OPC and Its strategies for Redundancy

PAVAN KUMAR PENDLI, VADIM GORBATCHEV, MICHAEL SCHWARZ, JOSEF BÖRCSÖK
Computer architecture & System programming
University of Kassel
Wilhelmshöher Allee 73 34121 Kassel
GERMANY
http://www.rs.eecs.uni-kassel.de

Abstract: - In the earlier days, providing seamless connectivity for communication and exchanging of information between various process control hardware devices and software applications has become a challenging task and advantageous. The application developers were need to develop different drivers for each device and more over much greater effort was to be placed on the management of the network architecture. A CORBA-based standard DAIS (Data Acquisition for Industrial Systems) and Microsoft’s windows platform-based OPC/DCOM (OLE for Process Control / Distributed Component Object Module) are the two major middleware technologies developed to solve the above problems. Because of the predominance of Microsoft’s windows its platform based OPC/DCOM is widely used and accepted.

This paper presents an introduction to OPC, OPC specifications, OPC architectures and OPC strategies for redundancy.

Key-Words: - Middleware Technologies, DAIS, OPC, OPC Specification, OPC Architecture, Redundancy.

1 Introduction
In order to adapt to the growing need of compliance, a greater interoperability was needed between the automation/control applications, business/office applications and field-bus /device. Fibre optics, Ethernet and TCP/IP were not just sufficient, they require additional application layer protocols like SNMP (Simple Network Management Protocol a layer 7 protocol in OSI (Open Systems Interconnection) reference model) for the exchange of management information between network devices to actually carry out the tasks over the network. But SNMP was considered to be insufficient for the HMI/SCADA (Human Machine Interface/Supervisory Control and Data Acquisition) control systems used widely in the control and automation industry. The reason is that network management software compatible with SNMP is hard to learn and hard to use, and as a result did not gain widespread acceptance by the control engineers [1]. To achieve interoperability, a group of software developers and the members from the leading automation industries with the support from Microsoft decided to develop common interfaces between them. OPC was the result of their work and they have named their organisation as “OPC Foundation”. OPC is the latest trend widely accepted as it is based on the principles adopted widely and applied Microsoft windows integration standards and it is also the mostly used technology in the process control and factory automation industries [2]. OPC is based on the Microsoft’s OLE (Object Linking and Embedding), COM (Component Object Module) and DCOM technologies.

Originally named ”OLE for Process Control” after Microsoft’s Object Linking and Embedding technology, it is now known simply as ”OPC,” which stands for Open Process Control or Open Connectivity. According to a recent survey (ARC Advisory Group) [3] nearly 95 percent of the companies in the field of process control and automation for programmable logic controllers (PLCs), distributed control systems (DCSs), supervisory control and data acquisition systems (SCADAs), and manufacturing execution systems (MESs) plan to use OPC an interoperable communications standard interface for their operations. The flow of this paper is as follows: section 2 gives different specifications developed by “OPC foundation” organisation to design the interfaces accordingly by the developer of OPC server. The architecture of OPC is explained in the next sections with an explanation of OPC DA and AE client/server architecture as an example, OPC strategies for redundancy comes next and the last section deals with benefits of using OPC technology.

2 OPC Specifications
OPC specifications define some common interfaces as shown in the figure Fig.1 below for specific purposes which allow applications/OPC clients or field-bus/devices to communicate and exchange information between them. The applications need only implement one specific OPC driver or OPC interface for each
specific purpose. Since OPC interfaces are based on technologies such as Microsoft windows COM/DCOM and ActiveX, they have the advantage of being easy to learn and implement, and does not require a lot of modifications to the existing system.

In Fig.1 application are software applications or OPC clients and the components are considered to be fieldbus or devices.

![Fig.1 OPC Specification](image)

The different standard specifications described by OPC foundation are discussed below.

2.1 Data Access (DA) specification
This specification is the first one released by OPC foundation in the year 1996, mainly designed for the purpose of real-time transferring of the data between components and applications, since then the other specifications have been released for other specific purposes [4].

2.2 Historical Data Access (HDA) Specification
This specification is released for accessing the historical data from Historical engines at the OPC servers, which must be distributed to applications or OPC clients, therefore historical information is considered to be another type of data. There are different types of historical servers supported by this specification, depending on the types of data like simple trend data servers for simple raw data storage containing less information like time, value and quality and the complex data compression and analysis servers for compression as well as raw data storage containing more attributes like average values, minimum values, maximum values and also some group of functions [4].

2.3 Alarm and Event (AE) Specification
This specification is mainly for the purpose of monitoring and acknowledging of Alarm and Event notifications between components and applications, the notifications include safety limits of components, event detection, and abnormal situations and these require an immediate attention. This specification not only just deals with detecting and reporting of alarms and events but also performs advanced sorting or filtering operations for the desired alarms and events by the applications. This specification complements the OPC Data Access and the OPC Historical Data Access specifications but it is different from them [4].

2.4 XML Data Access (XML-DA) Specification
This specification is for the purpose of exchanging structured form of information between components and applications in the form of XML. XML is an extensible mark-up language which is readily available across wide range of platforms and also facilitates the exchange of data across the internet using web technologies, this enables further OPC’s goal of interoperability [4].

2.5 Security Specification
This specification is mainly for the purpose of security issues, as they play a vital role in the field of communication. It focuses on client identification that is the exchanging of trusted credentials to be used for access authorization decisions by the OPC Server. It does not address which objects are to be secured, but leaves this matter to the OPC server implementers.

2.6 Complex Data Specification
This specification is for the purpose of more complicated data types such as binary structures and XML documents. It accompanies both DA and XML-DA specifications described above to achieve its goal.

2.7 Data exchange (DX) Specification
This specification enables the transfer of data from a source OPC DA or DX server item to a target OPC DX server item i.e. a server-to-server communication as shown in the Fig.2 below.

![Fig.2 DX Specification](image)

OPC clients use OPC DA interfaces to browse OPC DA and OPC DX servers for source and target items. Target items are always resident in the OPC DX server. Source items may be resident in OPC DA servers or OPC DX servers, including the target OPC DX server.
2.8 Unified Architecture (UA) Specification

This specification is the latest draft release from OPC foundation for the purpose of enabling communication between different operating systems and also across firewalls and it is also based on XML and web technologies similar to XML Data Access specification. This specification is organised as a multi-part specification, as shown in the Fig.3 below.

![Fig.3 UA Specification](image)

It consists of two different parts with core specification part and access type specification part. The core specification part consists of seven parts as shown in Fig.3 which specify the core capabilities of OPC Unified architecture and they define the structure of the OPC address space and the services that operate on it. Parts 8 through 13 apply these core capabilities to specific types of access previously addressed by separate OPC specifications, such as Data Access (DA), Alarms and Events (AE), and Historical Data Access (HDA) [5].

All the OPC specifications described above defines only the way for communication between components and applications but do not define what actually is to be included for communication, like “The Data Access specification is used for passing values, such as a flow rate or temperature, while the Alarm and Event specification is used for passing alarm and event notifications. OPC specifications do not dictate which temperature or pressure values to be sent using an OPC DA interface and which alarm and events to be sent by AE interface and so on, what actually to be included for communication is defined by the user of the interface while designing OPC server.

2.9 Web technologies

The term web technologies used in discussing the OPC specifications include Simple Object Access Protocol (SOAP), XML (Extensible Markup Language), Web Services Definition Language (WSDL), The Universal Description, Discovery and Integration (UDDI) and the hypertext transfer protocol (HTTP).

3 OPC Server Architecture

OPC server architecture depends on the purpose it is designed for, OPC as DA (Data Access) contains the architecture of DA and for AE (Alarm and Event) contains the architecture of AE and for both DA and AE contains both the architectures, i.e. the architecture is dependent on the type of purpose.

3.1 OPC Architecture for DA

In Fig.4 shown below OPC DA server’s reads the data from the devices. Devices are the PLCs, Sensors Controllers e.t.c. The data is stored into the cache of the OPC DA server which is there held to be made available to OPC clients via OPC Interface. Communication between client and server is n-to-m i.e. an OPC client can simultaneously interact with several OPC servers and several OPC clients can have the access to the same OPC server [6]. Each OPC DA server can simultaneously provide data to any number of OPC clients and multiple clients can at the same moment access the data from any DA server leading to a robust method of communication.

![Fig.4 Data Access Server](image)

3.1.1 Client / Server Architecture for DA

OPC server for Data Access is as shown in Fig.5, it consists of several objects: the server, the group, and the item.

OPC groups and OPC item:

OPC server for DA manages the data to be processed, for this it has to implement the required protocol to get that data from devices. OPC DA server makes this process data available via OPC items. An OPC DA server creates OPC items on behalf of an OPC client. The client’s OPC items are organized in OPC groups, OPC clients can only access their OPC items through their respective OPC groups [6].
Fig. 5 below shows how OPC clients can access process data managed by an OPC DA server.

![OPC DA Client / Server Architecture](image)

Each OPC item is associated with parameters itemID, value, quality, and timestamp. The itemID is the fully qualified definition of a data item in the server and the syntax of the itemID is server specific. The parameter value is the value of the data, quality is set good when the data is received from devices and timestamp is the time the OPC server updates the parameters value and quality.

3.2 Client / Server Architecture for AE

![OPC AE Client / Server Architecture](image)

OPC client / server architecture for Alarm and Event is as shown in Fig. 6, it consists of several objects: the server, area space and event space. With area and event space clients can filter the event subscriptions from the OPC server by specifying the process areas.

Within an OPC AE server an event is a detectable occurrence i.e. failure of a device, excess of a limit value or operator intervention and alarm is an abnormal condition that has to be dealt immediately and can be acknowledged after dealing with it by the OPC client.

4 OPC strategies for Redundancy

For industrial automation applications, redundancy plays an important role to increase the efficiency and reliability of the system. The need for redundancy arises because of the Object based failures and Link based failures shown in Fig. 7 which results in loss of information. Object based failures are considered to be the failures that occur when the actual link between OPC client and the target OPC server breaks down and the objects that have to be created at the server have failed. Link based failures are the failures that occur when the connection between the OPC server and the client is perfectly intact but the physical link to the underlying device or system is broken [7].

![Object and Link based failures](image)

4.1 Strategies for Redundancy:

OPC servers provide redundancy by increasing the reliability and availability of the information from the devices by allowing multiple OPC servers to be configured into redundant pairs with active and backup. OPC client can only select the OPC server which is to be active making the other as backup. Each redundant pair seamlessly appears as a single OPC server to the OPC client application.

Strategies for redundancy can be classified into two types:

1. Client controlled
2. Server controlled

4.1.1 Client Controlled:

Client controlled strategy is the one which is already implemented and tools were also provided by some companies which provide redundancy to the clients by installing the redundancy software or implementing the functionalities of the redundancy software at the client side as shown in Fig. 8 below. In this case client or the tools at the client side defines how the subscriptions of objects are configured and synchronized across the active OPC server 1 or 2 or both in the redundant pair in case of more crucial information and also the client defines how the switch-over is to be controlled when the active server fails to the backup server in the redundant pair.
4.1.2 Server Controlled:

Server controlled strategy is the one suggested for this paper with different cases providing more redundancy with more control at the server side which will be discussed in detail. In this strategy under normal conditions, active OPC server 1 or 2 selected by the OPC client in the redundant pair shown in Fig.9 identifies and communicates with the other server treating the other server as normal client to provide automatic copying and to control the synchronisation of the configured client subscription objects, making it as backup server. The switch over in the redundant pair from the active server to backup server is controlled by the OPC client.

Different cases showing server controlled redundancy:

Case 1 and 2:
In case 1 and 2 shown in Fig.10, OPC server 1 is active and OPC server 2 acts as backup. OPC server 2 is synchronized with the configured client subscription objects from the OPC server 1. When the connection between the device and the active OPC server 1 fails then the client receives the information of the device through OPC server 1 via OPC server 2 as shown in figure below in steps 1, 2 and 3. As the client sees that OPC server 1 is active it does not make switching.

Fig.10 a) Case 1 and b) Case 2.

In case 2 the connection between the active OPC server 1 and backup OPC server 2 fails then the client receives the information from the device through OPC server 1 in steps 1 and 2. Here the synchronisation of the configured client subscription objects at the backup server fails until there is a regain in connection between OPC server 1 and 2. The client will be informed of the regain of connection by an alarm or event and then the servers can be synchronised again with the current configuration and if any history data is needed an OPC HDA (Historical Data Access) server can be used.

Case 3 and 4:
In case 3 shown in Fig.11 below the connection between the active OPC server 1 and OPC client fails then the client switches to the OPC server 2 making it as active server and receives the information as shown in steps 1 and 2. Here OPC server 2 synchronises the configured client subscription objects to OPC server 1 making it as backup server.

Fig.11 Case 3 and 4.
In case 4 the connection between the active OPC server 1 and the backup OPC server 2 fails and also the connection of the client to the device via active OPC server 1 fails then OPC client switches to OPC server 2 making it as active and gets the information in steps 1 and 2 as shown. Again the client can be informed here when there is a regain in connection as discussed in case 2.

Case 5 and 6:

These are the cases shown in Fig.12 below where OPC client switches between OPC server 1 and OPC server 2 to see if any server is available to set as active and to receive information of the device from the active server, when the connection fails from the device to both servers as shown in case 5 and also when there is no connection between the servers and the device as shown in case 6.

5 Benefits and Conclusions

OPC provides benefits to both vendors and end users, vendors are OPC clients and end users are the users of client applications. They are benefited by using single interface to variety of process control devices and also have the chance of selecting the OPC client best suits for their requirements. OPC allows "plug-and-play" by which OPC devices can be connected together at anytime irrespective of their manufacturers and made immediately work with OPC clients by just plugging the appropriate OPC server for the new device and without shutting down the existing systems. This reduces the time needed for the installation of devices and also system configuration time. OPC reduces the integration costs and risks, in the management of the manufacturing of control devices as there is no need to develop different drivers for each device and also no risk in further developing the driver as it is based on OPC components rather than custom drivers, in terms of network architecture installation as common interface is being used and for back-end business applications by having the choice of selecting the desired application. OPC provides a high degree of interoperability as discussed in this paper and also provides more redundancy with server controlled strategy as explained with different cases providing more consistency with no loss of information even with the shutdown of the active OPC server.

References: