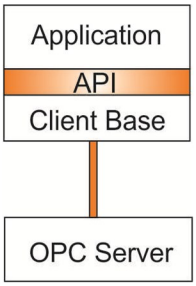
.NET OPC Client Applications

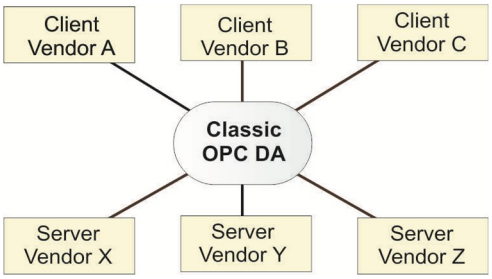
March 2012, Copyright Advosol Inc.

Overview
This paper gives an overview of the different OPC specifications and shows how developers can use Advosol's client base products to create .NET applications with flexible OPC server access.



Developers have many options for the development of .NET OPC client applications. What's best depends on many factors and on personal preferences. The basic choices are the **API** (Application Interface) and the **communication**.
Application requirements, such as the server location may dictate a specific communication type. Mismatches of requirements and existing servers can be resolved with converter servers. E.g. a remote OPC DA server can be accessed wit XML DA communication thru an XML DA converter server.

OPC Specifications

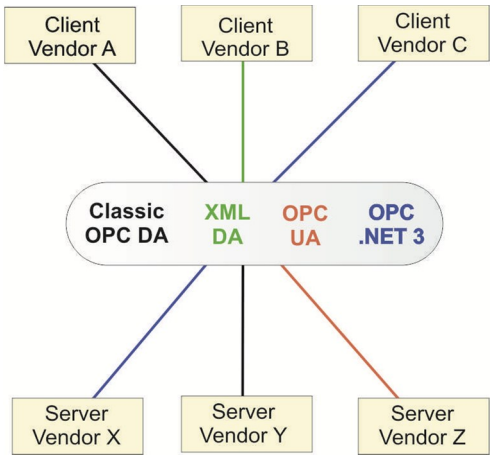


Before the first OPC specification was published in 1996 each application needed a set of drivers to be able to communicate with devices from different vendors.
The OPC specification simplified the situation dramatically. Server and client applications had to implement only the OPC interface and with this standardized interface the clients can access servers from different vendors.

Over the years the application environment evolved. C# and .NET are used in place of C++. This required OPC .NET wrappers, resulting in vendor specific interfaces, similar to a number of dialects instead of a single language.

With Microsoft moving from DCOM to web service based communication OPC released specifications designed for the newer environments.
With the large number of installed classic OPC applications there will be a long transition period with four different OPC specifications being used. Clients must either use converter servers or have multi-specification capabilities.

Application designers must decide which OPC specification to use.



Basic Considerations for the Client Design

OPC client designers need to consider:

- The types of OPC servers the client needs to be able to access,
  Classic OPC DA,HDA,AE, XML DA, OPC UA, OPC .NET 3 (Xi)
- The location of the servers
  For local servers (on the same machine as the client) the type of communication is not of much importance. DCOM is still a good choice. However, remote server access may not be possible with DCOM or is difficult to configure.
- OPC Component Application interface

OPC Application Interfaces

OPC client applications increasingly have to access different types of OPC servers. This can be accomplished with converter servers or with client components with embedded converters for the handling of multiple types of OPC servers thru the same API.
The chosen API and the servers that need to be accessed determine the needed wrappers.
It needs to be considered that the server access behavior thru a converter is likely not 100% equal to the direct access. Timing may be different and certain special features may not be available. Application designers should carefully assess which approach best fits the current and future application needs.

OPC defined interfaces are:

Classic OPC The classic OPC DA, HDA, AE specifications define a C++ COM interface and a COM Automation interface. For .NET the OPC Foundation provided sample .Net wrapper code but didn't create a .NET API specification.
Vendors offer .Net client components with widely different APIs:
- Methods that mirror the OPC specified interface functions.
  The application can use all OPC specified features.
- Usage oriented method that make the server access simple.
- Embedded into an overall product concept.

XML DA The OPC XML DA specification was created in 2003. It defines .NET2 web service SOAP messages for the functionality of OPC DA (Data Access). See OPC-DA / XML-DA Comparison for an overview and comparison of the two OPC specifications.
There are no corresponding web service specifications for OPC HDA and OPC AE.
The .NET web services WSDL tools create API methods from the specification.

OPC UA OPC UA was started in 2003 with the ambitious goal of a multi-platform specification with the capability to model all kinds of systems.
The specification was partially released in 2009 and vendors offer UA products, mostly with only DA functionality.
The OPC UA specification defines communication records. The multi-platform capability requires UA specific communication stacks for each platform.
The communication stack implementation defines the application interface.

OPC .NET 3 In 2009 the OPC Foundation adopted the Express Interface (Xi) specification as the OPC .NET 3 (WCF) API specification.
Xi was created by a number of OPC vendor companies as a .NET interface specification with the features of OPC DA, HDA, AE. It is designed to make the implementation of wrappers simple and efficient.
OPC .NET 3 is based on the Microsoft Communication Foundation (WCF) and specifies a set of WCF contracts. The contracts define the server and client API.

Type of Server Communication
The interface specification defines the communication supported by the server.
Existing OPC specifications base on the communication types listed below. Each communication type has specific advantages and limitations.

(D)COM

The **Classic OPC** Specification defines a COM interface. OPC DA, HDA, AE servers are COM server and need to be accessed through (D)COM.
COM was introduced by Microsoft in 1995 and was extended to Distributed COM (DCOM) in 1997. Microsoft still supports DCOM and for local server access COM is still a good and efficient solution.
Remote server access is limited and the necessary DCOM configuration is tricky and the cause of most OPC difficulties.

.NET2 web services

The .NET2 web services (ASMX) eliminate the DCOM remote communication limitations and configuration issues. However, the communication is restricted to HTTP and the security options are limited. HTTP communication is slower than the other communication options but has enough performance for a wide range of applications.
The .NET2 based **OPC XML DA** is available since 2003 and is often used to remotely access OPC DA servers through an XML DA gateway server.

OPC UA

OPC UA communication is either TCP or HTTP. The security handling and the upper layer communication are UA specific to achieve multi-platform capability.
The OPC Foundation is working with member companies to make communication stacks and tools available to its members for different platforms, such as PLCs.

WCF

The Windows Communication Foundation (WCF) introduced by Microsoft with .NET3 extends the .NET2 web services with additional W3C standards and Microsoft specific features. The communication can be configured for a wide range of communication and security options.
The **OPC .NET 3 (Express Interface)** specification is based on WCF. With this the specification targets mainly Microsoft platforms and gains the advantage that Microsoft maintains the communication software and provides helpful tools.

Custom

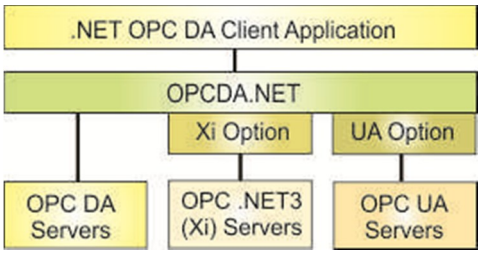
Tunneller products are being used to replace the DCOM communication. These products use vendor specific communication protocols based on TCP/IP. Tunneller modules from the same vendor have to be used on the server and client side.

Advosol offers a comprehensive set of client components, converter servers and server toolkits.

OPCDA.NET client component

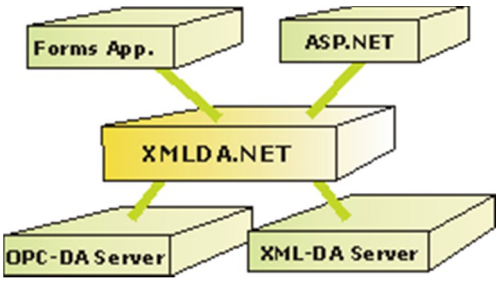
The Classic OPC DA (Data Access) interface is implemented in .NET classes. All OPC DA V2.05 and 3.0 features are supported. In a layered class structure the upper level classes provide server access features with a minimal amount of code.
OPCDA.NET can be combined with OPCHDA.NET and OPCAE.NET for HDA (Historical data access) and Alarm&Events

- **UA Option**
  UA servers can be accessed thru the same API as OPC DA servers. Existing OPCDA.NET based client applications can be upgraded to support UA without application code changes.
- **Xi Option**
  OPC .NET 3 (Xi) servers can be accessed thru the same API as OPC DA servers. Existing OPCDA.NET based client applications can be upgraded to support Xi without application code changes.



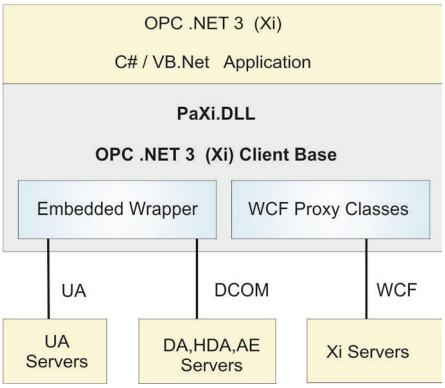
XMLDA.NET client component

Thru the XML DA API the client application can access:
- XML DA servers with .NET2 or .NET3 (WCF basicHttp) communication
- OPC DA V2.05 and 3.0 servers
XMLDA.NET is especially well suited for .NET2 web client applications.
XMLDA.NET is often used in combination with XDAGW-SS server-side XML DA gateway. The client application can access local OPC DA servers directly or remote server thru the XML DA gateway.



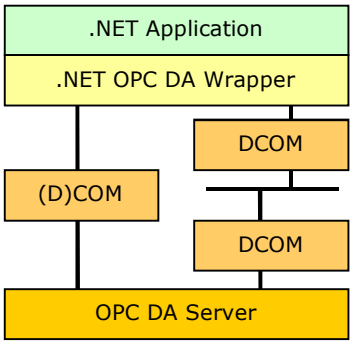
PaXi OPC DA .NET 3 Client Base

The PaXi Client Base implements the OPC .NET 3 specification and supports access to:
- OPC .NET 3 server thru WCF (Windows Communication Foundation) communication
- Classic OPC DA, HDA, AE server thru DCOM communication
- OPC UA servers thru the UA communication stack for .NET
The application can access any number of servers in any combination.

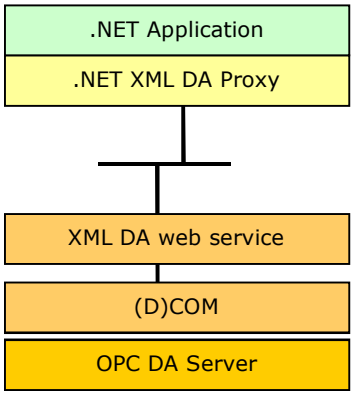
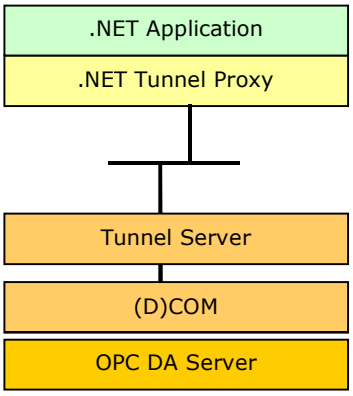


DCOM Issues

The DCOM configuration is the single most difficult and problematic issue in the deployment OPC applications.
It is therefore not surprising that some OPC .NET component products combine the .NET wrappers with features designed to reduce DCOM issues.
DCOM issues can best be reduced by eliminating DCOM network access. This is achieved by moving the OPC COM client to the same computer as the OPC server. Issues such as Firewall settings are eliminated. However, even on the same machine some DCOM access rights configuration is required.

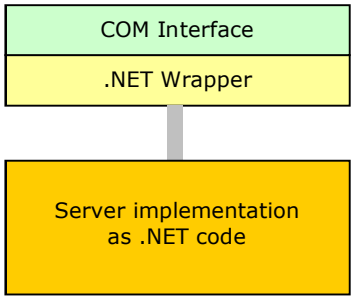
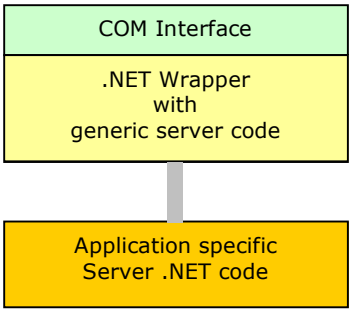


To **eliminate the DCOM network communication** in networked applications another communication mechanism has to be used between the .Net client component and a counterpart component on the OPC server machine.
Possibilities are:

Web service communication. XML DA or OPC .NET 3	Proprietary Communication protocol:
	
The OPC XML DA specification defines communication messages based on the SOAP web service standard. XML DA communication was often considered too slow but with .NET3 the performance can be comparable to a DCOM network communication. The client application uses the standardized XML DA or OPC .NET3 API	Tunneling products usually base on a TCP communication. Other implementations use .NET Remoting that can be configured for different communication protocols. All implementations are proprietary solutions that require client and server side components from the same vendor. The client application uses the vendor specific .NET application interface.

.NET based Classic OPC Servers

.NET is increasingly used for the development of classic OPC DA, AE, HDA servers. It offers advantages especially for servers that handle a database or have a TCP communication with the device.
As for clients, a .NET wrapper is required to convert the COM server calls into .NET method calls with the proper .Net data types.

	
The server functionality is implemented in .NET code. The .NET wrapper handles the call and data type conversions.	The OPC DA specification allows a large part of the server functionality to be implemented in application independent code. This code can be combined with the .NET wrapper and be hidden from the application implementation. The interface to the application specific server part can be simplified by omitting the features implemented in the generic server.

.NET Classic OPC servers can be implemented in different ways:

a. The COM Interface and the .NET Wrapper as a DLL and the application specific server as the COM server EXE file.

b. The COM Interface and the .NET Wrapper as the COM server EXE file. The application specific server part in a .NET assembly.
The **Advosol .NET Server Toolkits** (http://www.advosol.com/c-4-server-development-kits.aspx?) are implemented in this way. It has the advantage that the DLL with the application specific server functionality can be used with different server applications. The DANSrv OPC DA server plug-in DLL e.g. can be used with:
- The OPC DA generic server
- The XML DA generic web service (.NET2 or .NET3)
- The OPC .NET 3 generic web service for WCF
- The OPC UA generic server

Web Services OPC Servers

The newer OPC specifications XML DA, UA and OPC .NET3 are based on web services standards.
On Microsoft platforms web services are implemented on .NET and the .NET tools generate proxy classes for .NET clients. The web services can be accessed thru such proxy classes but client developers have to study the OPC specifications and make all necessary server calls in the proper sequence.
It is advisable to purchase client base components that handle the low level details necessary for specification compliance.