# XML-DA server-side Gateway Software

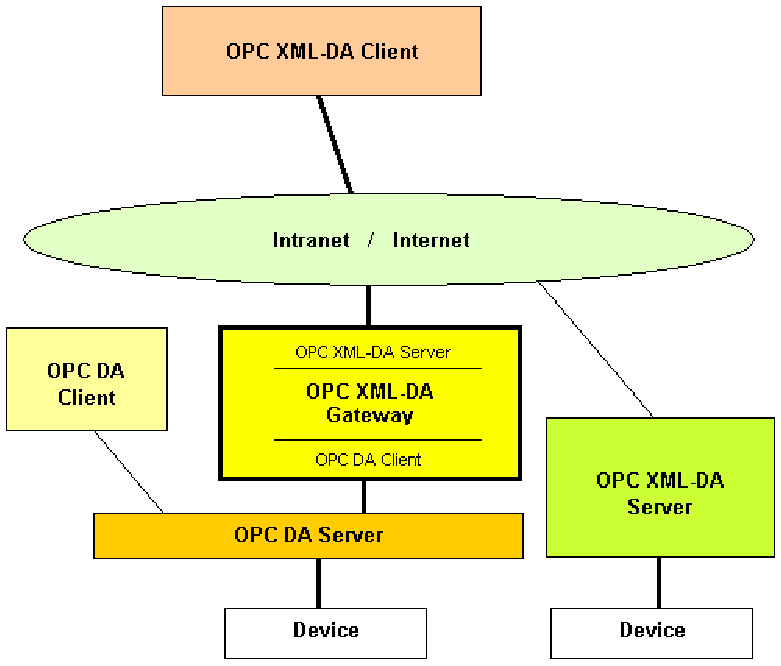
## Overview

The XDAGW-SS gateway enables XML-DA clients to access OPC-DA servers.

Classic OPC DA server can be accessed from remote clients using XML DA web services communication.

The gateway is used when:

- An existent OPC-DA server has to be accessed from a remote location.
- The device has an OPC-DA server for performance reasons. Local COM client can make high performance COM access while remote clients access through the XML-DA gateway.



The XDAGW-SS Gateway software package consists of the following components:

- XML-DA server as an ASP.NET and WCF web service with a built-in OPC-DA client
- Configuration utility
- XML-DA test client
- OPC-DA simulation server and test client as testing tools

## Requirements

- Windows 10/8/7 or Windows Server 2003/2008/2012/2016
- Framework 4.x and IIS with **ASP.NET activated**
- For the WCF gateway version: .NET4.x with WCF activated
- OPC DA V2.05 or V3.0 compliant target server
- 64bit OPC Core components on 64bit systems with XDAGW-SS and the OPC DA server on the same machine.

## Setup

The XDAGW-SS setup has a selection dialog to install the gateway as an IIS web service for:

- ASP.NET
  Use this gateway implementation unless the XML DA server supports WCF and you want to use communication other than basicHttp.
  The IIS application must be configured for a .NET4 application pool.
- WCF
  The WCF communication needs to be properly configured, including buffer sizes.
  For communication with ASP.NET XML DA web services the basicHttp binding must be used.
  Using the WCF Gateway version is advantageous only if the XML DA server supports WCF a communication binding other than basicHttp. is used.

Tools and sample clients are installed depending on the setup features dialog selection.

## Configuration

The gateway web service configuration settings are defined in the web.config file.
The basic gateway configuration is very simple, only the ProgID of the OPC DA server to be handled.
The rest of the configurations are security permission related.

**Security permissions** must be configured to allow the gateway web service application access to the OPC DA server.
The OPC DA server **DCOM configuration** specifies the user accounts that are allowed to launch and access the server. If the OPC DA server is on another machine than the gateway web service then also Windows security settings and the firewall need to allow the access.
For basics on these configurations see the document ***Using OPC via DCOM with XP SP2.pdf***
Hints to help minimize issues:

- If possible install the gateway on the same machine as the OPC DA server. The DCOM configuration is much simpler for a local connection.
- Define an account for the server to run in instead of using the "Launching User' default.
  This can be done in DCOMCnfg server properties – Identity tab

The **IIS configuration** defines the user account for the web service and optionally impersonation for the resource access.
The configuration options and best practices depend on the IIS version.

IIS 5 (XP)

Best define Impersonation in the web.config file to specify a specific user account.
The Impersonation option in the XDAGW-SS configuration utility does this.
If the OPC DA server is on another Workgroup machine then define a user that exists on both machines with the same password.

IIS 7 (Win7,8,10)

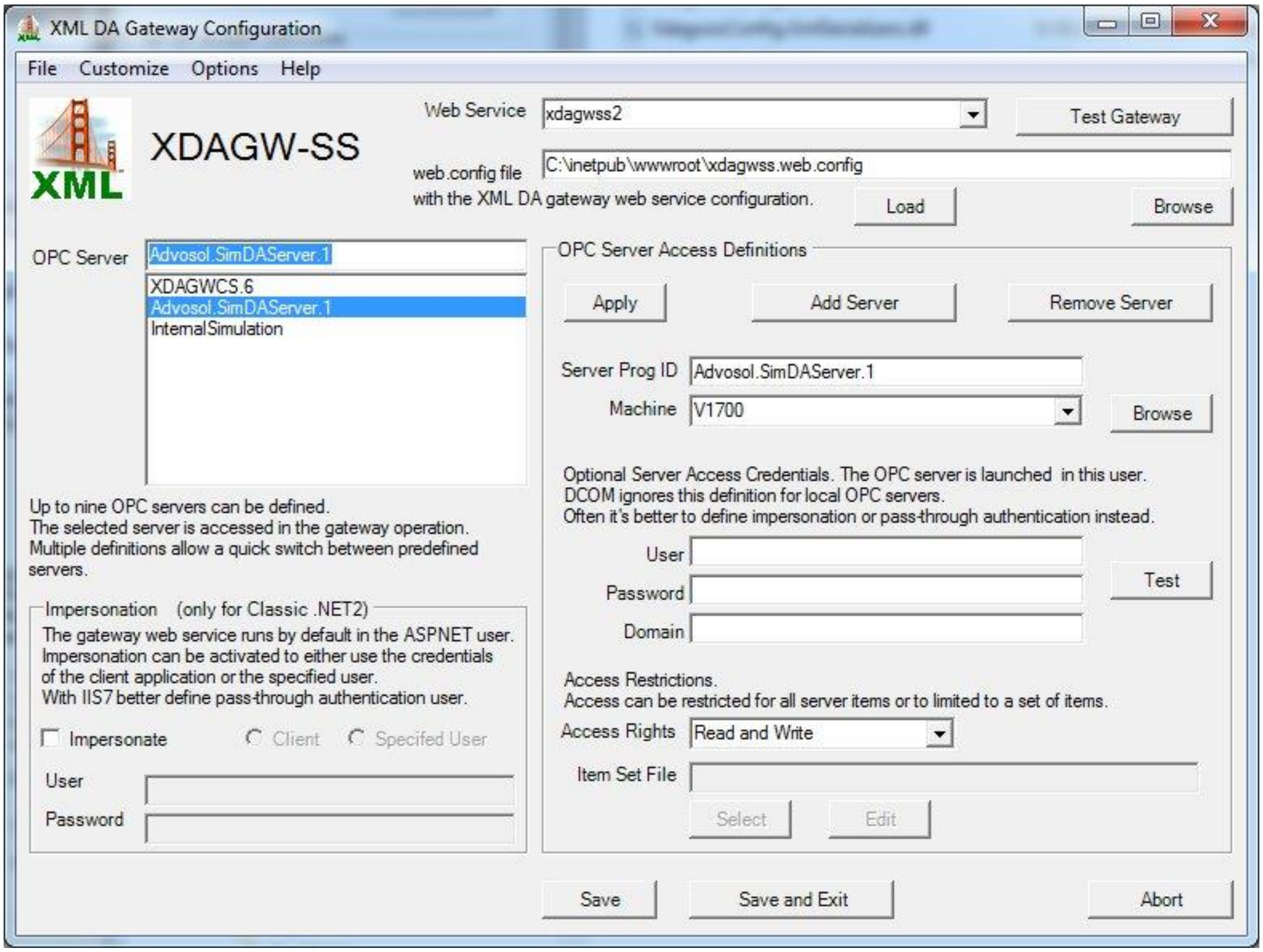
Use the IIS Manager to configure the gateway web service environment.

- Define a .NET4 application pool (classic or integrated)
- Assign the gateway server web service application to this application pool.
- In the web application Base Settings define a "Connect as.." user
  This account is used in place of ANONYMOUS LOGON
  If the OPC DA server is on another Workgroup machine then define a user that exists on both machines with the same password.

ASP .NET must be activated in the IIS setup. In *Control Panel – Programs – Turn Windows Features on or off* check ASP .NET in *IIS - Application development features*.

The **Gateway Configuration** utility displays the definitions and modifies the web.config file according the user selections.

Up to 9 OPC DA servers can be defined and the selected one is actually being used. This is intended to help quickly change between servers.
*InternalSimulation* is a diagnostic feature. Don't delete it. See *Trouble Shooting* for details.



Web Service: xdagwss2

web.config file: C:\inetpub\wwwroot\xdagwss.web.config
with the XML DA gateway web service configuration.

OPC Server: Advosol.SimDAServer.1
- XDAGWCS.6
- Advosol.SimDAServer.1
- InternalSimulation

Up to nine OPC servers can be defined.
The selected server is accessed in the gateway operation.
Multiple definitions allow a quick switch between predefined servers.

Impersonation (only for Classic .NET2)
The gateway web service runs by default in the ASPNET user.
Impersonation can be activated to either use the credentials of the client application or the specified user.
With IIS7 better define pass-through authentication user.

☐ Impersonate ○ Client ○ Specifed User

User

Password

OPC Server Access Definitions

Server Prog ID: Advosol.SimDAServer.1

Machine: V1700

Optional Server Access Credentials. The OPC server is launched in this user.
DCOM ignores this definition for local OPC servers.
Often it's better to define impersonation or pass-through authentication instead.

User

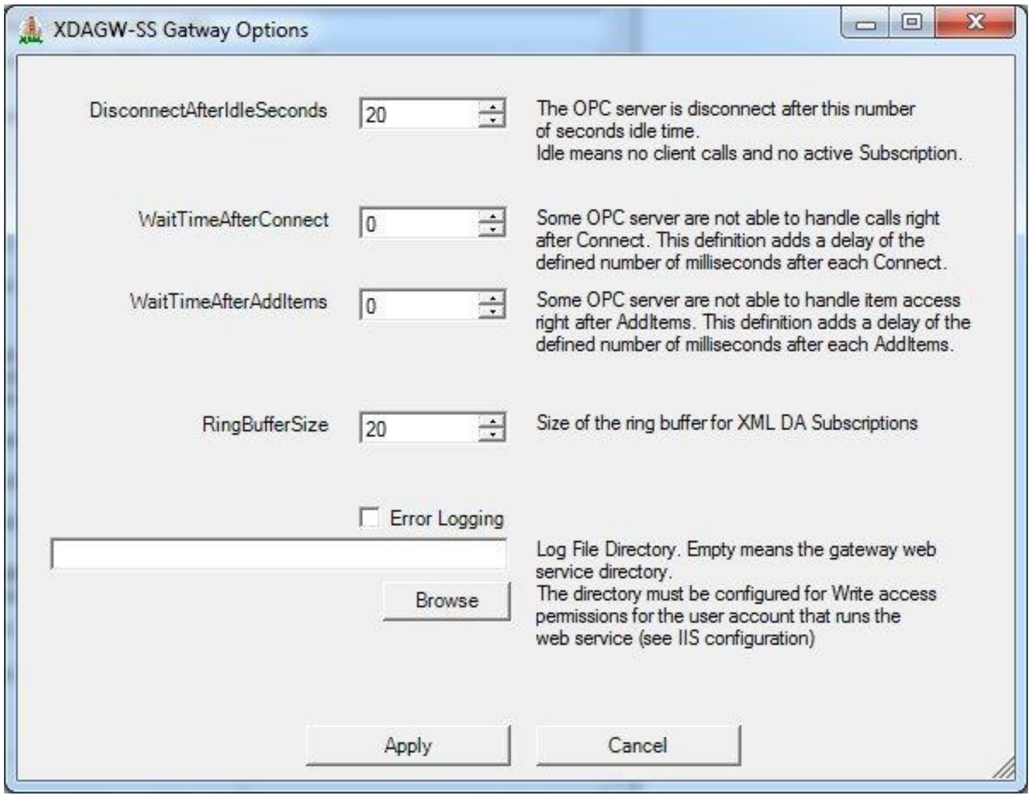
Password

Domain

Access Restrictions.
Access can be restricted for all server items or to limited to a set of items.

Access Rights: Read and Write

Item Set File



| | | |
|---|---|---|
| DisconnectAfterIdleSeconds | 20 | The OPC server is disconnect after this number of seconds idle time. Idle means no client calls and no active Subscription. |
| WaitTimeAfterConnect | 0 | Some OPC server are not able to handle calls right after Connect. This definition adds a delay of the defined number of milliseconds after each Connect. |
| WaitTimeAfterAddItems | 0 | Some OPC server are not able to handle item access right after AddItems. This definition adds a delay of the defined number of milliseconds after each AddItems. |
| RingBufferSize | 20 | Size of the ring buffer for XML DA Subscriptions |

☐ Error Logging

Log File Directory. Empty means the gateway web service directory.
The directory must be configured for Write access permissions for the user account that runs the web service (see IIS configuration)

## Installation Details

On Windows Vista the .NET Framework and the Windows Communication Foundation (WCF) is installed with the system. On earlier Windows systems the web server (IIS) and the .NET Framework have to be installed as optional components.

### Install IIS and .NET Framework

If the required software components are not yet installed then you need to cancel the setup and install the software as follows:

**1. Install IIS** ( Internet information Services ) if not already installed
Windows 7, 8, 10
IIS is activated in "*Turn Windows Features On or Off*".
*ASP.NET* in *Application Development Features* must be selected.

Windows XP:
IIS is part of Windows-XP Professional but is not installed in the default installation.
From the Windows-XP Professional distribution CD start *'Install Optional Components'* and select *IIS*.
----> There is now a directory tree *'c:\Inetpub\wwwroot\...'*
that's also accessibly as Web Folders '\*localhost\...'*

**2. Install the .NET Framework** if not already installed
Windows 7,8,10
The .NET Framework versions are installed. The ASP.NET selection in the IIS activation installs .NET properly into IIS.
Windows XP:
If Visual Studio .NET is installed on this computer then the .NET Framework is already installed.
Otherwise it needs to be installed from the .NET redistributable file *DotNetFx.exe*
Download the file *DOTNETREDIST.EXE* from microsoft.com and unpack the downloaded file to get *DotNetFx.exe.*

**Registering ASP.NET on IIS after installing the .NET Framework**
If the .NET Framework is installed on a system that has IIS already installed, IIS is automatically configured to handle requests to ASP.NET pages, and to redirect the execution to the ASP.NET runtime. However, it may happen that the .NET Framework was installed on a system where IIS was not already present, and IIS was added later. In this case IIS is configured to handle web services and does not know the asmx file extension.
Registering ASP.NET on IIS is not just a matter of associating the various .aspx, .asmx, .axd, .ashx and the other ASP.NET extensions to the aspnet_isapi.dll ISAPI. More has to be done to create the ASP.NET account and to set it for ASP.NET requests, register the ISAPI itself and other stuff. Doing all this manually is a difficult operation, and requires a good understanding of many details.
Fortunately there is a utility, shipped with the .NET Framework but not documented, that can take care of these configuration chores for you.
The utility is *aspnet_regiis.exe*, it is located under
%WindowsDir%\Microsoft.NET\Framework\vx.y.zzzz\ and you should call it with the -i parameter:
***aspnet_regiis.exe –i***

### Installation of the OPC XML-DA Gateway Software

The setup installs features as selected in the feature dialog:

- Gateway Web Service
  - WCF version'
- Tools
  - Test Client
  - Simulation Test OPC DA server
- Documentation
- Sample XML DA client applications

## Security

In remote access application security is of critical importance and has to be properly implemented on different levels:

| | |
|---|---|
| OPC Server Access | Windows protects COM components with a complex permission checking system. Depending on your security requirements you need to make the appropriate settings. See the chapter Configuration for an overview. |
| Web Service Access | The XML-DA Gateway Web Service may be accessible from the Internet and therefore needs to be protected from unauthorized access. Anonymous access should be disabled and the proper authentication selected. |
| Secure Communication | **WCF Gateway Version**<br>WCF (Windows Communication Foundation, also called .NET3) supports variety of communication protocols.<br>The basicHTTP binding is ASP.NET web service compatible and has to be used if not both, client and server, are WCF applications.<br>Several kinds of bindings for secure communication can be selected, including X.509 certificates.<br>Use the WCF configuration tool to configure the communication settings.<br>See chapter *WCF based Web Server Access* for details. |
| OPC Server Item Access | Some applications may want to restrict the access to the OPC-DA server to e.g. read-only access or to only a subset of its items. The XDAGW-SS gateway web service can be configured to handle such restrictions.<br>See XML-DA Gateway Configuration for details. |

## XDAGW-SS Configuration Details

The XDAGW-SS Gateway Web Service can be configured by a number of definitions in the ***WEB.CONFIG*** file that is in the same directory as the *OpcXmlDaGateway.asmx* file.
The web service virtual directory can be found in the IIS manager under the web service application name. The default name is: ***http://localhost/xdagwss.***

For WCF services the essential WCF definitions Endpoint/Binding/Contract are defined in the ***<system.serviceModel>*** section. There may be multiple <*service*> definitions with different bindings.
The EndPoint name can be selected according the user preferences. It is to be used in the client application.
The Binding has to be defined as ***basicHttpBinding*** for compatibility with ASP.NET clients. For WCF clients any of the available bindings may be used.
The Contract has to be ***xmldanet.Service*** for the gateway web service.

The Gateway configuration settings are defined in the ***<AppSettings>*** section and direct and restrict the access to the underlying OPC-DA Server.

### OPC-DA Server Definition

The OPC-DA server is defined in two steps. A list of OPC servers can be defined and a further definition specifies which server definition is to be used.
For more efficient debug handling the server can be defined in the client handle. This feature should be used for test purposes only.
The **active OPC Server** is

- Defined in the web.config file
  The configuration definition: `<add key="ServerIdSource" value="Definition 2" />`
  makes the OPC Server defined in definition *ServerId2* the active server.

- Defined by the client application ( Intended only for test purposes! )
  The configuration definition: `<add key="ServerIdSource" value="ClientHandle" />`
  forces the server name to be taken from the passed *"Client Handle".* This is intended to make testing with the test client more flexible, allowing access to different OPC servers without changing the configuration file. The client needs to provide the OPC Server ProgId Name or the defined server alias name in the *Client Handle* field with each request.
  The gateway can handle access only to one OPC DA server. The client is not allowed to make parallel access to different OPC DA servers.

### List of supported OPC Servers

The configuration definitions

```
   <add key="ServerId1" value="InternalSimulation" />    <!-- no OPC server is accessed. Use this to locate
access problems -->
   <add key="ServerId2" value="OCSTK.DA.Sim.32" />
   <add key="ServerId3" value="\\I-3000\Matrikon.OPC.Simulation.1?User=xx?Password=xx?Domain=xx" />
   <add key="ServerId4" value="Softing.OPCToolboxDemo_ServerDA.1?User=xx?Password=xx " />
   <add key="ServerId5" value="\\ccccc\OCSTK.DataSample.32" />
```

define the Prog Id names of the OPC-DA Servers. The OPC-DA Servers need to be installed and registered under these names.
Only the OPC-DA Server *OCSTK.DA.Sim.32* is supplied in the distribution package as a simulation test server.

### Server ProgId and Access Info

The ProgId defined in the OPC Server List may be extended with computer name and access information.
When only the ProgId is defined then the OPC server is accessed locally, using the .Net COM InterOp services. Otherwise access is done through Win32 functions with Authentication at Connect and Impersonation, using the definitions appended to the ProgID.
For access to remote OPC servers and to be able to define credentials, a computer name need to be specified.
The format is:
a) the ProgId only for local OPC DA-Servers. E.g. *"OCSTK.DA.Sim.32*""
b) with computer name for remote servers. E.g. *"//I-3000/OCSTK.DA.Sim.32*"
c) with access definitions. E.g. *"//I-3000/OCSTK.DA.Sim.32?User=xx&Password=xx&Domain=xx*"
d) optionally the definition can be preceded with *opda:* E.g. *"opcda://I-3000/OCSTK.DA.Sim.32"*

**Server Access Restrictions**
Access to the OPC Server can be restricted by the following configuration definitions:

```
    <add key="ServerAccess1" value="RW" />     <!-- read/write access for server 1   -->
    <add key="ServerAccess2" value="RO" />     <!-- read only access for server 2    -->
    <add key="ServerAccess3" value="WO" />     <!-- write only access for server 3   -->
    <add key="ServerAccess4" value="ITEM" />   <!-- read/write access for server 4   -->
    <add key="ServerAccess5" value="RW" />     <!-- read/write access for server 5   -->
```

The RW,RO,WO definitions limit the access to all items in the server's address space. Access can be limited to a subset of the items be defining an item list (see below)
If "item defined access rights" are selected ( as above for server 4 ) then there has to be an *ItemList*x definition and an item definition file with the access right definition in the first column.

**Item List Files**
The access rights can be limited to a subset of the server address space.
If there is a configuration definition like:

```
    <add key="ItemList4" value="ItemsTSDA.def" />
```

then only the items listed in the definition file
- are displayed in browse operations
- can be accessed in read/write operations

The definition file is a simple text file with one item per line.
Sample:

```
RW DW_SPEED1
RO DI_SWITCH1
WO DO_LED1
RW BSTR_With_10_WCHAR
WO InOut_UI4
```

Lines that don’t start with either “RO “, “WO “ or “RW “ are treated as “RO” with the full line used as the item name.
The item name must be the full XML DA ItemName as returned from the XML DA server in a Browse result.

**RingBufferSize**
The size of the XML-DA Subscription ring buffers.
When a client creates a subscription with *enableBuffering=true* then the XML-DA server maintains a ring buffer for each subscribed item. The default size is 20.

```
    <add key="RingBufferSize5" value="20" />
```

**WaitTimeAfterConnect** in ms
Some OPC-DA servers need some time after they are connected before they are ready to be called. By default there are no wait times. If there are problem with OPC-DA server access then define a time of e.g. 1000 ms and when everything is working properly, try to reduce the wait time.

```
    <add key="WaitAfterConnect" value="0" />
```

**WaitTimeAfterAddItems** in ms
Some OPC servers need some time after items are added to group, before functions of this group may be called. By default there are no wait times. If there are problem with OPC-DA server access then define a time of e.g. 500 ms and when everything is working properly, try to reduce the wait time.

```
    <add key="WaitAfterAddItems" value="0" />
```

**DisconnectAfterIdleSeconds** in seconds
The OPC server is disconnected if there are no active subscriptions and there was no client call for the specified number of seconds. The default time is 20 seconds.

```
    <add key=" DisconnectAfterIdleSeconds" value="20" />
```

**Force V6.2 Compatibilty**
Timestamps were returned as local time instead of UTC in versions previous to V7.0.
The AppSettings definition

```
     <add key="ForceV6.2TimestampsLocalTime" value="true"/>
```

forces timestamp to be returned as they were in versions previous to V7.0.
The default for this setting is FALSE.

## Trace and Error Logging

The gateway can be configured to log error and trace information:

a) in the communication with the OPC DA server
b) in the processing of the XML DA client requests

The logging is into a monthly file with the name ***xdagwss.log.MMM.txt***
MMM represents the month abbreviation, e.g. JAN, FEB

The application configuration defines the directory in which the gateway log file is created. The default is the project web directory.
The IIS worker process must have write rights for the directory ! The may require the IIS worker process user account being added to the security settings of the directory.
The first access to the gateway web service fails with an exception if the process doesn't have write rights. In the next service access the gateway will work with logging disabled.
Sample **configuration** definitions:

```
<!-- Directory in which the log file is created. The default is the project web directory.
     The user ASP_NET must have write rights for the directory.
     The log entries are appended to the file "xdagwss.Log.MMM.txt" in the specified directory.
     MMM represents the current month.
     Sample: <add key="LogDirectory" value="" />          the web service directory
             <add key="LogDirectory" value="c:\" />
             <add key="LogDirectory" value="disable" />   disable all logging  -->
     <add key="LogDirectory" value="disable"/>

<!-- Enable the logging of client requests.
     Levels:  0=disabled, 1=log on method level, 2=log on method and item level -->
     <add key="LogLevelClientRequest" value="0"/>
```

## Testing the Installation

The XML-DA Gateway is easy to use and simple to configure. However for it to work, a number of communication interfaces have to be configured correctly.
It is therefore recommended to test the installation step-by-step as follows:

### 1. Test the XDAGW-SS Gateway locally

1.1 Use the OPC-DA Test client to verify that the OPC-DA server is registered and accessible

1.2 Check the configuration file inetpub\wwwroot\xdagwss\Web.config to see if the correct ProgId of the OPC-DA server is defined.
For WCF make sure that the same binding is configured as in the client. The contract has to be defined as ***contract="xmldanet.Service".*** Use the ***basicHTTP*** binding for Asp.NET compatibility.

1.3 Start the XML-DA Test Client, either the ASMX or WCF version

1.4 Select or type the proper XML-DA web service URL, e.g.
http://localhost/OpcXmlGateway/OpcDaGateway.asmx
In the WCF version also the proper EndPoint name has to be defined.

1.5 Select the *Get Status* tab and click the *Get Status* button
→The XDAGW-SS gateway web service is accessed and it calls the configured OPC-DA server to request its status. Either the status or an error message is displayed.
The first time this may take while because a number of system components may need to be loaded.
Errors are most likely due to wrong configuration or security permission problems. The gateway runs as an ASP.NET web service and launches the OPC-DA server under the user *ASPNET*. DCOM must be configured to have user that ASPNET defined in the launch and access user list.

Sometimes it's not obvious if the access problem is with the web service or the OPC-DA COM server. To find the source of the access problem, the gateway configuration can be changed to *"Internal Simulation"* (Definition 1). In this configuration no OPC-DA server is accessed. If there is still an access problem then it has to be the web service access. You may also try to access the web service with the Internet Explorer. Just enter the URL of the service, e.g. http://localhost/OpcXmlGateway/OpcDaGateway.asmx. If the service is accessible then a list of the web service methods is displayed.

**2. Test the XDAGW-SS Gateway remotely**

2.1 Determine the network name or the IP address of the computer running the Gateway

2.2 Start the Web browser and enter the URL http://xxx.xxx.xxx.xxx/OpcxmlGateway/OpcDAGateway.asmx ( xxx…) is the computer IP address or network name. The browser does only read the web service definition and does not actually start it.
→ A page with the description of the methods supported by the XML-DA Gateway is displayed if the URL address is correct and the service is accessible

2.3 Install the XML-DA test client and start it

2.4 Select or type the URL previously used in the browser

2.5 Select the *Get Status* tab and click the *Get Status* button
→ You should get the same result as describe in 1.5

**Testing the client application with a working gateway**

The XDAGW-SS gateway service is available through Internet access at
http://advosol.us/XMLDADemo/ts_sim/OpcDaGateway.asmx
To test your client application you can access this gateway. It is mapped to the same OPC DA simulation server that is provided with the gateway.

## Trouble Shooting

If the XML DA client (e.g. the Advosol XML DA Test Client) cannot not successfully access the XDAGW-SS gateway web service then the issue may be either in:

- The web service access
- The OPC DA server access form the gateway web service.

The error message cannot always indicate the cause.

Step 1: **Test if the gateway web service is accessible**

a) Check if the correct URL is used.
The usual URL is: http://localhost/xdagwss/OpcDaGateway.asmx
'*xdagwss'* is the name of the gateway web service application in IIS.
The file '*OpcDaGateway.asmx'* must be in the web service directory.

b) Access the web service with the Internet Explorer.
A page with the names of the eight XML DA interface methods is displayed if the access is successful. Otherwise error messages are shown that are usually helpful.
The Internet Explorer retrieves data but does not actually start the web service.

c) Configure XDAGW-SS for Internal Simulation
and access the gateway with the XML DA test client. This starts the gateway without access to an OPC DA server and is a good indication if the issue is in the OPC DA server access.

Step 2: **Test the OPC DA server access**

If possible install XDAGW-SS on the same machine as the OPC DA server. This simplifies the security permissions configuration significantly. For remote access Windows and the Firewall must be configured to allow the access in addition to DCOM. Consult the document *Using OPC via DCOM with XP SP2.pdf* for an explanation of the configuration steps.

**User Account for the OPC DA server access**
Best practice is to configure the account to be used in the IIS configuration.
IIS7: In the web service *Basic Settings* open '*Connect As..'* and define a specific user
IIS4: Define *Impersonation* with a specific user in the XDAGW-SS gateway configuration
For remote OPC DA server access in Workgroup network configuration the configured user account must exist on both machines with the same password.

**64bit Systems**
On 64bit systems the XDAGW-SS runs in 64bit mode. 64bit OPC Core Components need to be installed if the accessed OCP DA server is on the same machine as the gateway.
The Advosol 32/64bit OPC Core Components are part of the XDAGW-SS distribution. Make sure that the Core Components are selected in the setup features dialog. You may have to run the setup in Modify mode.

**Gateway Logging**
Gateway logging can be enabled in the Option dialog of the XDAGW-SS configuration utility.
The IIS worker process must have write access to the configured directory. This is usually not the case by default. With missing Write permission to the gateway web service fails with an exception. In the next service access the gateway will start with logging disabled.
The user account used by the IIS worker process is e.g. shown in the Windows Task Manager. IIS7 may have multiple worker processes. In this case use the IIS Manager to determine the application pool assigned to the web service and then the account configured in the application pool.

## Client Development

In Visual Studio .Net the development of web service client applications is simple. Create a new application and add the web service proxy stubs. This can be done in two ways:

a) Select *Add Web Reference* and enter the URL of your webservice. The Reference file created and added to the project. The Url property is initialized with the URL of the referenced web service. To access another web service the Url property can be changed.
b) Use the wsdl.exe tool to convert the XMLDA10.WSDL file. Add the generated source file to the project. You may also need to add System.Web.Services and System.XML to the references. The Url property needs to be assigned in the application.

The following sample code calls the web service *GetStatus* method:

```
Service srv = new Service();
srv.Url = "http://localhost/xdagwss/OpcDAGateway.asmx" ;
ServerStatus stat ;
try
{
   ReplyBase reply = srv.GetStatus("en-us", null, out stat ) ;
}
catch( Exception ex )
{
}
```

**Write Calls**

XML-DA and OPC DA V3 allow the writing of quality and timestamp together with the value, while OPC-DA V2 does not support this feature.
The XDAGW-SS gateway determines if the associated OPC-DA server is V3 compliant and uses the WriteVQT() function if this is the case. Otherwise the OPC-DA V2 Write() function is used and the specified quality and timestamp values are ignored.

The OPC XML-DA specification can be downloaded from http://www.opcfoundation.org

# Tools

## XML-DA Test Client

From the program menu select the *XDAGW-SS* program group and start the *XML-DA Test Client*.
This test client can access XML DA and OPC DA servers. If the server URL starts with "http://" then the server is accessed as an XML DA web service, otherwise as an OPC DA V2 server.
To access the XDAGW-SS gateway:
In the *Service URL* combo box select or type the URL of the XDAGW-SS gateway, e.g.
*http://localhost/xdagwss/OpcDaGateway.asmx*
Click Browse to browse the items of the OPC-DA server and select the items in the browse control for other operations.

### Get Server Status

By clicking this button the active OPC Server is accessed and some status information is displayed.

### Browse Operations

The root is browsed and displayed. Branches are browsed when they are first selected. Don't check *All Initially* if the server has a large address space. Browsing the whole server could take a long time.

### Read Operations

Select the item(s) in the browse control and right-click. In the pop-up menu select *Copy selected items to Read List* .
Select the *Read* tab and then click the *Read* button.

### Write Operations

Select the item(s) in the browse control and right-click. In the pop-up menu select *Copy selected items to Write List*
Select the *Write* tab and right click the item name. In the modify item control change the value and optionally the quality/timestamp.
Click the *Write* button when all items show the value to write.

### Subscriptions

For subscribed items the XML-DA server buffers value changes and returns all changes in the next client poll.
To create a subscription, select the item(s) in the browse control and right-click. In the pop-up menu select *Copy selected items to Subscription List*
Select the *Subscriptions* tab and click the *Add* button. The handle of the added subscription is displayed.

### Refresh

To poll new values of the subscribed items select the *Refresh* tab and click the *Poll* button.

### Get Properties

The client can read all or some of the properties defined for the requested items.
Select the item(s) in the browse control and right-click. In the pop-up menu select *Copy selected items to Properties List*
Select the *Properties* tab and select the properties to be read. Click the *Get Properties* button to read the item properties from the server.

## OPC Security Analyzer

The OpcSecurityAnalyzer utility helps resolving OPC DA server access problems due to DCOM and Windows Security settings.
Errors are explained and user settings can quickly be changed to check if the access is properly allowed/denied.

## WCF based Web Service Access

The XDAGW-SS gateway is provided in versions for ASP.NET4.x and WCF.
The WCF version communicates through WCF and requires WCF configuration settings in the application configuration file.
These definitions can be created and edited with the WCF *SvcConfigEditor* utility.
The WCF basicHttpBinding is compatible with ASP.NET. The binding needs to be used to communicate with ASP.NET based XML DA servers.

**WCF Advantages**

The WCF communication brings advantages only when client and server are based on WCF. In this case the server and client can be configured to use any of the WCF supported communication options. The communication can be configured for TCP communication for high performance or encrypted communication for high security.

**Disadvantages**

The configuration is more complex because WCF offers many communication options.
The WCF default communication buffer sizes are rather small (only 64kB) and often need to be increased to allow the required number of items and values to be handled.