Express Interface

January 2010,   Copyright Advosol Inc.

This document is intended to give an overview of what the Express Interface (OPC Xi) is, how server and client applications can be developed and how the transition from OPC DA to OPC Xi can be realized.

Xi Basics

Express Interface (OPC Xi) is a communication interface specification designed for secure and reliable access to automation systems.
Xi provides a set of methods for accessing current data, historical data, events, and alarms. It has been designed for fast local communication and secure communication through firewalls.
Xi defines a Service Oriented Architecture (SOA) that is based on MMS (Manufacturing Messaging Service) and WCF (Windows Communication Foundation).
The main design goals were:

- Security – for secure communications including access through firewalls
- Simplicity - to make servers and clients easy to implement, configure, and deploy
- Robustness – for reliable communications with error recovery
- Plug-and-Play – for automatic discovery of servers and their communications capabilities
- Backward compatibility - for accessing OPC DA, AE and HDA servers using a single .NET interface
- Protocol transparency – for efficient client/server communications when both the client and server are on the same platform or within the same local area network, and when the client is located on external networks, including the Internet (see Figure below).

WCF Communication Base

The Windows Communication Foundation (WCF) offers a rich set of communication features for local and remote communication. It is based on a number of web services standards.
To understand the features and design goals of WCF it's best to look at the **history of the communication interfaces** offered in Windows systems.

COM	The Component Object Model is a binary-interface standard for software componentry introduced by Microsoft in 1993, as a successor of OLE. COM is still widely used for integrating components into an application, especially for in-process communication.
DCOM	The Distributed Component Object Model is an extension of COM for communication between processes and over networks. DCOM was introduced in 1996 and in 1997 renamed to Active-X. The DCOM name is however still being used. DCOM competed with CORBA for acceptance as a general communication standard but neither succeeded, mainly because of difficulties with communication through firewalls. HTTP in combination with web browsers won out. However, DCOM is still widely used for communication between Windows applications and is the base for the classic OPC specifications.
Web Services	With .NET Microsoft added web services support based on the XML and SOAP standards. This was a major step toward open communication but, the .NET web services are based on http communication and lack performance and security features for general use.
.NET Remoting	.NET Remoting was introduced with .NET1 for binary-optimized interprocess communication. It addresses performance and security issues with support for multiple communication protocols, HTTP, TCP and named Pipes for local communication.
WCF	With .NET3 Microsoft combined the Web Services and .NET Remoting APIs into the Windows Communication Foundation (WCF) and based it fully on WS standards.

	WCF is designed in accordance with Service oriented architecture principles to support Distributed computing where services are consumed by consumers. Clients can consume multiple services and services can be consumed by multiple clients. Services typically have a WSDL interface which any WCF client can use to consume the service, irrespective of which platform the service is hosted on. WCF implements many advanced web services (WS) standards such as WS-Addressing, WS-ReliableMessaging and WS-Security.

OPC Xi Communication Interface

Xi specifies a set of WCF contracts as .NET interfaces and as a WSDL file.
Xi servers reference the Xi Interface and implement the contracts.
Xi client applications can be built in Visual Studio by referencing the Xi interface assembly or the WSDL file.

XI is designed for high performance secure communication.
The functionality is divided into multiple interfaces. Each interface (WCF contract) can be accessed through associated WCF endpoints. The endpoints can be configured with the communication settings appropriate for the interface and the application.

IServerDiscovery	This interface is used to locate Xi servers on the network and retrieve information of individual servers. There is always an endpoint with the basicHttp binding for this interface.
IResourceDiscovery	This interface provides methods manage the server connection, discover objects/events and manage lists for access to objects, alarms and events. The IResourceDiscovery methods are mostly used at server startup and performance is not a main issue. However, security may be because meaningful names have to be communicated. The endpoints for this interface are typically configured with secure communication bindings. The server can configure multiple endpoints and the client can use the best fitting one, e.g. a Name Pipe binding for access to a local server or an encrypted http binding for access through firewalls to an Internet hosted server.
IRead	The methods of IRead interface access the server using handles retrieved in IResourceDiscovery methods. The communicated is hard to interpret even it could be eavesdroped. Application are able to reduce the communication security in order to boost performance.
IWrite	The write methods are in a separate interface to make it easier for the server to restrict write access to certain clients.
ICallback	The methods of the ICallback interface manage and handle subscriptions and data change callbacks from the Xi server to the client. WCF supports dual communication only in certain bindings. E.g. it’s not available with basicHttp.
IPoll	This interface supports the subscription handling through a poll mechanism. This is possible with all communication bindings. The Xi client can determine the available communication bindings and use either callbacks or polling.

Xi Communication Configuration

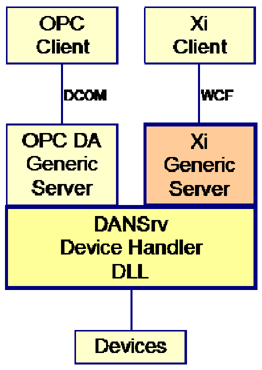
The Xi communication configuration is basically the WCF endpoint configuration.
Xi has communication configuration settings only on the server side.
Xi clients retrieve the configuration from the server through IServerDiscovery and WCF metadata exchange. These endpoints have standardized communication settings and deliver the URIs and settings of all server endpoints.
The server can configure multiple endpoints for each interface. This allows the client application to choose the best fitted communication binding. A client that is located on the same machine as the server can use an endpoint with a Named Pipe binding, while for server access though the Internet the client may use a secure http binding.

Xi Server Structure

A typical Xi server design has the layers:

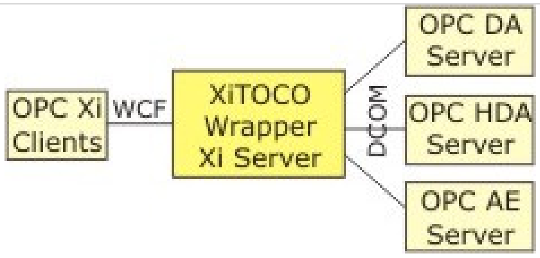
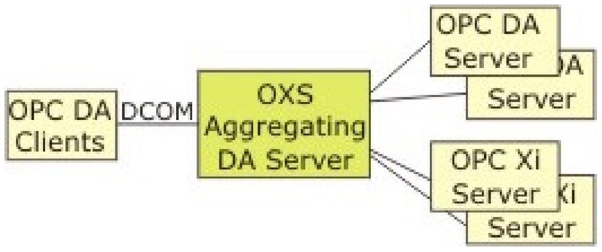
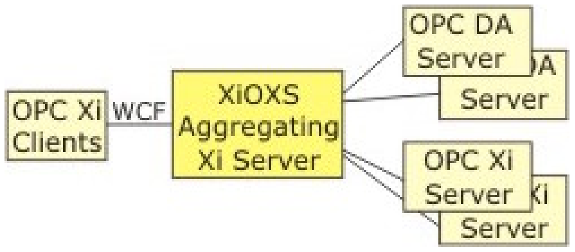
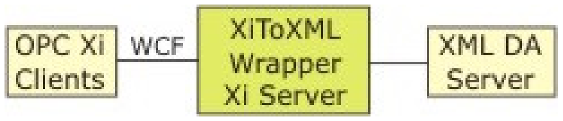
Contracts	The Xi server contracts define the Xi interface. All Xi servers must implement the exact same interfaces. The contracts are provided as a .NET 3.5 assembly with the specification and as a Visual Studio 2008 project with the Xi reference code.
Server Base	The server base code implements the Xi contracts and handles application independent features. A base layer implementation is provided Visual Studio 2008 project with the Xi reference code.
Implementation	This is the application specific part of the Xi server.

Xi Server Development

Based on the Xi Reference Code	The 'Server Base' part of the reference code is publicly available and is designed to handle the application independent server features and enhance interoperability. The Implementation layer is provided as C# code that wraps to a OPC DA, HDA and AE server. This code is available only to OPC Foundation members. This layer needs either to be modified or replaced with code that handles the application specific server functionality.
With the Advosol DANSrv Toolkit	The Advosol DANSrv toolkit is a .NET toolkit for the development of OPC DA servers. The OPC DA DCOM interface is handled in the DANSrv.exe generic OPC DA server. The application specific server part is in a .NET assembly DLL. The DLL interface is rather simple but flexible enough for complex servers. For simple servers only a few methods need to be implemented. The **XiPLUS** generic Xi server can replace the generic OPC DA server. The same plugin DLL can be used with both generic servers. The main advantages are: • Quick and easy to implement • Without additional development effort the user gets an Xi and an OPC DA server. • No DCOM and .NET Wrapping with the generic Xi Server  OPC Client Xi Client DCOM WCF OPC DA Generic Server Xi Generic Server DANSrv Device Handler DLL Devices

OPC Xi Wrapper Servers

Wrapper servers make classic OPC servers accessible from Xi client applications or enable classic OPC client application to access Xi servers.
The use of wrappers servers allows a quick transition to OPC Xi.
The wrapper server approach makes sense if the goal is to make an existing OPC DA server remotely accessible. If the goal is to eliminate the often troublesome DCOM communication, then the Xi wrapper can only partially improve the situation because DCOM is still being used between the Xi wrapper server and the OPC DA server.

Xi Reference Server	The Xi reference server implementation is a wrapper server for OPC DA, AE and HDA. It is available as source code to OPC Foundation members. Only limited support is available.
Advosol Wrapper Server Products	**XiTOCO** The XiTOCO server wraps classic OPC servers DA, HDA and AE server into an Xi server. Only a simple configuration is required to specify the OPC servers to be wrapped. XiTOCO works in 32/64bit mode either self hosted as a console application or a Windows Service or IIS hosted.  OPC Xi Clients WCF XiTOCO Wrapper Xi Server DCOM OPC DA Server OPC HDA Server OPC AE Server **OXS** The OXS Exchange server aggregates multiple OPC DA or Xi servers into its address space and can be configured to exchange data between the servers. Configured as an OPC DA server the Exchange server enables OPC DA clients to access Xi servers. Configured as an Xi server the Exchange server enables Xi clients to access OPC DA servers.  OPC DA Clients DCOM OXS Aggregating DA Server OPC DA Server OPC Xi Server **XiOXS** The upgraded Advosol OPC Exchange Server supports access to Xi servers. OPC DA client applications can access Xi servers through the OXS OPC DA V2/V3 compliant OPC DA server. OXS can also act as an XML DA server, allowing XML DA clients access to Xi servers.  OPC Xi Clients WCF XiOXS Aggregating Xi Server OPC DA Server OPC Xi Server **XiToXML** The XML DA Wrapper Xi Server makes XML DA servers accessible from Xi clients. Existing XML DA servers can be integrated into an Xi configuration and take advantage of the communication options supported by Xi.  OPC Xi Clients WCF XiToXML Wrapper Xi Server XML DA Server

Xi Client Development

Xi client are regular WCF client applications and be developed as such. Different approaches are possible:

- **Visual Studio generated WCF proxy classes**

Visual Studio can create the client proxy classes from a running Xi server, the Xi Contract assembly or the Xi WSDL file.
The generated code doesn't contain all the constant definitions, enumerators, constructors and helper methods included in the contract classes.
It's not recommended to choose this approach.

- **Based on the Xi Generic Client Reference code**

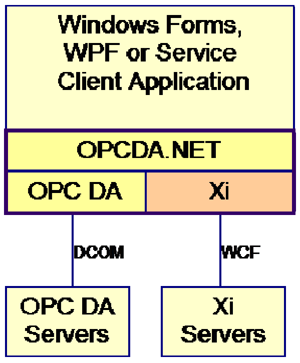
The Xi reference code includes generic client base classes with methods for the server access.
These classes currently support only synchronous server access.

- **Based on the Advosol PaXi Component**

The PaXi client component provides classes with methods for synchronous and asynchronous server access. It supports all Xi specified features.
Asynchronous server access improves the quality of user interface applications. The application doesn't become unresponsive due to slow communication (e.g. through the Internet) or timeouts in error conditions. The performance can be improved with parallel call execution.
Visual Studio Designer components reduce the coding enormously. The required features are selected in Visual Studio Designer dialogs instead of coding all necessary server calls.

- **With the DANW-Xi Option for the Advosol OPCDA.NET Client Component**

The Xi option for OPCDA.NET is an Xi wrapper that runs within the client process.
Existing OPCDA.NET based applications can be upgraded to Xi support by adding the Xi option. The application code doesn't have to be modified.



OPCDA.NET with DANW-Xi also offers important advantages for new applications. OPC DA and OPC Xi servers can be accessed without an external wrapper server. For Xi server access DCOM communication and .NET wrapping are completely eliminated.
Multiple OPC DA and OPC Xi servers can simultaneously be accessed.
The upper layer OPCDA.NET classes like DataBind or background server access work also with Xi servers.

Transition from OPC DA to Xi

Xi is specified with backward compatibility in mind. Different wrappers are available for conversions in both directions.

- The Xi reference server and the Advosol XiTOCO Wrapper Server
  provides Xi clients access to OPC DA, AE and HDA servers

- The Advosol OXS Exchange Server
  provides OPC DA and XML DA clients access to Xi servers

- The Advosol XiOXS Exchange Server
  provides OPC Xi clients access to multiple OPC DA servers

- The Advosol XML DA Wrapper Xi Server
  provides Xi clients access to XML DA servers.

- The Advosol DANW-Xi Option for OPCDA.NET
  is a Xi wrapper built into the OPC DA client process. It provides .Net client applications access to OPC DA and Xi servers through the same API.

- The Advosol XiPLUS Server Toolkit
  enables server vendors to provide their servers as OPC DA and Xi servers with a single server development.