.g.



2. Go to "SampleSTARTClient" project and open "app.config" file.



3. Put your values in lines **106, 117, 122, 123, 139, 141** according to your keystore and truststore. The filename path must be:

"App_Data\Certificates\yourcertificate.pfx".

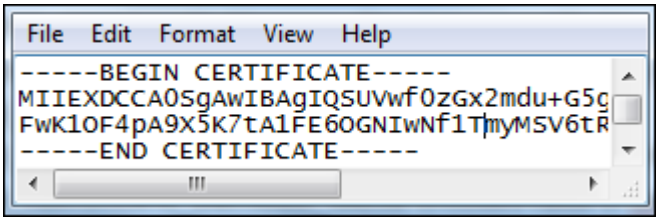
```
106        <serviceCertificate filename="App_Data\Certificates\truststore.pfx" password="yourpassword" />
```

```
117        <serviceCertificate filename="App_Data\Certificates\truststore.pfx" password="yourpassword" />
```

```
122        <clientCertificate filename="App_Data\Certificates\keystore.pfx" password="yourpassword" />
```

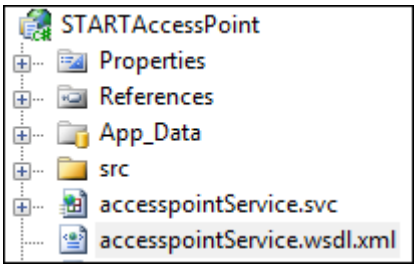
```
123            <serviceCertificate encoded="" />
```

Encoded value of "keystore.pfx" certificate. For getting encoded value, drag "keystore.pfx" file to notepad, copy text between *---BEGIN CERTIFICATE---* and *---END CERTIFICATE---* and copy/paste that value.



How to put the encoded value into the WSDL file?

This value must be included in the wsdl file, go to "STARTAccessPoint" project and open "accesspointService.wsdl.xml" (The explanation about using an static XML will be later)

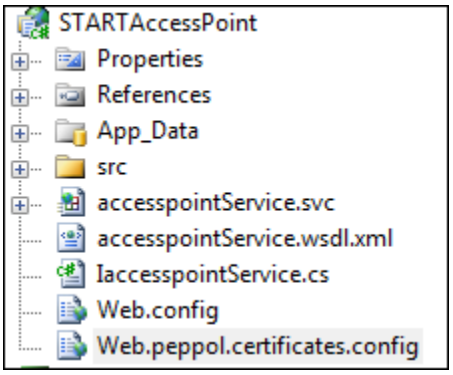


Press CTRL+F and find "<X509Certificate>" element and put there your encoded value.

```
<X509Data>
  <X509Certificate>ENCODED_VALUE_GOES_HERE</X509Certificate>
</X509Data>
```

```
139          <clientCertificate filename="App_Data\Certificates\keystore.pfx" password="yourpassword" />
141          <serviceCertificate filename="App_Data\Certificates\truststore.pfx" password="yourpassword" />
```

4. Go to "STARTAccesspoint" project and open "Web.peppol.certificates.config" file.

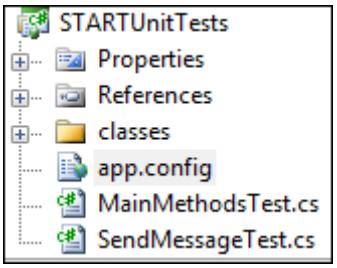


5. Put your values in lines **23, 36, 44, 47, 64, 68** according to your keystore and truststore. **The filename path must be "App_Data\Certificates\yourcertificate.pfx".**

6. Now open "Web.config" and put your encoded value from your "keystore.pfx" at line **197**

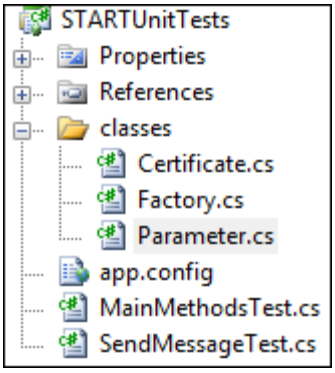
```
196          <identity>
197            <certificate encodedValue="" />
198          </identity>
```

7. Go to “STARTUnitTests” project and open “app.config” file:



8. Put your values in lines **102, 113, 118, 119, 135, 137** according to your keystore and truststore. **The filename path must be “App_Data\Certificates\yourcertificate.pfx”. In this case, we are just specifiying the certificates according to configuration files from other projects, this file must be configured the same just to make tests because is an imitation of the others.**

9. However, it is necessary to specify your credentials (keystore and truststore) due to the test for sending a secure message. To do this, go to “STARTUnitTests” project “classes” folder and open “Parameter.cs” class.



10. Change the “Certificates config.” Parameters according to your keystore and truststore. To do this, you need to put your certificates on a separate path of your computer or server because this is a “Test project” and if we do the same for getting the certificates as the other configuration files (for example: obtaining the certificates from “<projectpath>\project\SampleSTARTClient\bin\Debug\App_Data\Certificates”) you will get an error due to the “temp” folder that this project makes each time you execute a test. This makes impossible to get the certificate from a path in the project. *(Reference: change parameters)*

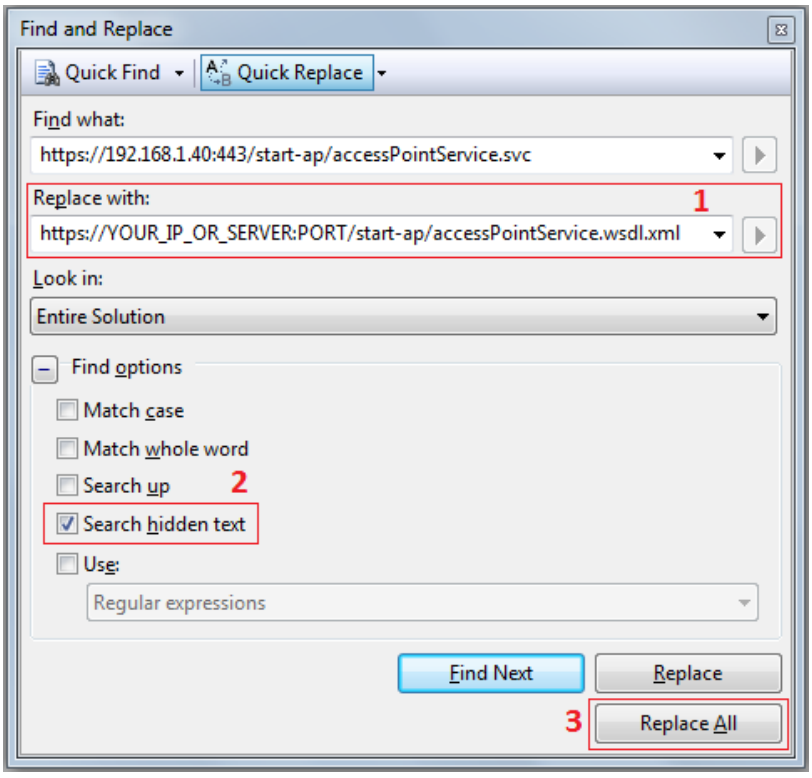
```
26          //Certificates config.
27          public static string clientCertificatePath = @"E:\Certificates\keystore.pfx";
28          public static string clientCertificatePwd = "yourpassword";
29          public static string serviceCertificatePath = @"E:\Certificates\truststore.pfx";
30          public static string serviceCertificatePwd = "yourpassword";
```

11. Changing configuration parameters to make them work

- The important thing here is to change all parameters according to your computer or server.

- To be more specific, the project has this URL for the web service: https://192.168.1.40:443/start-ap/accessPointService.svc and you need to change it according to your computer or server web service URL.

11.1 Press CTRL+F and the "Search" prompt will appear, go to "Quick Replace" and put the options as the image. **Don't forget to "Look in – Entire Solution" and specify the port, also if it is "443" by default.**



11.2 Everything will be replaced, save all and rebuild solution.

11.3 There is one more value to replace, go to "Web.config" file from "STARTAccessPoint" project (line **207**) do it manually according to your web service URL that will reference the wsdl as "externalMetadataLocation" (accesspointService.wsdl.xml file).

```
207  httpsGetEnabled="true" externalMetadataLocation="https://192.168.1.40:443/start-ap/accesspointService.wsdl.xml"
```

https://YOUR_IP_OR_SERVER:PORT/start-ap/accesspointService.wsdl.xml

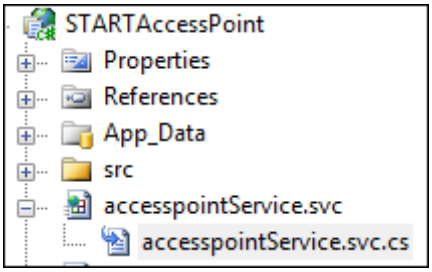
12. **Considerations when using an IP or server domain for the service address/url:**

- When using WCF, it finds by default the computer name or server name of the machine. That's why we use a class for replacing the computer or server name in case you want to use IP address in the URL.

  If you will use an IP replace the values to the ones you will use (see image below as reference) and skip next steps. If you will use a server, follow all steps.

```
293             /* Change according to your IP or Server:                          */
294             /* Remove this when using server.                                  */
295             /* (This method is used when trying to change the host of wsdl to IP */
296             HttpToHttpsWsdlBehavior.AdjustWsdlOutput(serviceHost,
297                 "https://servername/start-ap/accesspointService.svc",
298                 "https://your_ip:443/start-ap/accesspointService.svc");
```

12.1 Go to "STARTAccessPoint" project and open "accessPointService.svc.cs" class.



12.2 Comment lines **296 – 298** like this:

```
293             /* Change according to your IP or Server:                          */
294             /* Remove this when using server.                                  */
295             /* (This method is used when trying to change the host of wsdl to IP for testing */
296             /*HttpToHttpsWsdlBehavior.AdjustWsdlOutput(serviceHost,
297                 "https://pc-net00/start-ap/accesspointService.svc",
298                 "https://192.168.1.40:443/start-ap/accesspointService.svc");*/
```

12.3 Uncomment line **93** and comment line **94** like this:

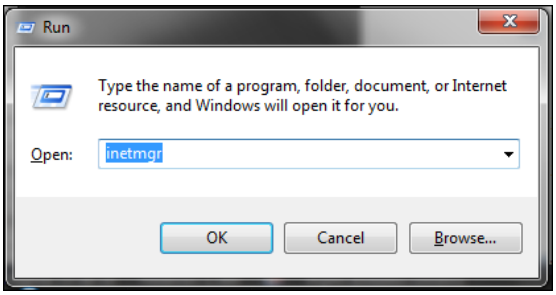
```
92  /* Only for servers */
93  string myEndPointAddress = OperationContext.Current.EndpointDispatcher.EndpointAddress.Uri.AbsoluteUri;
94  //string myEndPointAddress = ConfigurationManager.AppSettings["myEndPointAddress"];*/
```

12.4 If you change in future your AP address to an IP, do the reverse of these steps.

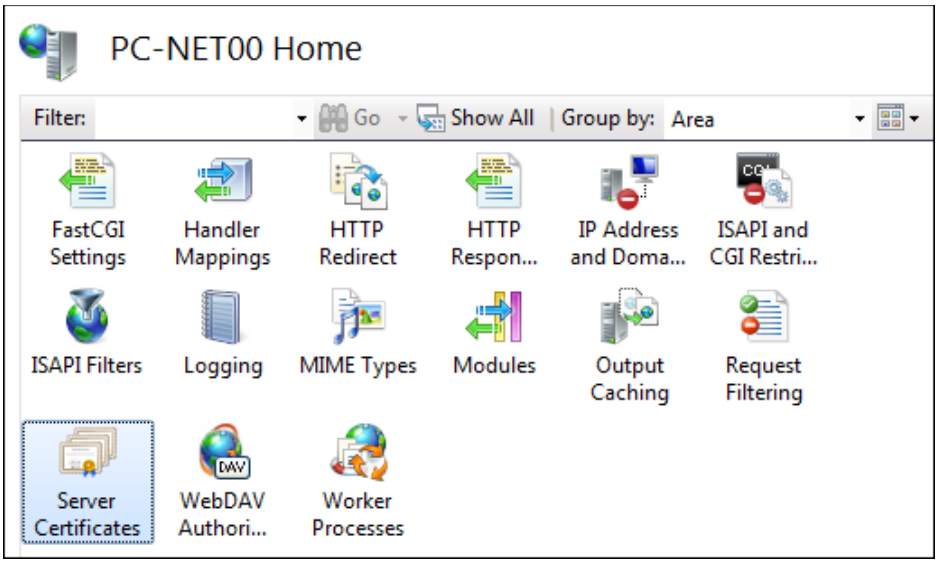
2.3.4 Deployment

To deploy the project follow the next steps:

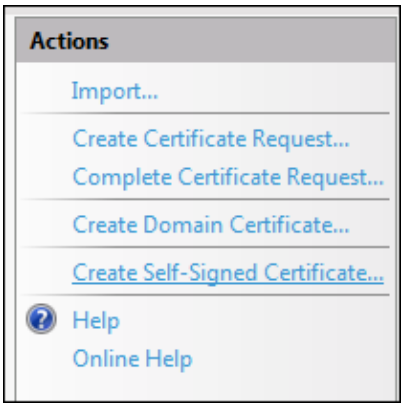
1. Rebuild everything and go to run and type “inetmgr”.



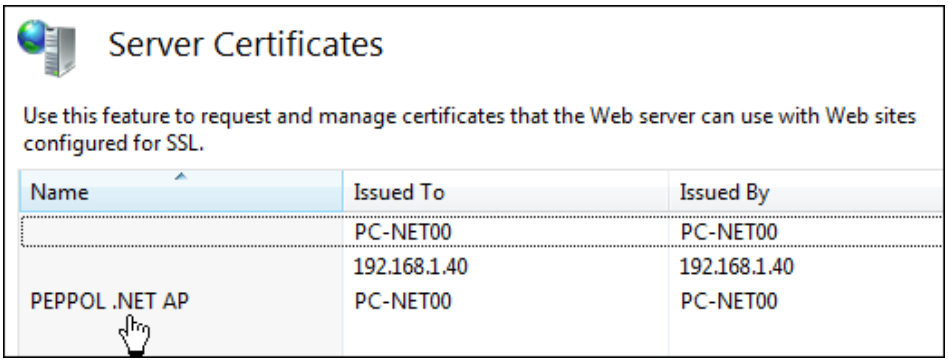
2. Go to “Server Certificates”



- In the right part, a menu will appear with actions, click on “Create Self-Signed Certificate”.

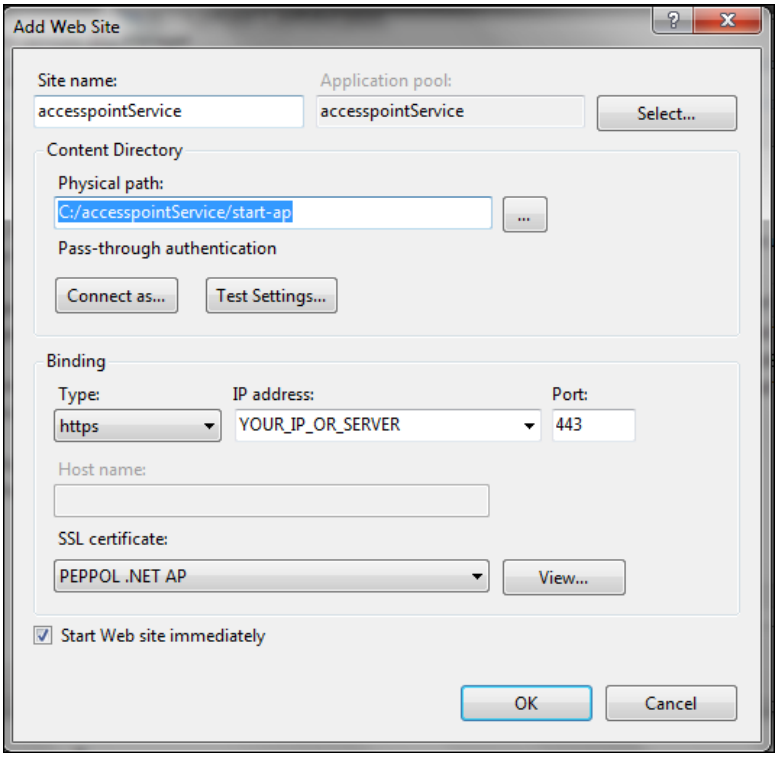


- Type a “Friendly Name” for example “PEPPOL .NET AP” and you will get the following:

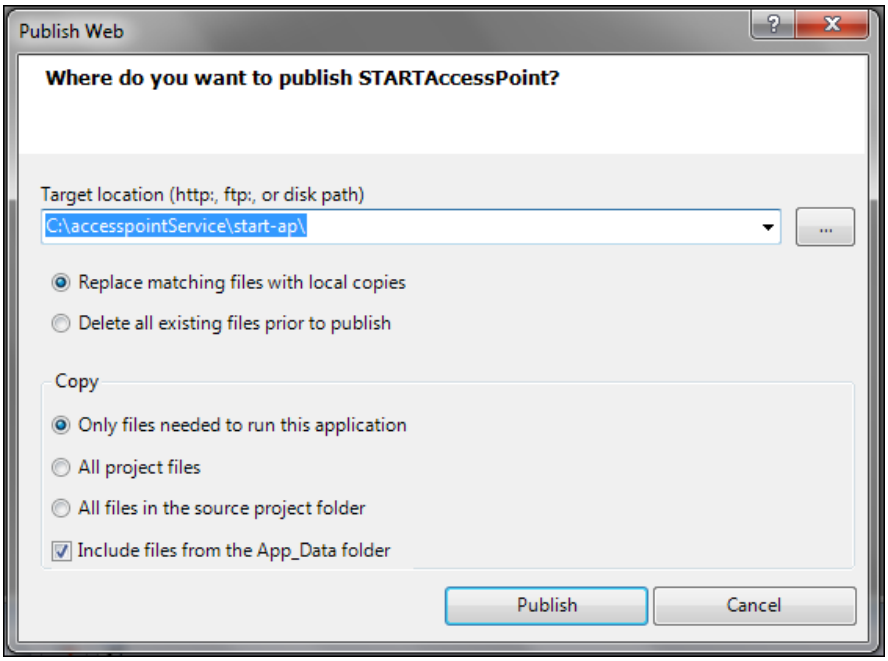


- Create a path to publish website, example “C:/accesspointService/start-ap”

- Add new website with these configurations:

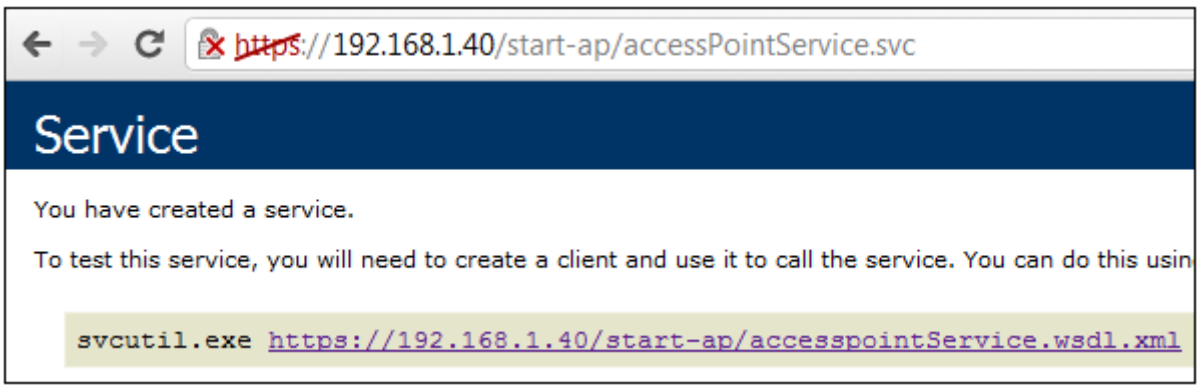


- Click “Ok” and go back to the project in Visual Studio, right click on “STARTAccessPoint” project and select “Publish” and a prompt like this will appear:



- Click on publish and make sure “**accesspointService.wsdl.xml**” is in the same place as “accessPointService.svc” and the folders “**bin/App_Data/Certificates**” with your keystore and truststore exists in the folder where you publish the AP.

- Type your url through the browser and you will get something like this:



- As you can see, the SSL certificate is not trusted, because your selfsigned was issued by your “COMPUTER-NAME” and issued to your “COMPUTER-NAME” but not for your IP address (this does not happen on servers). To make it trusted, follow next steps.

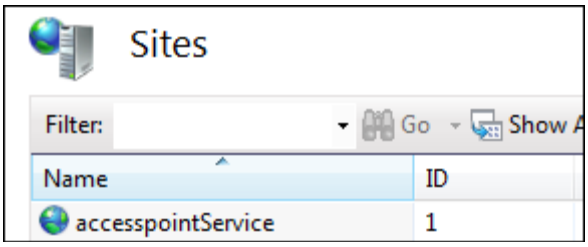
- Note: It is not recommended to assign a certificate to an IP but we are not in production stage so we can use it by now.

- Skip steps ”a” to ”c” if you will use your server or computer name instead of an IP.

a. Get SelfSSL, download IIS 6.0 Resource Kit Tools from http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=17275 or find it as an independent executable.

b. Open SelfSSL console and type the following command:
SelfSSL /T /N:CN=”[HOST_NAME]” /V:1000 /S:[SITE_ID]

c. Replace “[HOST_NAME] ” for your IP or server name and the “[SITE_ID]” from here:



- Go to server certificates section on IIS, right click and export the certificate you want. A file will be generated in the path you specified in the process, now go to the “Trusted Root Certification Authorities” at certificate store (Local machine) as was explained in steps before and import your selfsigned certificate.

- To get more information about doing all process with details go to this link: http://www.robbagby.com/iis/self-signed-certificates-on-iis-7-the-easy-way-and-the-most-effective-way/

2.3.5 Common Issues

2.3.5.1 Error when compiling or trying to run .NET Access Point

This happens because your computer or server does not have included the necessary assemblies in the environment. Please read “a” and “b” cases.

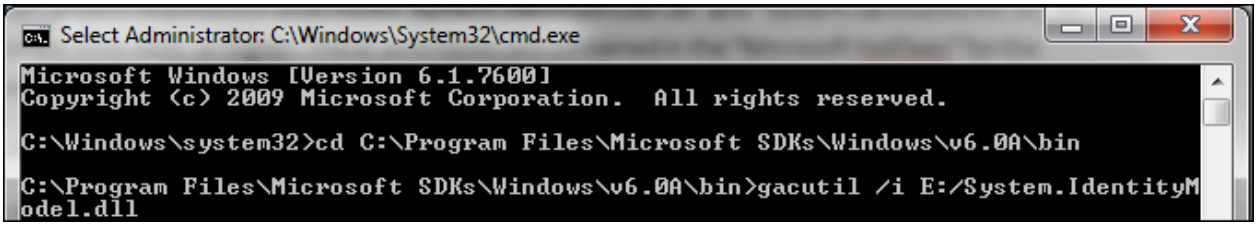
You can find them on the PEPPOL svn and in the project. Just go to “../resources/assemblies” path to get them.

a. The start project requires the assemblies "System.IdentityModel.dll" and "System.ServiceModel.dll" in version 3.0.4506.4446 or higher. These assemblies are contained in the “Microsoft hotfixes” for the various OS listed at http://support.microsoft.com/kb/974842/

It is strongly recommended to install the hotfix and when installing it you do not have to install the assemblies manually.

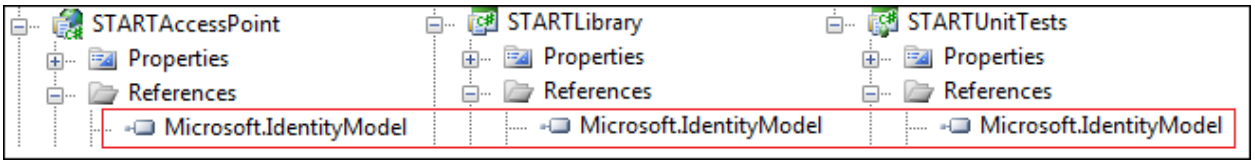
But if you decide to install assemblies manually, do the following:

1. Open your command prompt (cmd.exe) as administrator.
2. Type "cd C:\Program Files\Microsoft SDKs\Windows\v6.0A\Bin"
3. Now type "gacutil /i E:/System.IdentityModel.dll" (/i <assembly path>)
4. Do the same to register the "System.ServiceModel.dll"



b. There is other assembly needed to run the project "Microsoft.IdentityModel.dll".

This assembly is used for SAML and it is necessary to reference it to the projects references that need it. You can look if the project needs or not this assembly if you look an exclamation mark at this part.



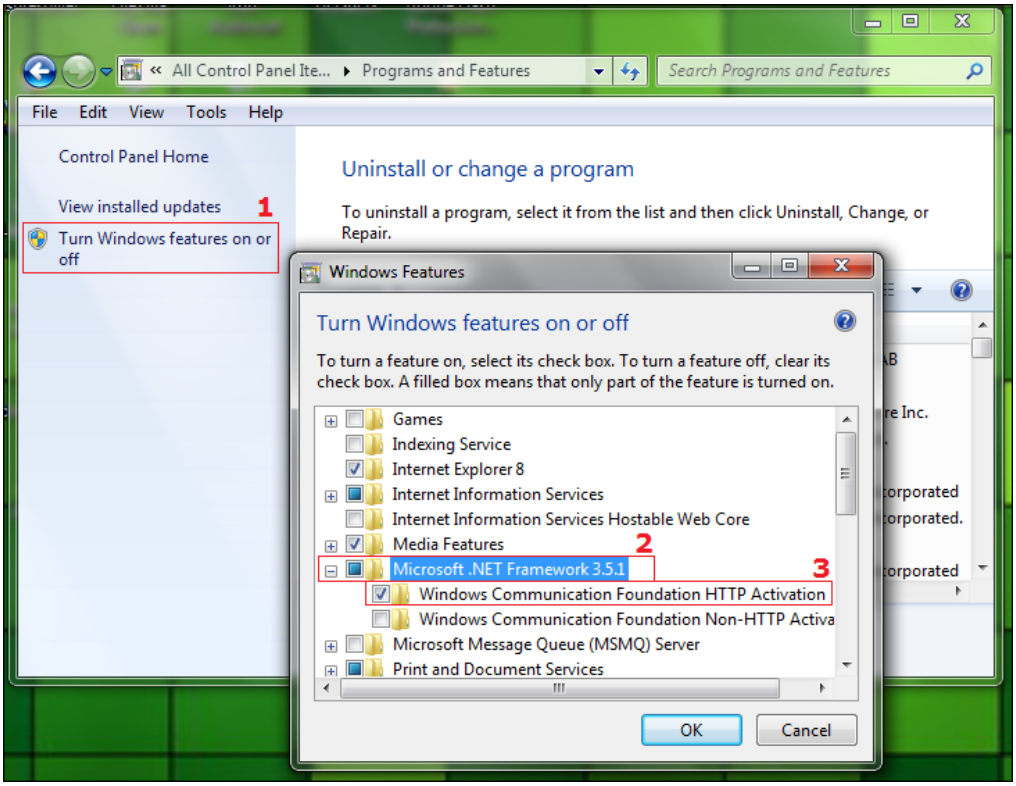
If you look an exclamation mark in the project references is because your computer or server does not have the "Windows Identity Foundation" but for this project you just need "Microsoft.IdentityModel" assembly (You can find it on "../resources/assemblies" path)

* If you want to download the "Windows Identity Foundation" go to:
http://www.microsoft.com/download/en/details.aspx?id=17331

2.3.5.2 Error opening WCF (.svc) extension in browser

Sometimes, your computer or server is not able to run WCF services and you first need to ensure if it is able to run this kind of service.

1. Go to Control Panel /Programs and Features.
2. Click on "Turn Windows features on or off".
3. Select "Microsoft .NET Framework 3.5.1" and look if "Windows Communication Foundation HTTP Activation" is checked.
4. If is not checked, enable it and click "Ok" to enable your computer or server to use WCF services.



2.3.6 Common Questions

2.3.6.1 Explanation of why WSDL is an static xml file

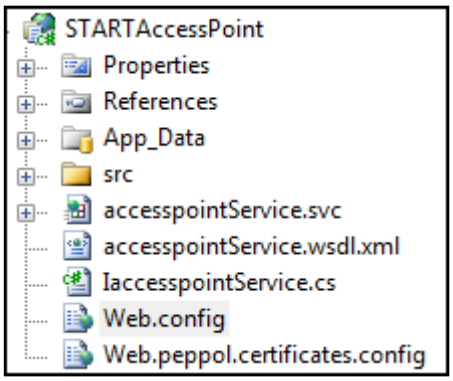
Because the "httpsTransport" element in "Web.config" from “STARTAccessPoint” project made the WSDL for that binding to require both “assymmetricBinding” and “transportBinding” assertions to be generated and this is not supported by WS-SecurityPolicy. This is the reason you need to specify an "externalMetadataLocation" and if you remove it, you will get the following error when browsing your service but it will also works.

“Security policy export failed. The binding contains both a SymmetricSecurityBindingElement and a secure transport binding element. Policy export for such a binding is not supported. System.InvalidOperationException: Security policy export failed. The binding contains both a SymmetricSecurityBindingElement and a secure transport binding element. Policy export for such a binding is not supported.”

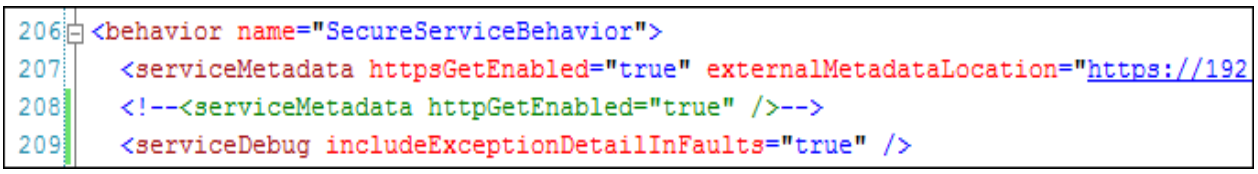
2.3.6.2 Explanation of how to get original WSDL file

If you want to retrieve the original WSDL from the service, do the following steps:

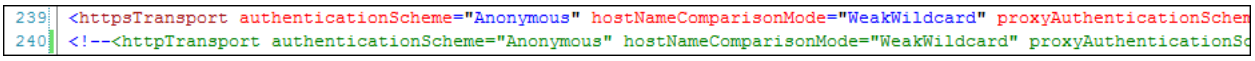
1. Go to “STARTAccessPoint” project and open “Web.config” file.



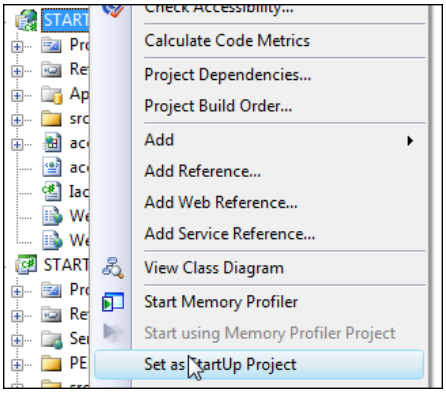
2. Go to line **208** and uncomment it.



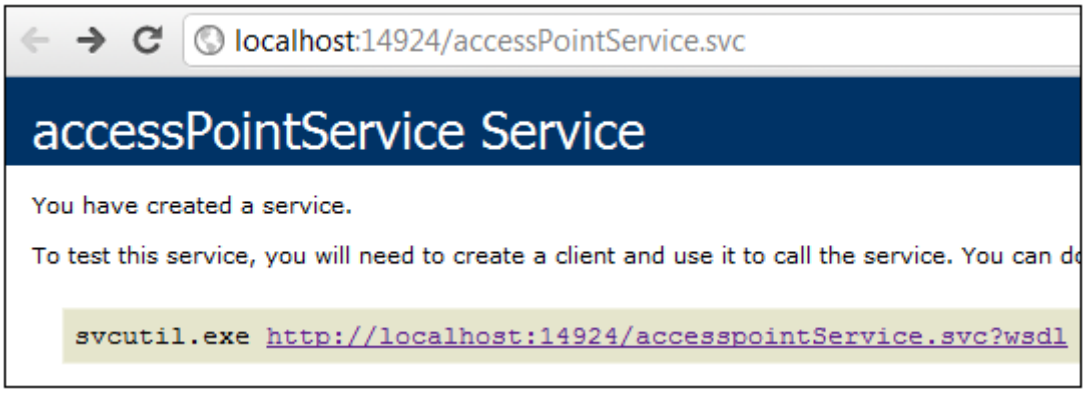
3. Now comment line 207.
4. Go to line 240 and uncomment it.



5. Now comment line 239.
6. Set your project as the main project, right click on the project name and click on “Set as StartUp Project”.



7. The name of the project now will be in bold, and you can press F5 or play button in the toolbar to run the .NET Access Point service. The browser will be opened and now the url for the wsdl is another one and you are running the service through “secure” http at localhost.



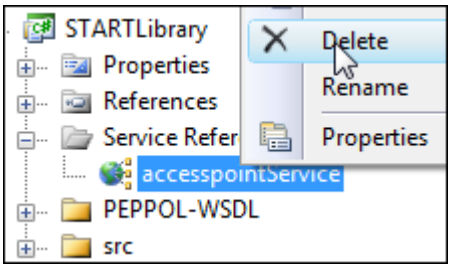
8. To go back to “https” comment and uncomment lines in steps before and make the “SampleSTARTClient” the “StartUp” project.

2.3.6.3 Explanation of how to update PEPPOL WSDL file

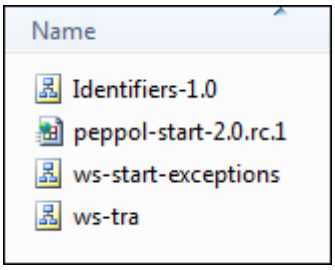
If a new version is released:

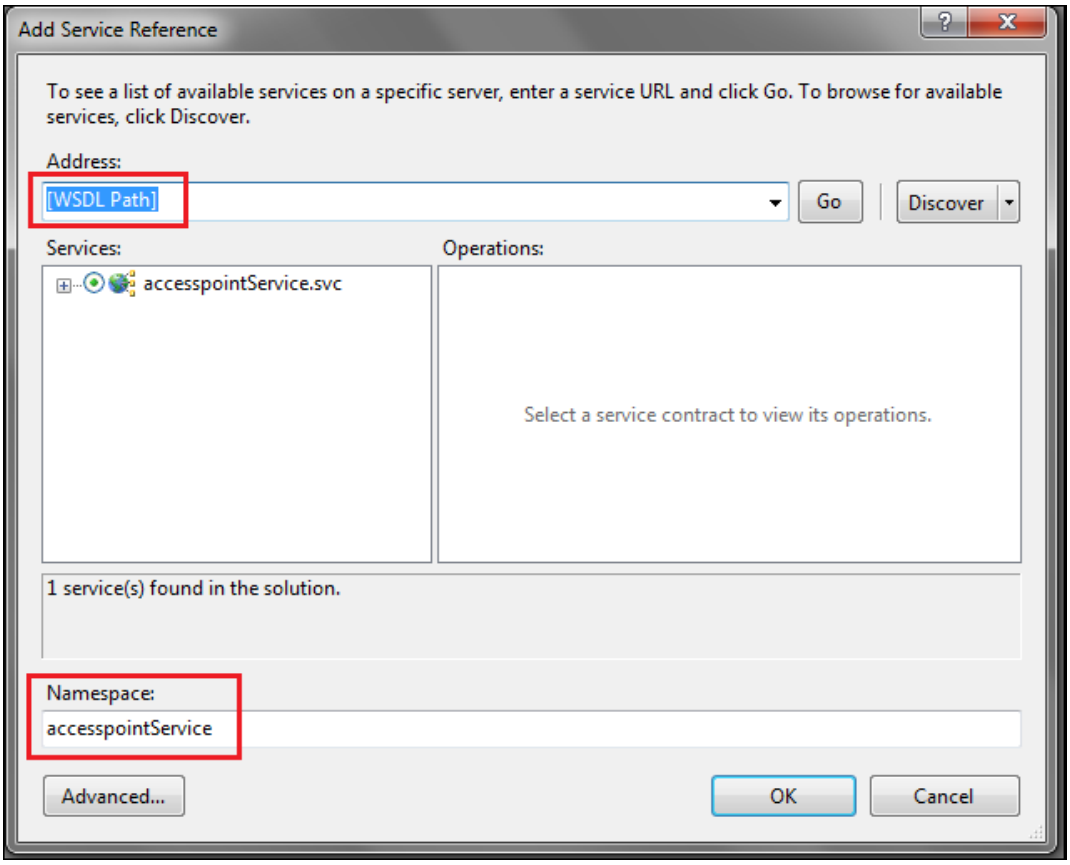
- Original files of PEPPOL WSDL file for this project are situated at: **<project path> \project\START\STARTLibrary\PEPPOL-WSDL**

- Go to “STARTLibrary” project and delete the previous service reference.

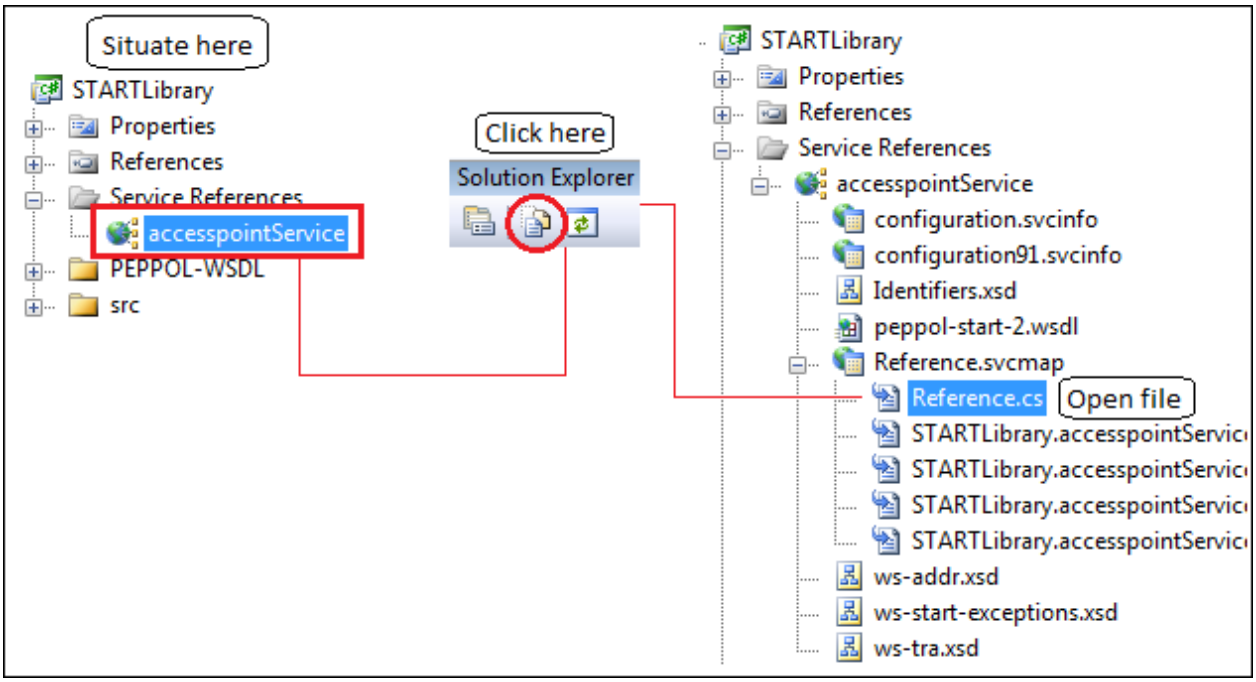


- Once deleted, add a new “Service Reference”. The “Namespace” must be “accesspointService” and the WSDL path must contains XML schemas (xsd) like the images:





- Once added, do the following:



- In the "ConfigurationName" in line **106** from "Reference.cs" you will find this value: "accesspointService.Resource" and you have to replaced it to this: "**STARTLibrary.accesspointService.Resource**"

```
106  009/02/ws-tra", ConfigurationName = "STARTLibrary.accesspointService.Resource")]
```

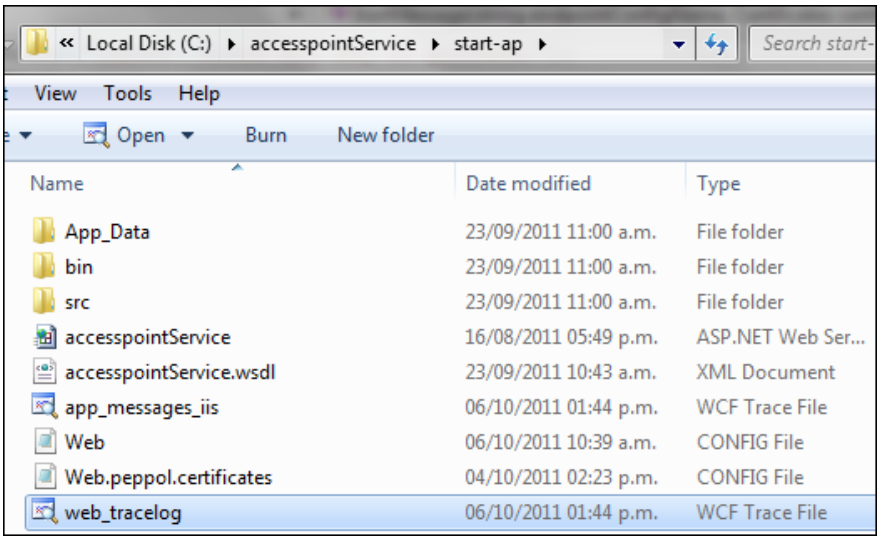
- The final result will look like this:

"[System.ServiceModel.ServiceContractAttribute(Namespace="http://www.w3.org/2009/02/ws-tra", ConfigurationName="STARTLibrary.accesspointService.Resource")]"

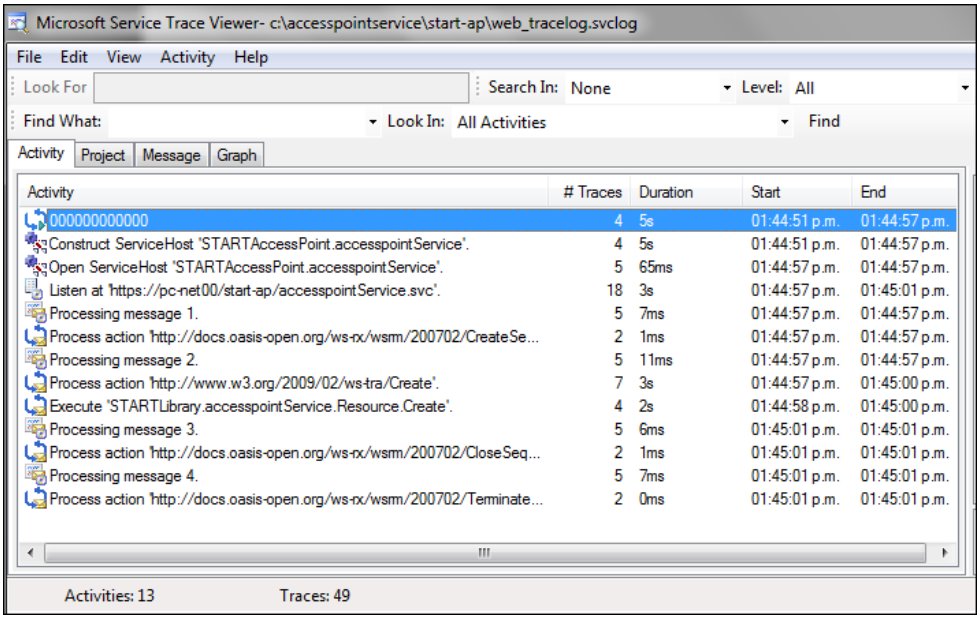
2.3.6.4 Explanation of how to find service and client logs

Service:

1. When your AP is published and you receive documents a trace log will appear in the same folder like this:

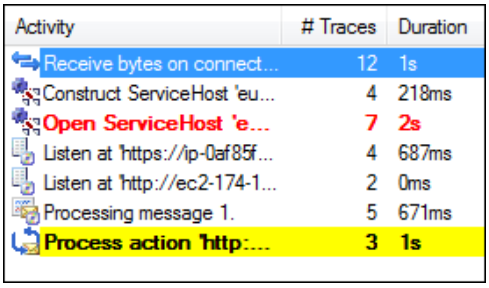


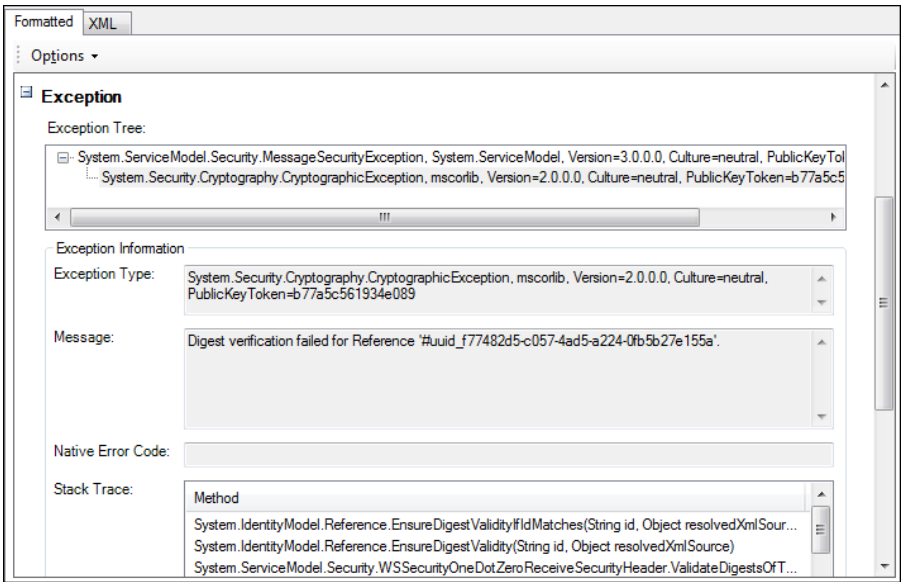
2. Open it and you will see the trace of the service.



3. Click on any activity and you will look details of the action on the right.

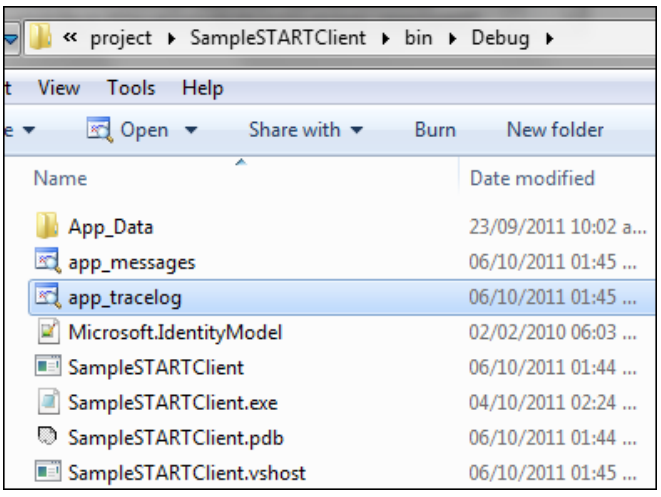
4. Exceptions are highlighted in the Activity view.





Client:

- The same logic is used for the client trace log, in this case you can find it with other name “app_tracelog” and in this path **<project path>\project\SampleSTARTClient\bin\Debug**

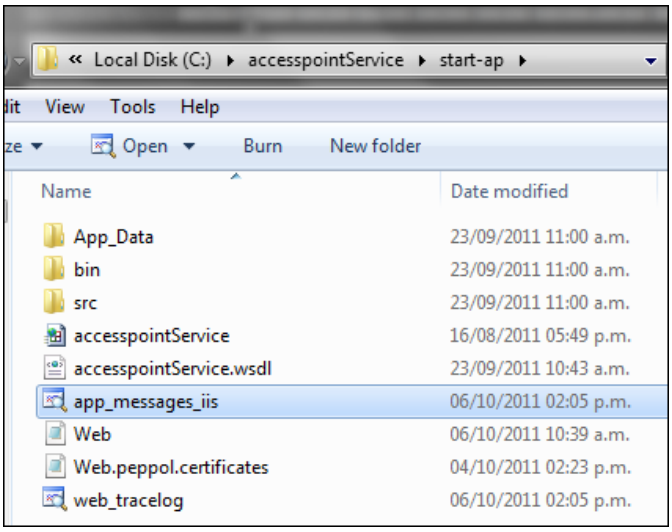


2.3.6.5 Explanation of how to find SOAP messages from client and web service

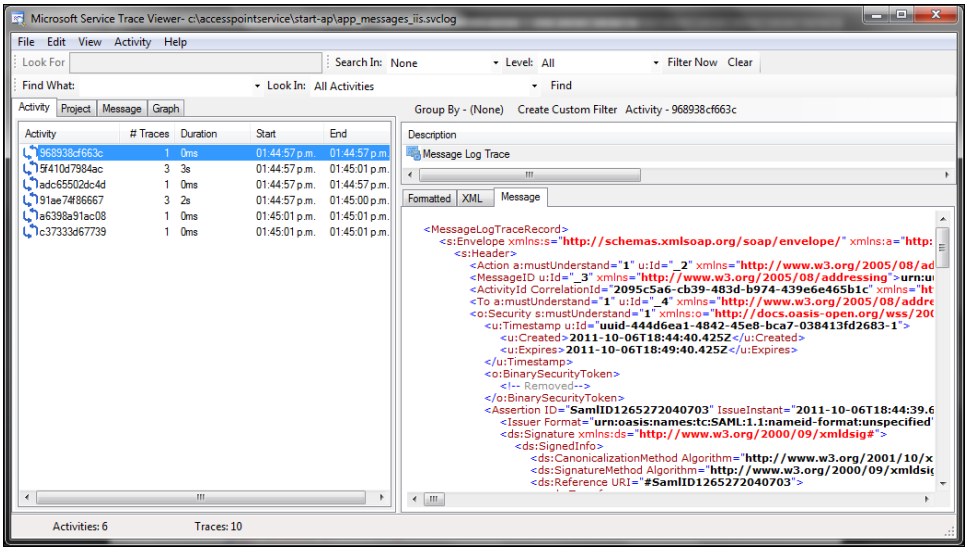
Client --> Access Point	Access Point --> Client	Note
CreateSequence	CreateSequenceResponse	*Reliable Messaging*
Create	CreateResponse	*Business message*
CloseSequence	CloseSequenceResponse	*Reliable Messaging*
TerminateSequence		*Reliable Messaging*

Service:

1. When your AP is published and your client sent documents or you receive a document, a message log “app_messages_iis” will appear in the same folder like this:

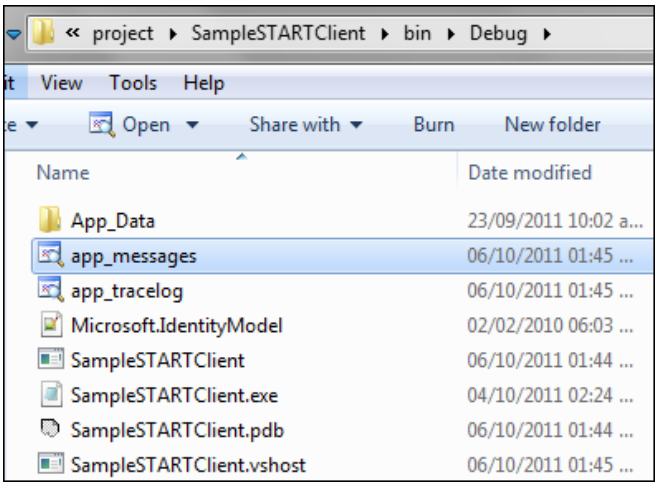


2. Open the file and you will see the SOAP messages received for the service:



Client:

- The same logic is used for the messages log, in this case you can find it with other name "app_messages" and in this path **<project path>\project\SampleSTARTClient\bin\Debug**



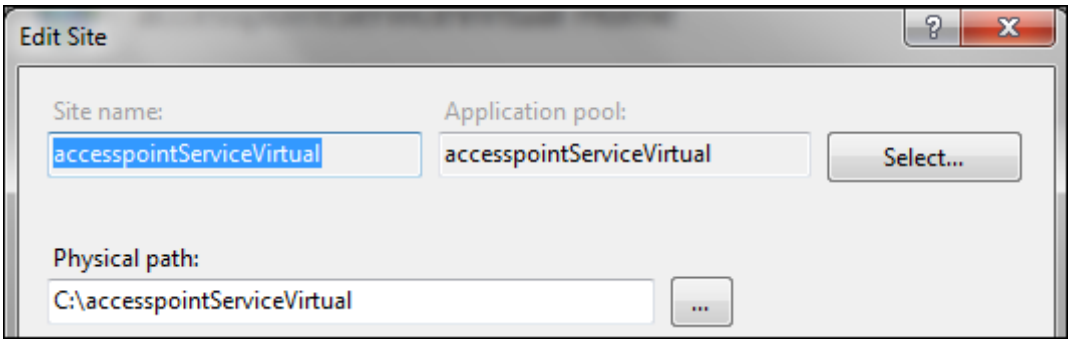
- When opened the file, you will find messages that your client emitted in the process of sending a document.

2.3.6.6 Explanation of how to debug web service methods through https

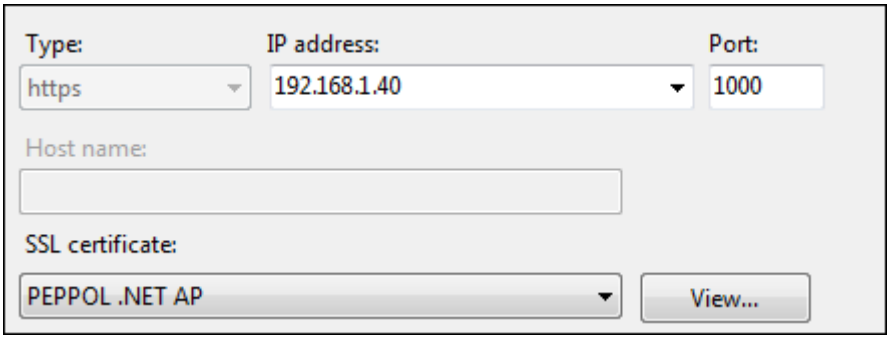
- **If you decide to follow this instructions is strongly recommended to make a copy of your .NET access point project and work with it in these steps.**

- If you don't like this method for debugging, you can just look at the logs for client and service trace that we will talk in the next question about how to find them.

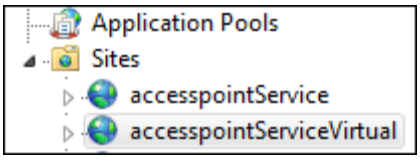
1. Create a new website on IIS (eg. "accesspointServiceVirtual") and also an empty folder and type the path.



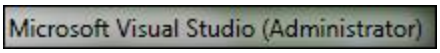
2. Select "https" binding, the certificate you previously made and specify other port (preference to use one that is never used). In this case we used "1000".



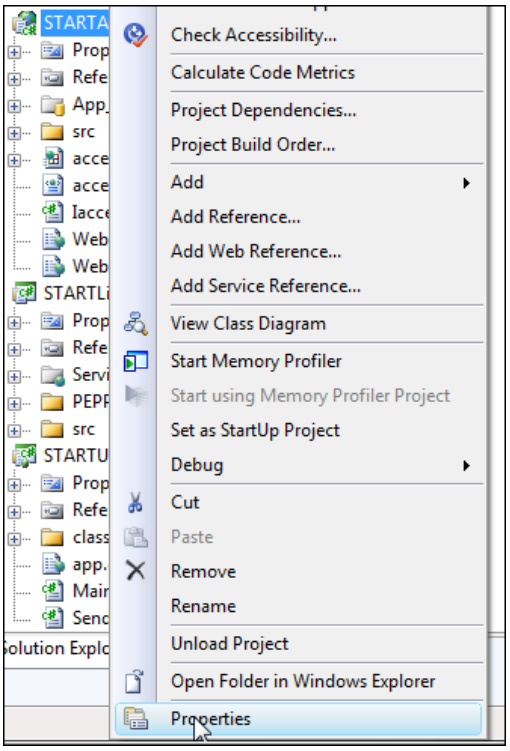
3. Click “Ok” and you will now see in your left the 2 sites that you created “accesspointService” (the one you are using and registered on PEPPOL) and the “accesspointServiceVirtual” (the one you will use for testing).



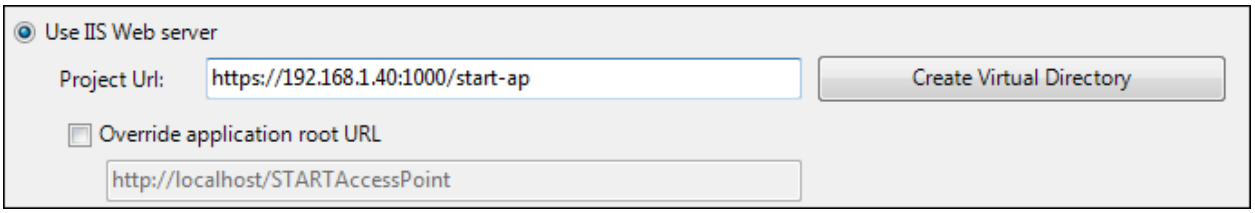
4. Use Visual Studio with admin rights and open the copy of .NET access point project.



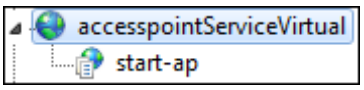
5. Go to “STARTAccessPoint” project, right click and select “Properties”.



6. Select “Use IIS Web server” and type the project url that you will use, in this case: https://192.168.1.40:1000/startap (do not include “accessPointService.svc”) and click on “Create Virtual Directory”.

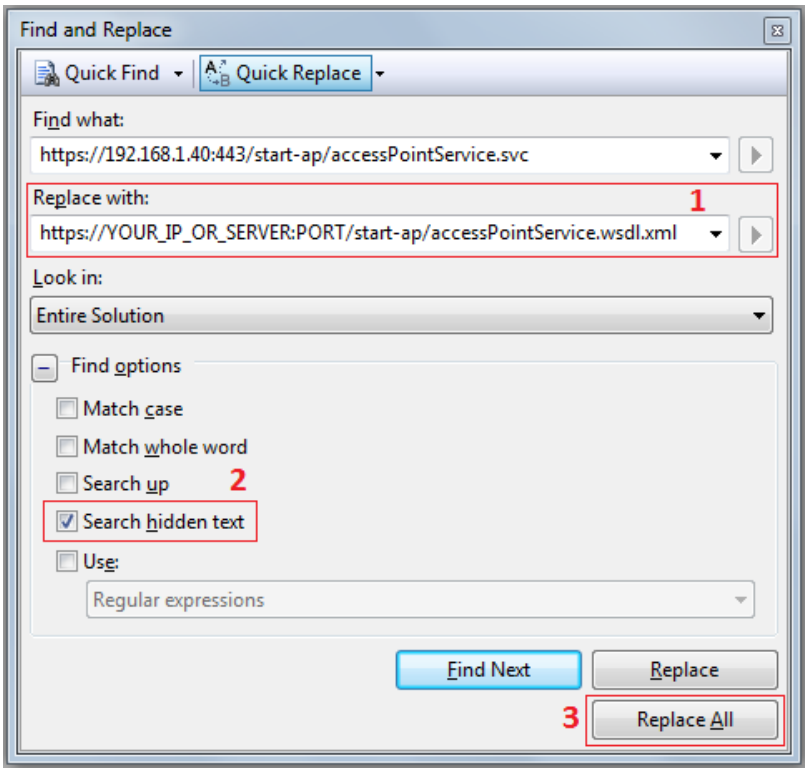


7. Save the properties and rebuild the solution, if you go back to the IIS you will see this:



8. Since now, each time you want to publish the web service, you just rebuild your project solution, it is not necessary to publish it anymore because IIS will work with your project folder.

9. Don’t forget to replace your new url in all the project because it stills with the old one https://192.168.1.40:443/start-ap/accessPointService.svc”



Replace in the “Web.config” file from “STARTAccessPoint”project, the line **207** manually according to your web service URL that will reference the “externalMetadataLocation” (accesspointService.wsdl.xml file).

10. To make tests, start sending a message through your client and a prompt will appear. Click **"Attach Process"**.

11. Change the logic of your "Create" web service method as you want and debug everything, just set a breakpoint and Visual Studio now will take you inside your method.

2.3.7 Testing Access Point

2.3.7.1 Example of sending with client (outbound)

1. It is necessary to get a participant identifier and be registered in PEPPOL SML (Service Metadata Locator) because the web service will make a "lookup" to find the recipient participant identifier and if you are registered or not.

   NOTE:
   Message can be send through 2 projects and you can change the parameters for the message to be sent on "SampleSTARTClient" project "Client.cs" file or at "STARTUnitTests" project, "classes" folder and then "Parameter.cs" file.

   *Where is the document? We just made a sample XmlDocument programmatically. If you want to send other kind of documents, replace line **120** to your xml path like the third image on "Client.cs" file.*

   IMPORTANT:
   Read "Example on receiving (inbound)?" section to know where are your received messages.

   Image 1

```
119     XmlDocument body = new XmlDocument();
120     body.LoadXml("<?xml version='1.0' encoding='UTF-8'?><s:Test xmlns:s=\"http://test.dk\">"
121      + "MessageID: " + "Test" + "</s:Test>");
```

   If you are using "*STARTUnitTests" project*, change it on "SendMessageTest.cs" line **147** like the third image too.

   Image 2

```
146     XmlDocument body = new XmlDocument();
147     body.LoadXml("<?xml version='1.0' encoding='UTF-8'?><s:Test xmlns:s=\"http://test.dk\">"
148      + "Message sent through 'SendMessage' unit test" + "</s:Test>");
```

   Image 3

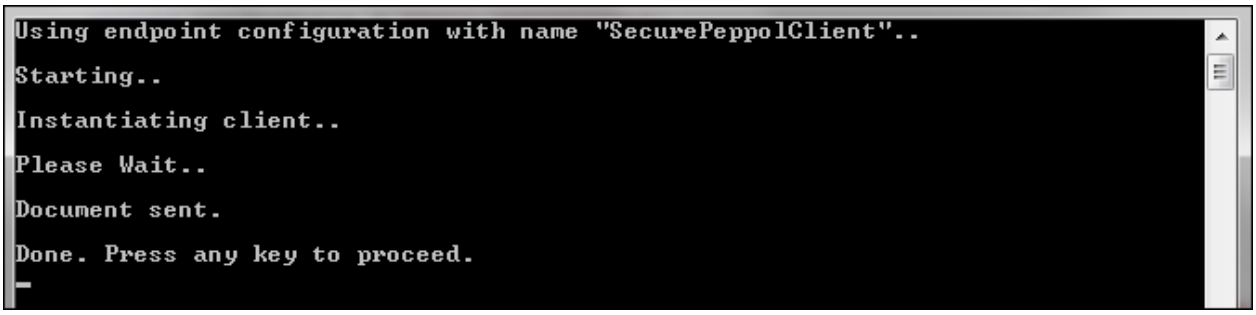
```
XmlDocument body = new XmlDocument();
body.Load(@"D:\test-docs\Invoice.xml");
```

   Now run *"SampleSTARTClient" or "STARTUnitTests" project to send the message (Put one of them as the start up project).*

   Note that in first and second image, the "body.LoadXml" was replaced to "body.Load" because this is a file that will be loaded.

2. You can now send secure messages to yourself forgetting about participant identifiers through “SampleSTARTClient” or through “STARTUnitTests” projects. Just set one of them as the “StartUp Project” and run project (left default values for the “senderParticipant” and “recipientParticipant” they are not needed if you choose ”STARTUnitTests” as your “StartUp Project”).

- Through **“SampleSTARTClient”**:



- Through **"STARTUnitTests"**:

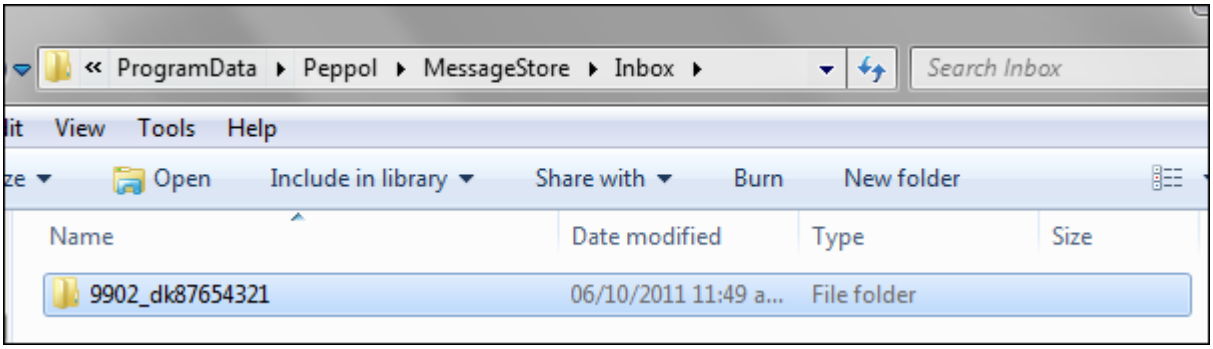
3. Change other parameters as you want (Read the **note** at starting this section), "documentIdValue" and others. In the case you change the logic, no validation will be done because the "lookup" is commented for participants and finding document capabilities of the recipient with other validations won't be done.

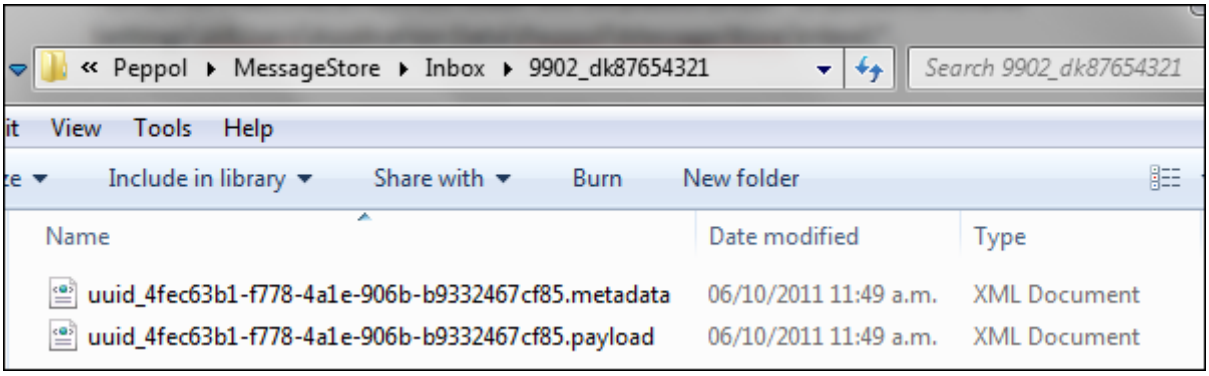
2.3.7.2 Example of receiving (inbound)

When an identifier receives a document, it is usually located on a folder (PEPPOL folder) where the path can be different depending of the O.S:

- On an XP machine, a PEPPOL folder will be placed under "C:\Documents and Settings\AllUsers\Application Data\Peppol\MessageStore\Inbox\".

- Other O.S machines like Windows 7, PEPPOL folder will be placed under "C:\ProgramData\Peppol\MessageStore\Inbox\".





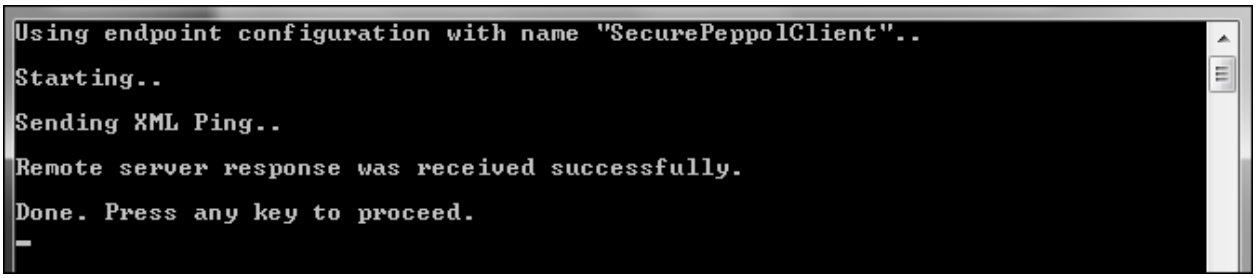
2.3.7.3 Example of making a ping message

This method will always work according to START Profile specification.

1. To make a ping message, go to “SampleSTARTClient” project and comment line **71** and uncomment line **73**.

```
70      //Send message
71      //StartMessage(endpointName, Certificates.FromConfig(endpointName));
72      //Make ping
73      MakePing(endpointName, "https://192.168.1.40:443/start-ap/accessPointService.svc");
```

2. Set “SampleSTARTClient” as “StartUp Project” and replace the URL with the recipient endpoint address (read steps **4-7**) or left with your url and then run project.



3. If the web service is correctly hosted or has the “Ping” method implemented according to the START Profile, you will receive that message.