Understanding OPC:
Open Connectivity via Open Standards

BY TONY PAINE

The OPC Foundation

In the mid-1990s, a group of vendors convened to address the growing concern regarding connectivity to the plant floor referred to as the "Device Driver Problem."

At that time, HMI and SCADA vendors were responsible for building their own driver libraries. This approach created great solutions when it included all the connectivity requirements that their end users would need, but incomplete solutions when it did not. The vendors were faced with a decision: they either needed to invest resource application-level functionality or extend connectivity.

Some vendors decided to create their own Application Programming Interfaces (API) or Driver Toolkits. Although this solved their own connectivity needs, it limited end users to how they could approach purchasing additional solutions. Luckily, it was not too long before the market persuaded the vendors to collaborate and make changes that were in the end users' best interests.

The initial task force consisted of half a dozen companies, including Fisher-Rosemount, Intellution, and Rockwell Software, among others. They took off their competitive hats and ventured out to solve this problem. The direction was pretty clear. All software developments at the time were targeting Microsoft Windows as the platform of choice. Microsoft's client/server technology, Object Linking and Embedding (OLE), was used to share information between applications using vendor-specified interfaces and rules. The group's initial plan was to create a solution that would generalize data transfer between applications and any data source. The result was the first OPC specification, referred to as OPC Data Access (OPC DA), released in 1996.

The OPC initialism was originally called OLE for Process Control, but its meaning has changed over the years as a result of changes in the market. First, Microsoft rebranded OLE as Component Object Model (COM) not long after OPC DA was released, which essentially made OLE feel like legacy technology. Second, OPC has since found a home in many automation environments—not just process control. Therefore, the OPC Foundation saw it necessary to update OPC's meaning to include the changes in market terminology and in the technology's application. OPC is simply called Open Connectivity via Open Standards today.

OPC leveraged Microsoft's COM technology for quite some time. It was the basis for Alarm & Events (AE), Historical Data Access (HDA), and several other less-adopted specifications (like Commands, Batch, Security, and Complex Data).

Over the years, data has transformed into information or data with context. As such, the classic standards have evolved as best as possible to meet the needs of today. The latest generation of OPC is known as OPC Unified Architecture (OPC UA). Like its predecessors, OPC UA provides the same benefits of OPC such as device connectivity while offering so much more. As a member community, the OPC Foundation has since learned that they can take what they like and change what they do not like. Doing so has created a much more robust and cohesive technology for linking different domains compared to classic specifications.

The OPC Foundation continues to enjoy growth and success since its beginnings in 1994. Today, it can count more than 500 companies as members, most of which build multiple OPC-enabled applications, including servers and clients that support one or more technologies like Data Access, A&E, HDA, and so forth.

Despite its North American origins, most of the OPC Foundation's members are now in Europe (48%), followed by North America (35%), Japan (8%), and China (3%). All other regions of the world take up the remaining 6% of membership. OPC has clearly become a global standard; their latest OPC UA specification has also achieved well-deserved status by becoming an International Electrotechnical Commission (IEC) standard.

There are now thousands of OPC-enabled products registered by OPC members alone. There are also many non-members who have developed their own OPC-enabled solutions both client and server applications with the use of OPC toolkits.

Programs and Evangelism

Unfortunately, open standards are not enough to ensure the best solutions for end users. Vendors can interpret the specifications differently, which results in non-interoperable or inconsistent solutions. Although the specifications have clarified and removed many ambiguities over the years, the success of any standard relies on some sort of conformance tool. Fortunately, the OPC Foundation has solved this problem.

The OPC Foundation allows vendors to certify their products. There are two types of certification: self-certification and independent lab certification. The ability to perform self-certification came first, through a Compliance Test Tool (CTT) that allowed members to download, install, and run against their applications. There has always been a better offering of server-based compliance tools because servers must implement all the required functionality that is defined by the OPC specifications, whereas clients only need to implement the functionality that makes sense for their application. It can be difficult to determine whether the client application is operating correctly without having specific knowledge of the product. Luckily, Compliance Test Tools are very rich and allow vendors to easily understand the product areas that are non-compliant. They also assist with the debugging and resolution of these problematic areas. If an application can run through a CTT suite of testing without errors, a report will be generated that is then sent to the OPC Foundation for a self-certified logo. These tests exist for a majority of OPC interfaces, including DA, A&E, HDA, and UA.

One key disadvantage in this type of testing is that vendors may not always test in a real-world scenario. It's much easier to run a product reliably and consistently in an environment that is tightly controlled.

To address this and to provide end users with a more confident stamp of approval, the OPC Foundation had the idea of creating an independent test lab. There is currently one in Germany (run by a company known as Ascolab) and one in Scottsdale, Arizona (run by the OPC Foundation at their North American headquarters). Self-certification tests are run before the labs put a real-world test against the OPC-enabled applications. The labs also support client-side testing, which is very hard to accomplish using a self-automated tool. Lab-certified products are then used to assist with the third-party certification of other products. If successful, the vendor will be presented with a Lab Certified logo that specifies the product and version that underwent testing.

Another testing method leverages the basic concept behind the OPC Foundation: it gets vendors (who are sometimes competitors) together to test clients against servers using a pre-defined set of tests. These interoperability (IOP) workshops pre-date both self-certification and third-party certification, and give vendors an excellent opportunity to test and validate against the very products that their end users will utilize. There are three IOP workshops each year: one in North America, one in Europe, and one in Asia. In 2012, most companies focused on testing first generation UA-enabled products, because most classic OPC-enabled products have matured and are stable by now.

Self-certification, investment in third-party certification, and participation in IOP workshops are key differentiators between product vendors.

In order to educate and update the engineering community on OPC advancements, the OPC Foundation began hosting OPC Roadshows. The events were held 6 to 8 times per year in various cities in North America. They were free to end users and paid for by sponsors, who gained the opportunity to engage with potentially new customers during breaks. Presentations were given by various sponsors to evangelize and educate the attendees— not to advertise their companies.

Instead of hosting several small events, European members decided to arrange fewer large events called "OPC Days." OPC Days are very similar to the North American road shows in that sponsors attend and exhibit to help with the cost of the event; however, attendees must pay a nominal charge. OPC Days aim to educate the community on the latest OPC advancements, and encourage end users to present their OPC success stories. They also provide a venue for networking, and for demonstrations that show OPC interoperability in a live setting.

Today, the OPC Foundation continues its attempts to simplify and act in the best interests of the end users. They recognize that end users face a challenge in selecting from a vast array of standards and products. Questions often arise regarding standards in the same market: How do I choose? Which standard is better? Which standard will meet my needs? Do these standards complement each other or compete with one another?

In an effort to encourage interoperability across the board, the OPC Foundation has partnered with different standards organizations to determine how to leverage beneficial features and produce an optimal result for the market. They are currently working with PLCOpen, Field Device Integration (FDI), and Electronic Device Description Language (EDDL) on how to configure devices with OPC UA. They are also working on how to leverage information models that already dominate in certain verticals (such as BACnet for Building Automation, DNP for Power, and WITSML for Oil & Gas) with the power and abundance of OPC products.

OPC is not being positioned to take over these existing specifications, but rather to provide the glue that binds the different standards and information models together. It will be interesting to see how well these standards organizations work together, because there are clearly some cases where they would compete.

The Foundations of OPC Data Access

Although most people in the automation industry are likely more knowledgeable about OPC Data Access (OPC DA) than any other OPC standard, it is helpful to review some key elements of this "Classic" specification. It is worth noting that "Classic" does not mean "Legacy:" OPC DA is not going away any time soon. In a way, it can be compared to Microsoft's DDE. Although DDE is a very old technology by Microsoft standards, it is not old by industrial automation standards where hardened and proven technologies stay in use for quite some time. The same may be said for OPC DA (or OPC Classic).

OPC generalized Data Access down to a value, a quality, and a timestamp. The value represents the data, the quality indicates whether the data is trustworthy, and the timestamp indicates the data's freshness. In order to use the data, the OPC Foundation created a well-known interface for OPC client and server applications to adhere to that is known as an Application Programming Interface (API). They also provided the redistributable binaries that are required to enable OPC on a Windows machine.

The API provides a mechanism to discover both the OPC DA servers that are available and the information or data that they contain. It also gives client applications the ability to read, write, and/or subscribe to data. Clients can decide whether they want to read data from a device or from a cache that is updated independently of the client request. They can also select whether to poll the server for data periodically or to only subscribe to the data that has changed within a specified interval. To clarify, a change in data is a change in either the value or the quality that is associated with the value. To a client and the end user, it is essentially a data event of importance.

Because OPC is built on Microsoft COM (new OLE) technologies, it benefits from the distributed nature of COM, referred to as Distributed COM or DCOM. DCOM has its pros and cons. Many end users see DCOM as challenging to configure, and an outright pain to correct when it does not work well. For example, some users have experienced situations where the client can communicate with the server, but the server fails to update the client. DCOM has also been known to lock up for over six minutes if communications are lost between the client machine and the server machine. Six minutes is a long time to wait for data updates in a manufacturing environment.

Luckily, OPC DA evolved over the years and gained new functionalities that help end users bypass DCOM anomalies and enhance DA application. The greatest improvement is the server's ability to periodically send Keep-Alive requests, which ensures that a subscription callback has not failed.

The Benefits of OPC Data Access

The benefits of OPC DA are quite simple. This technology continues to solve the device connectivity problem that prompted the establishment of the OPC Foundation almost 20 years ago.

When the specification was released, it was expected that there would only be one OPC server per piece of hardware developed and provided by the device manufacturer. This was not the case. Instead, a new market was created for companies that specialize in developing OPC-based connectivity solutions for a wide variety of data sources, making it viable for end users to have a consistent connectivity experience.

OPC's distributed nature and underlying technologies allow data requests to be aggregated through a single server that feeds data to many client applications. Multiple clients with native drivers no longer need to make the same requests for the same data to the same devices. OPC has reduced the burden on both the devices and the communications infrastructure.

Today OPC DA is a mature specification that has not experienced changes for several years. This means that the products based on the technology have also matured, and any issues have been already been identified. It is highly probable that OPC DA implementations will succeed in solving end users' connectivity needs.

Data Access trends into their solutions did not wait for the OPC Foundation to develop a specification around those types of data. Instead, they built the support on top of OPC DA. Many Alarm & Events Manager or Historian products are simply Data Access clients. For end users, there are many OPC-enabled products from which to choose.

The Marketplace Acceptance of OPC Data Access

Vendors, system integrators, and end users have all become familiar with OPC. They understand what a typical OPC server installation looks like and how to configure an OPC client. Even though many end users have experienced some challenging DCOM situations, many have either become accustomed to or accepting of the DCOM security model.

The breadth of available products enabled OPC to become widely adopted among the people who configure automation systems—even those who do not entirely understand how OPC works behind the scenes. This is similar to the way many people interact with the different peripherals connecting to personal computers: they just know that the components are going to work.

OPC DA technology is proven. Any specification that can remain untouched for a period of time and still be leveraged today really speaks to the robustness of the technology. As such, OPC has become part of an automation engineer's toolkit. Engineers have learned the tricks of the trade and understand what to expect from implementation and performance. End users have become accustomed to OPC Classic's ease of use including available data, data types, read/write permissions, update rates, and additional properties).

OPC Data Access Environments

So where is OPC DA used? The short answer is everywhere; the long answer is everywhere with a Microsoft Windows environment. Unfortunately, COM and DCOM implementations are tied to Microsoft and do not allow platform independence.

Furthermore, clients and servers must live in the same domain and behind the same firewall. Users might get OPC Classic to work through firewalls, but the process is so cumbersome that they may as well have just eliminated the firewall by the time they are successful. DCOM requires an unbelievable amount of IP ports. And the server must be able to call back to the client on random ports. It is clear that this configuration is not firewall-friendly. Unfortunately, VPN access cannot be used to get through the firewall, either.

For users that can get beyond the Microsoft requirements and live within the same firewall, OPC DA can be used successfully in a variety of different environments—from Discrete to Batch to Continuous Processing automation. In most cases, OPC solutions have been able to meet the performance, scalability, and reliability requirements of these environments.

Additional OPC Classic Specifications

Although this paper will not examine all of the other OPC Classic specifications, both Alarm & Events and Historical Data Access should be discussed. Alarm & Events and Historical Data Access are the second most widely-adopted OPC specifications. Both have similar benefits, marketplace acceptance, and use cases as OPC DA.

Alarms & Events

As with OPC DA, there was a need to generalize the management, collection, and acknowledgement of alarm and event sources. In an automated environment, there are certain process characteristics that need to be monitored for abnormal conditions. When such a condition occurs, the appropriate personnel or systems must be notified so that corrective measures can be taken.

Alarms & Events (A&E) are richer than data items in that they provide additional metadata. They can also belong to specific categories for grouping purposes, such as Process Events or System Events. Depending on the state or condition, they may have different levels of severity (such as Informational or Catastrophic). Further Alarms & Events can be logically grouped into what are known as Areas, which can be mapped to actual work cells, personnel teams, or something entirely different. Because the information can be supplied from an underlying system or device, quality applies to A&E information just as it does with OPC DA.

Clients can discover A&E servers and the alarm and event information they contain in a manner that is similar to how Data Access clients discover information. They can also determine the conditions that the alarm supports, and the underlying source data that is driving the alarm state.

Similarly, A&E clients can read and subscribe to Alarm & Event objects, and acknowledge specific conditions.

The A&E specification gives end users the flexibility to create everything from a simple event server to a more sophisticated alarm and event management system. Although implementing A&E support on top of the native Data Access initially delayed the wide adoption of this specification, many vendors have since looked closely at adopting OPC A&E in recent development efforts.

Historical Data Access

As in OPC DA and A&E, there was a need to generalize the collection and management of historical data. Historical data is much like real-time data, except it works with a collection of values, qualities, and timestamps over a period of time (instead of just one for each).

The OPC Historical Data Access (OPC HDA) specification is flexible, and allows users to create everything from a simple trend data server (such as buffered collections of raw data) to more complex data compression and analysis servers that are capable of providing summary data, history updates, history data annotations, and backfilling.

The specification defines behaviors for many well-known aggregates—methods that summarize data values over a particular time domain at the time of data retrieval. The specification also allows vendors to extend basic OPC support and provide for custom or vendor-specific aggregates.

In addition to its read and write capabilities, HDA allows for the playback of raw or aggregated data (which could be used for simulation or recreation). It also supports the annotation and documentation of a piece of history data at some instance in time. It lets clients manipulate history data by replacing or modifying existing history data through write operations or by backfilling missing data through insert capabilities.

Clients can discover HDA servers and the historical information they contain in a manner that is similar to the other OPC specifications.

It is evident that there are many commonalities between OPC DA, A&E, and HDA. There are browse services, read/write/subscribe services, quality concepts, and many other more subtle concepts that have not been discussed. Unfortunately, each specification defined its own set of interfaces in slightly different ways. This is an important detail to keep in mind.

The XML Data Access Generation

After OPC DA, A&E, and HDA were released, vendors discovered that they needed to push information into areas that were not achievable with Microsoft DCOM technology. The first area was non-Microsoft platforms, such as enterprise applications, appliances, embedded devices, and web browsers.

Around this time in the early 2000s, the Extensible Markup Language (XML) was gaining popularity in platform independence. Like HTML, XML is intentionally firewall friendly. Although HTML had been used successfully to get and post data between web browsers and web servers through firewalls, end users required a more mature and feature-rich API. The software industry standardized on Web Services by leveraging the more structured XML over HTML. It was a good fit for the time being, and met OPC's needs at the time.

A working group set out to determine how much of what had already been developed could be preserved. They investigated how to browse for XML DA servers or Web Services, in this context. Although the software industry was working on discovery mechanisms, it was not mature enough at the time to write into the XML specification. Therefore, the OPC Foundation opted to let users manually enter in the Universal Resource Identifier (URI) of an underlying XML DA Server, much like entering a URL in a web browser. They were able to retain the benefit of discovering data (or browsing for items) once connected to the server, and were able to build this feature into the specification.

XML DA clients needed to be able to read, write, and subscribe to data. The first two are relatively straightforward in that they are initiated by the client on demand—which works well in a web service model. Because Web Services are stateless by nature, users never know when a Web Client may request something of a server. In general, the client does not rely on a server to remember something from a previous request.

Furthermore, Web Services are firewall-friendly for a reason: They use one-way communications. In this architecture, a client makes a request and the server responds by an outbound connection using the well-known HTTP ports used daily in our modern world.

Because XML is text-based and "fat" in nature, the OPC Foundation decided they would need to allow some "state" to be kept in the server and that a continuous polling model (closely emulating a subscription) would be needed in order to guarantee performance. This is known as a Polled Refresh. Depending on the cleverness of the client, the server is able to achieve performance on par with true subscription based behavior over web services.

The Evolution of OPC Unified Architecture

Shortly after the XML DA specification was released, an initiative began to create XML companion specifications for Alarm & Events and Historical Data Access. The question was raised whether the OPC Foundation would have to go through that same exercise for other specifications that had similar but also dissimilar interfaces. What about in five years' time, when there are likely better ways to exchange data between applications? What about when the XML Web Services are replaced?

The OPC Foundation decided to step back and identify the commonalties shared by the different specifications. They determined that it was necessary to be able to decouple the API from the underling wire protocols, so that the new technology could be mapped to any communications transport or medium in the future—without requiring the specifications to be rewritten. With this insight, OPC Unified Architecture was born.

The team began its research by looking to the past for answers. What features are liked? What development should have been done differently? What were the problems that OPC had tried to solve but had been previously limited? Were there others in the software industry with similar problems and solutions that could be leveraged?

After compiling a long list of questions, several key objectives became obvious. The first objective was to create a technology that could run on any platform (not just Window or Linux), but something that could run up to the highest layers of an enterprise down to an appliance or embedded device. Vendors should be allowed to implement the technology on a wide range of systems, independent of the tools available for that particular platform and regardless of whether it is running a particular operating system or the applications require a particular programming language.

About a year into the UA effort, the OPC Foundation came to the conclusion that it had to invent its own high-performing OPC-specific wire protocol in order to achieve the expected performance. In doing so, they decided that it was also necessary develop a set of UA protocol stacks in multiple programming languages that application vendors could utilize.

The second objective for the new UA architecture was to consolidate the service set to deal with all the types of information in which users are interested real-time data, alarm and events data, historical data, and so forth. To do this, the OPC Foundation looked for commonalities. In an effort to avoid rewriting the technology in a few years due to a new way of exchanging data over a wire, they considered how software vendors moved data between different levels in an enterprise. The answer was Service Oriented Architecture (SOA).

SOA allows users to create a very abstract set of services that could be mapped to multiple transports without affecting the interface between the application and the service set. Ideally, the application could take advantage of new transports without having to be rebuilt with specific knowledge of the new medium.

These requirements result in a low-level base set of services like building blocks that can be specified by OPC and extended for different types of information. For example, to be able to read "something;" to be able to write "something;" to be able to discover "something." This "something" is much more than some primitive piece of data. It is an object that has one to many properties. A property could be the name of the object, the data type of the object, the value of the object, and so forth. These objects are discoverable within the UA server's address space, which is not the classic tree-based hierarchy browse space. Instead, it is a fully-integrated address space that can be viewed as a hierarchy, indicates relationships between different nodes, and allows native complex or structured-type data to be defined and accessed generically.

OPC UA can layer on the specifics to deal with real-time data, alarms and conditions data, historical data, and so forth. It can also work with other standards organizations to map others' data models to OPC; and, finally, to allow vendors to extend the generic information model for their own needs.

The third objective for the new UA architecture was to ensure the security of the information as it is delivered between client and server applications. Moving outside of a firewalled domain requires the development of a technology that protects the data's authenticity and integrity. The UA security model allows for user authentication, communication integrity and confidentiality, and verification of functional claims. These features ensure that only authenticated users or applications can communicate, that the information being exchanged cannot be compromised by an external agent, and that a client and server can predefine what the other is capable of doing. This is accomplished both by exchanging certificates and by obtaining UA-specific profiles that indicate the level of UA conformance for each party involved in the conversation.

Generically, UA requires that clients create a secure channel to configure authenticity. It also requires a session to ensure message integration, which is usually only done once due to its expense. The underlying implementation is transparent to the application levels, completed by the communications stack, and depends on the protocol or transport used to exchange messages. For example, UA Binary over TCP may be secured with the use of secure sockets (SSL), and XML Web Services may be secured with the use of HTTPS.

Furthermore, UA allows any client/server interaction to be audited and traced, which is useful and often required in regulated environments.

The Benefits of OPC Unified Architecture

The first clear benefit of OPC UA is the decoupling of the API from the wire. UA is designed to fit into Field Devices, Control Layer Applications, Manufacturing Execution Systems (MES), and Enterprise Resource Planning (ERP) applications. Its generic information model supports primitive data types (such as integers, floating point values, and strings), binary structures (such as timers, counters, and PIDs), or XML documents (which can be thought of as text based structures). OPC UA delivers an interoperability standard that provides access from shop-floor to top-floor.

UA moved away from the Microsoft-centric security model to something that is more familiar with IT departments. Most of this is defined on the protocol or transport that is utilized. Although that protocol or transport may change dramatically in the future, it should have little effect on UA as it is specified today.

Lastly, UA supports an information model that can be extended and defined to interoperate with the simplest and most complex systems.

The Marketplace Acceptance of OPC Unified Architecture

OPC UA is still relatively new to an industry that adopts change slowly. The industry needs assurance that new solutions are built on stable, reliable and proven technology that can run 24 hours a day, 7 days a week, and 365 days a year.

The UA working groups are still active, making revisions to existing parts, finishing parts that are still in development, and continuing to create companion standards. For end users, these ongoing improvements make it seem as though UA is not complete. They also prompt the question of who is going to implement OPC UA first: the servers or the clients?

Adoption does appear to be increasing, however. The engineers from Kepware Technologies that attended the 2012 North American Interoperability Workshop stated there were more UA client/server-based applications being tested than OPC Classic applications. This is to be expected as OPC Classic applications have matured whereas UA-enabled products continue to be developed.

UA is gaining more traction in Europe as well. This may be the result of initiatives to lessen dependency on Microsoft Windows operating systems, or it may also be that European vendors and customers are more agile in adopting newer technology. Whatever the reason, Europe is outpacing the rest of the world in UA adoption.

OPC Unified Architecture Environments

As described earlier, OPC UA knows no boundaries. Its environment can be anywhere from the plant floor to the Internet. It can work on any platform, and be used in markets even where OPC is not known today. It will be exciting to see what the future holds for this technology.

An Alternative to OPC UA: OPC .NET

Before concluding, it is necessary to briefly discuss OPC Xi: the "Express Interface" that also is referred to as OPC .NET.

OPC UA's complexity caused some vendors to step back and reconsider what they really needed. In doing so, they found that they wanted to migrate their COM-based applications to Microsoft's latest client/server technology (referred to as the .NET framework) instead.

Like COM, Microsoft allows users to develop their own industry interface for exchanging information without worrying about the communication and security details. Alternatively, much like UA, the .NET framework supports a decoupled API and wire protocol interface. (In fact, this MS technology was actually researched very closely while creating the UA specification.)

Some vendors have united to create a consolidated set of .NET interfaces that allow for the exchange of real-time data, alarms and events data, and historical data. OPC .NET offers many of the feature sets for these three types of data as found in OPC Classic.

The Microsoft .NET communications framework has multiple protocol/transport pairs, some of which are firewall-friendly and independent to the applications built on top of this framework. Users that don't mind Microsoft dependencies can almost think of OPC .NET as a stepping stone from OPC Classic to OPC UA.

It is interesting to note that OPC .NET was developed as a layer that can sit on top of an unmodified OPC Classic application. Vendors are encouraged to take the reference implementation wrapper and brand it on a per product basis. With vendors deciding where to put their efforts, this approach greatly simplifies the decision on how to adopt OPC .NET.

Conclusion

OPC is a widely accepted industrial communication standard that enables the exchange of data between multi-vendor devices and control applications without any proprietary restrictions. An OPC server can communicate data continuously among PLCs on the shop floor, RTUs in the field, HMI stations, and software applications on desktop PCs. Even when the hardware and software are from different vendors, OPC compliance makes continuous real-time communication possible.

OPC has contributed to improved cooperation between technology providers and users alike. OPC has helped automation suppliers provide solutions that are truly open, which in turn has given users more choices in their automation applications. This is an exciting time in the industry. Interoperability and open solutions have helped automation professionals around the globe realize the advantages of incorporating OPC into their industrial applications and take advantage of best of breed software to solve the industry's toughest application challenges.