



OPC UA: An End-User's Perspective

OPC Training Institute

OPC Training Institute
Tel: 1-780-784-4444 | Fax: 1-780-784-4445
Web: www.opcti.com | Email: info@opcti.com

Copyright © 2008 OPC Training Institute (OPCTI). All rights reserved. The information contained in this document is proprietary to OPCTI. No part of this document may be reproduced, stored in a retrieval system, translated, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without prior written permission from OPCTI.

OPC UA (Unified Architecture) represents the OPC Foundation's most recent set of specifications for Process Control and Automation system interconnectivity. This paper explains OPC UA from the perspective of the organization that will benefit from the connectivity, in other words: the End User.

The first form of OPC relied on DCOM for its data transportation, which was very powerful and versatile, but posed a problem for those who did not understand how to configure DCOM. Instead of DCOM, OPC UA relies on Web Services for its data transportation. OPC UA also uses objects to help with data description. Even though these are major additions and modification to OPC, OPC UA will be backwards compatible with older products through the use of wrappers. All this will ensure that OPC UA will be even better suited to penetrate the entire plant enterprise. Of course, with all the new connectivity that OPC UA offers, the new challenge will be system security.

OPC Overview

OPC is an industrial communication standard that enables manufacturers to use data to optimize production, make operation decisions quickly and generate reports. OPC enables plants to automate the transfer of data from a control system (PLC, DCS, analyzer, etc) to an industrial software application (HMI, Historian, Production system, Management system, etc). OPC is typically found in Level 3 networks and higher. Thus, OPC transfers process control data between the Control (Level 2) network and the Operations/Manufacturing (Level 3) network. It also exchanges data between the Operations/Manufacturing network and the Business (Level 4) network. In essence, OPC is the Modbus of the new century. It is not a replacement for low-level communication standards such as 4-20mA, Hart, Profibus, or Foundation Fieldbus. Rather, organizations use OPC in high-level communication.

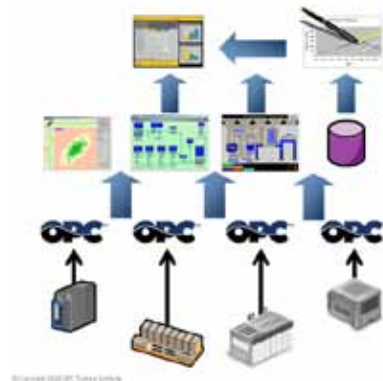


Image 1: OPC communication enables applications to interoperate and simplifies system architecture.

Note: OPC is no longer an acronym. When OPC first released in 1996 it was an acronym for OLE for Process Control, and was restricted to the Windows operating system. OPC is now available on other operating systems and enjoys significant adoption outside of process control. So, the original name (OLE for Process Control) is no longer appropriate and OPC changed from an acronym to a word. Thus, OPC no longer stands for anything (OPC is just OPC).

OPC communication started with DCOM

When OPC first released in 1996, with OPC Data Access 1.0 (OPC DA 1.0) specification, it used Microsoft's DCOM as the data transportation mechanism. Data moved between OPC applications on different computers using DCOM. At the time, DCOM was an outstanding choice because it provided a working communication infrastructure complete with all the necessary security services (authentication, authorization, and encryption). Thousands of vendors were already using DCOM because it was a relatively versatile API (Application Programming Interface). DCOM was a clear winner at the time, but while it provided a reliable communication backbone for OPC, it did have several challenges.

First, DCOM configuration eludes Automation personnel who do not take time to learn it. DCOM is actually very predictable and is not difficult to configure. While there are training classes that explain DCOM configuration in detail, most people do not take the time to learn it and so DCOM's behavior frustrates them. Consequently, Automation personnel needlessly experience problems when connecting two computers and configuring firewalls. Nevertheless, knowledgeable users can easily configure DCOM in a matter of minutes.

Second, many programmers assume that network communication will occur without any data loss. This assumption leads them to create products that are highly susceptible to data loss and communication timeouts. As a result, End-Users might sometimes experience a delay in application responses and complain to their vendors. However, since the programmers fail to understand DCOM's behavior, they often incorrectly blame DCOM for poor application behavior, which further promotes the false myth that DCOM is unreliable. Again, informed programmers are easily able to compensate for data loss and are able to make DCOM work reliably, and in a way that End-Users would expect.

The third problem is DCOM does not work through NAT (Network Address Translation). Thus, DCOM does not work in the rare cases that communication must occur between two private networks that are separated by a public network. Such is the case when two plants attempt communication over the Internet. NAT is



Image 2: OPC initially relied on DCOM for data transportation. DCOM is a highly secure and fast API from Microsoft that provides applications with services for authentication, authorization, and encryption.

sometimes used inside industrial facilities, but this is often unnecessary as a firewall would suffice.

The fourth problem is DCOM is proprietary to Microsoft. This makes OPC difficult (to impossible) for vendors to port to non-Windows operating systems. While some vendors are able to embed Windows directly on their own controller (PLC, DCS, analyzer, etc) hardware, others are unable to do this. Also, companies that use non-Windows operating systems (UNIX/Linux, VMS, etc) are having a difficult time importing OPC data into their applications.

OPC UA uses Web Services

OPC UA uses Web Services instead of DCOM for data transportation. This is the change that most End-Users will notice immediately. Two of the biggest advantages for Web Services are ease of communication between networks and independence from specific operating systems. The challenge for the plant itself will be the implementation of security to keep the data safe.

Perhaps the biggest technical advantage for Web Services is that they enable OPC to communicate over a single port using a protocol that most firewalls will allow to pass by default. This should make it easier for Integrators to setup a system for communication between networks. Many firewalls are already configured to let web traffic pass across port 80. This will make it easier for IT to open the ports necessary to implement OPC communication. Previously, DCOM required multiple ports to establish communication. While this was possible to configure, a significant portion of automation personnel did not take the time to learn how to do it. Nevertheless, opening port 80 opens communication for a plethora of applications (not just those that are needed for Operations), so emphasis on security will be required immediately.

In addition, Web Services are not bound to any specific operating system. Thus, vendors will have an easier time implementing OPC Servers on their automation hardware and non-Windows operating systems. Vendors are already working on PLCs that include an embedded native OPC Server that does not require an external computer. However, this implementation might not be as simple as it seems because an automation application (HMI, Historian, APC, etc) typically requires a PC anyway. Nevertheless, it would be possible to have a PLC communicate with a software application using OPC without requiring an intermediate computer that uses Windows.

OPC UA uses an Object Oriented Data Model

Classic OPC has a fairly simple data model. Each of the OPC specifications handles a different aspect of the data. For example, the OPC DA (Data Access) specification communicates real-time values, the OPC HDA (Historical Data Access) specification communicates archived values, the OPC A&E (Alarms and Events) specification communicates various process and system events (such as a temperature that exceeds a pre-specified limit), and so on. In addition, Classic OPC implements each specification separately; essentially in a different executable. Thus, it is time-consuming for people to match item names with real-time data and item names with historical data. Even worse, automated applications may not be able to do it at all.

OPC UA provides a unified data model. Thus, when an application uses OPC UA to send a temperature reading, the receiver is able to retrieve the real-time value, any associated historical values, and even alarms and events. All this data is available from pointing at a single OPC item. The OPC Server is able to associate all the data together so that the OPC Client does not need to redo the association work. For example, in DCOM-based OPC, an End-User who is interested in a pressure reading would have had to point to the OPC DA server to look at the real-time value. Then they would have to point to an OPC HDA server to trend the pressure over the past shift. If they wanted to take a look at associated events, they would have to point to the OPC A&E server. But with OPC UA, the End-User can simply point to a pressure reading, view its real-time value, look at the past shift's trend (historical data), and view all the associated events by connecting to a single OPC UA server.

OPC UA also provides the ability to create more complex objects. For example, one could create a pump that is composed of various temperature, level, pressure, flow, and vibration readings. Included would be the history of all values as well as a picture of the pump. One could even associate P&ID schematic diagrams and maintenance orders. This presents a powerful mechanism for integrators from various companies to share data without having to recreate it in their different proprietary software applications.

Improving Existing Specifications

As OPC evolved over the years, the OPC Foundation provided constant updates and improvements to the specifications. OPC UA continues this tradition. After consulting end-users, integrators, and vendors, the OPC Foundation decided on various additions to the specifications to handle the most common challenges. OPC UA includes mechanisms to quickly inform users of broken communication, identify lost data, and even provide for redundancy.

OPC UA uses a poll-report-by-exception mechanism. Thus, the OPC Client polls the OPC Server for changes. The server then responds with any data changes. A failure to respond would immediately tell the OPC Client that the communication is no longer active. In addition, updates can come as quickly as the polling itself. However, unlike common protocols that must poll each point individually and consume precious bandwidth, OPC UA enables the OPC Server to respond with any data that changed. Thus a single efficient poll can bring back a large amount of data that includes all the changes in the process as well as the health of the OPC Server itself. By contrast, before OPC UA, DCOM communication sent all changes to the OPC Client by exception. Thus, an OPC Client did not have to poll the OPC Server periodically. While this was efficient, many programmers overlooked the possibility that no updates would be received when communication breaks. As a result, the OPC Client would wait for updates that would never arrive. Various companies overcame these difficulties, but some did not and blamed DCOM instead.

OPC UA also enables an easier implementation of redundancy. OPC UA Servers can update a set of clients. By contrast, DCOM-based OPC Servers could only update OPC Clients that explicitly subscribed to the data. As well, since the OPC Client can easily tell when communication with an OPC Server fails (as above), the OPC Client can now quickly failover to a standby OPC Server. In DCOM-based implementations, most vendors relied on third party OPC redundancy applications that cost them additional funds.

Backwards Compatibility and Tunneling

The OPC Foundation has promised to supply the industry with two simple software applications that will enable people to quickly convert their DCOM-based OPC products to OPC UA. These software applications are called "wrappers." Wrappers will ensure that any new OPC UA product will communicate with an existing DCOM-based OPC product. As a result, there is no need to contemplate whether or not one should wait for OPC UA products. It is easy to implement DCOM-based OPC products today and rest assured that future OPC UA products will communicate with the old software.

Two wrappers will be available: one for OPC Clients and the other for OPC Servers. The first

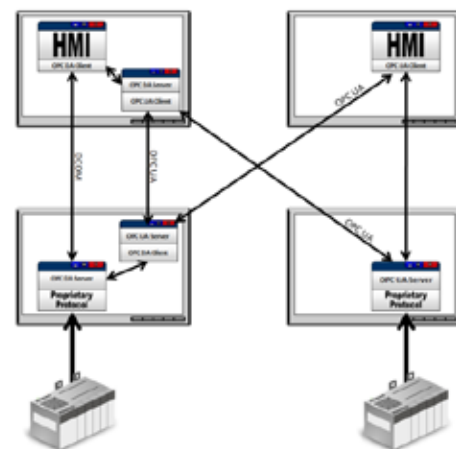


Image 3: OPC UA wrappers will enable legacy DCOM-based OPC products to communicate with new OPC UA products. The wrappers will also make it possible for two DCOM based applications to eliminate DCOM from their communication.

wrapper will convert a DCOM-based OPC Server to an OPC UA Server. Thus, an OPC UA Client will be able to connect to the existing DCOM-based OPC Server without any changes. The second wrapper will convert a DCOM-based OPC Client to an OPC UA Client. So an existing DCOM-based OPC Client application (such as an HMI) will be able to communicate with an OPC UA Server that could be purchased a year from now. Using wrappers, OPC is sure to ease the transition from the old to the new technology.

Wrappers will tunnel OPC to places where DCOM-based OPC can't penetrate on its own. For example, when an OPC Client and Server are separated by NAT (Network Address Translation), DCOM will not be able to make the connection. However, by converting the DCOM-based call to OPC UA at the source, and converting it back from OPC UA to DCOM at the destination, the call will transport the required data. Tunneling will likely be the first form of OPC UA implementation as OPC UA products begin to emerge.

Shop Floor to Top Floor: OPC to the Enterprise

OPC UA introduces an object model to industrial data, and Web Services will enable the OPC applications to transport the data across firewalls, networks, and the Internet. A variety of applications will be able to supply the enterprise with data. An HMI will be able to pass equipment events to the Maintenance system. The Historian will be able to pass calculations to various engineering systems. As well, inventory management systems will

be easily able to obtain production figures directly from automation equipment.

Plant-floor data will finally find its way to the Business LAN (Local Area Network) and enable a variety of applications to benefit from the newly available data. For instance, CMMS (Computer Maintenance Management Systems) or EAMS (Enterprise Asset Management Systems) such as Maximo, Indus, IFS, and Ivara will be able to obtain equipment condition data so they can implement a CBM (Conditions Based Maintenance) program. ERP (Enterprise Resource Planning) applications such as SAP, Oracle, PeopleSoft, JD Edwards, and Baan will be able to obtain inventory information, or even send production orders without any manual intervention.

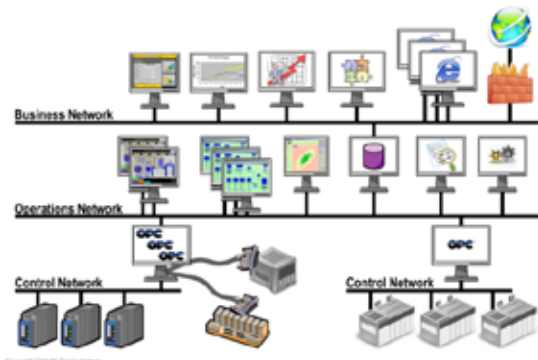


Image 4: OPC UA will enable data to move from the shop floor to the top floor.

Security: the new challenge for Automation

OPC UA makes it relatively easy for a multitude of applications to connect with each other. So the new challenge for Automation personnel will be to secure their systems from unwanted connections. Web services will make it easy to cross firewalls and networks. So, unwanted connections from people and applications will become far more common. However, unlike IT systems, Automation

systems are responsible for production and safety. Therefore, security will quickly rise to the forefront. Automation personnel will have to learn how to secure their systems in a way that still enables them to provide access to those who need it.

It remains to be seen how vendors will enable their applications with the three pillars of secure connectivity: authentication, authorization, and encryption. Various products that are already in the planning stages still do not include the necessary facilities for proper security. These applications use "security by obscurity," which essentially relies on a hacker's inability to understand how a system works to make it behave inappropriately. Both process and attitudes toward security will have to change.

Conclusion

OPC UA unifies the existing OPC specifications. It enables plants to replace the existing reliance on DCOM with Web Services. It also introduces the concept of objects, which enables people, in a range of roles at the plant, to access the same data in different ways. This enables them to produce a variety of reports and analytical calculations without having to cobble together data from many different sources. The challenge for companies implementing OPC UA is to ensure their data is secure from unauthorized access. However, given all the promise that OPC UA holds, most industries will experience a sharp increase in OPC's penetration of their plants.



Image 5: Since OPC UA provides connectivity across firewalls and networks, the first challenge for Process Control and Automation personnel will be to secure the OPC Server's data.