

For start magazine OPC supplement, September 1998

By John Baier, Development Leader, Communications Business Unit Rockwell Software

The Nature of OPC

It's been nearly two years since the specifications of OLE for Process Control (OPC) were announced by the OPC Foundation. Since that time, over 135 member companies have implemented OPC in their software products, and many more are evaluating OPC to see if it makes sense for their products. The main goal of OPC is to develop a standard for interoperability among multiple vendor products. End-users are demanding higher levels of interoperability and have come to expect that software products for automation will work together seamlessly, like desktop products work together in office environments. In the past vendors were required to develop their own custom suite of servers and drivers to meet the end-users expectation and create a product that would be interoperable.

The OPC Foundation charter states that its members are dedicated to developing an open and interoperable interface standard, based upon the functional requirements of OLE/COM and DCOM technology, that fosters greater interoperability between automation/control applications, field systems/devices, and business/office applications. The OPC standard can produce the interoperability that users are demanding, giving users what they require and vendors an open standard from which to build upon.

Spec'd for Performance

The OPC standard has also been developed with optimal performance in mind. Since the foundation of OPC is Microsoft's Component Object Model (COM), much of how OPC performs is based on how COM performs. Assuming that Microsoft has done their part to make sure COM performs well, some issues remain about how OPC is specified to use COM. One performance problem that can arise with COM interfaces is whether the interface is specified to take into account a distributed environment. In other words, a COM interface may perform well when both client and server are on the same machine, but may not when the client and server are on different machines. The way to avoid this is to design the COM interface so that it limits the number of round trips (function calls) between the client and server for any given task. The OPC

specification takes care of this by using the appropriate base COM interface types, and by defining a method for efficient distributed data transfer between the client and server.

The OPC Foundation membership has shown strong support for OPC client/server development, however, some have been skeptical that vendors would be able to develop client and server applications that offer an adequate level of performance and throughput compared to vendor-specific proprietary data exchange mechanisms. There is a concern that the specifications alone can not ensure development of applications with adequate performance and throughput. The easy answer to these skeptics is: You are right.

The reason I can say that these skeptics are correct is this: *it takes more than standards and specifications to ensure that any technology will perform up to par.* Standards and specs can take you only so far — the rest is a process of good product implementation and testing including evaluation, training and more.

As a founding member of OPC, Rockwell Software is committed to making sure that OPC is widely accepted, adopted and implemented by client and server vendors as the primary mechanism for exchanging data. But, we also need to educate users that implementation of OPC depends on a multitude of factors — not just the defined specs. Below are some other factors you need to think about when implementing with OPC.

Implementation - Think Before You do

Developers implementing OPC need to evaluate their application requirements to determine realistic levels of performance and throughput that are needed. There are many features built into OPC that help clients and servers deal with grouping large amounts of data efficiently, and transferring only the necessary data between the client and the server. Choosing the correct methods within the OPC specification and applying them in an appropriate fashion can be the key on whether the implementation performs as expected.

Training - Get your Class in Gear

With any new technology, implementation can only be achieved effectively after some amount of training and research. Since the basis for OPC is COM, developers need to be familiar with programming with COM and the finer points of building highly performing COM applications. Companies all across the world offer classes on COM, but the key is finding a good one. The best approach is to look for training through the Microsoft Developer Network. They offer classes through Microsoft Certified educators in many locations. DevelopMentor, who offers a Core COM Curriculum and other COM classes has training facilities on both coasts and in London. Intertech offers classes based in the midwest, and other certified educators can be found on the Microsoft Developer Network web page at www.microsoft.com/msdn/training/com.htm.

Testing - Three Times the Charm...

Even the best engineers can't always identify potential performance problems during product design. Many times, problems cannot be seen until the software coding is nearly complete and the system is stressed in many different ways. The biggest challenge in this area is coming up with test scenarios that really stress the software to identify the bottlenecks. This requires not only good test design, but also good analysis tools to locate bottlenecks in the software. Once a bottleneck is found and removed, the process needs to be repeated to find each bottleneck in the implementation until performance is hardware bound.

We encourage vendors to measure performance on various hardware platforms, and to provide those numbers for OPC Foundation members, as an ongoing test and evaluation of the specifications, as Rockwell Software has done. We performed tests with **RSLinx**, our OPC Server to connect to Rockwell Automation Networks, to measure the amount and how fast a server could serve data to client applications. The testing focused on the subscription based architecture of OPC, where the server acquires the data from the data source and sends the data to the client application when the value (or state of the value) changed from the last value sent to the client application. The results of our study indicate that the amount of data that a server can serve to client applications (whether the client is local or remote) actually exceeds the amount of data that could realistically be processed by a typical client application.

By conducting these tests, we were able to determine the amount of performance and throughput that was possible, based on the implementation. Rockwell Software performed tests with 5 client applications to evaluate OPC performance in a distributed environment. Each client application requested 10,000 items to be updated every 250 milliseconds. A single server application was able to deterministically update each client application with 40,000 items per second. Therefore the server was able to serve two hundred thousand (200,000) items per second continuously. See Table A.

A second test was performed to evaluate the number of data words a server could serve. This test was performed with 5 client applications each requesting 100 items (each item 200 words long) to be updated every 70 milliseconds. A single server application was able to deterministically update each client application with 280,000 words per second. Therefore the server was able to one million four hundred thousand (1,400,000) words per second continuously. See Table B.

Table A.

	<i>Items</i>	<i>Changes/sec</i>	<i>Items/sec</i>
<i>Client 1</i>	10,000	4	40,000
<i>Client 2</i>	10,000	4	40,000
<i>Client 3</i>	10,000	4	40,000
<i>Client 4</i>	10,000	4	40,000
<i>Client 5</i>	10,000	4	40,000
<i>Server Total</i>			200,000

Table B.

	<i>Items</i>	<i>Words/Item</i>	<i>Changes/sec</i>	<i>Items/sec</i>	<i>Words/sec</i>
<i>Client 1</i>	100	200	14	1400	280,000
<i>Client 2</i>	100	200	14	1400	280,000
<i>Client 3</i>	100	200	14	1400	280,000

<i>Client 4</i>	100	200	14	1400	280,000
<i>Client 5</i>	100	200	14	1400	280,000
<i>Server Total</i>				7000	1,400,000

By performing similar tests, vendors can test their own implementation and provide testing data to other OPC members for reference.

The OPC Foundation and member companies focus on making sure that OPC products are interoperable, and encourage developers and users to learn more about OPC. Vendors are invited to join the Foundation as a member, implement OPC and develop tests for performance and throughput on additional platforms, and users are encouraged to research programs that use OPC when talking with vendors. The objects, interfaces and functionality defined by OPC will continue to evolve and change as the technology grows, ensuring that performance and throughput grow along with levels of interoperability. As more developers start implementing OPC, we will discover new ways to use OPC and extend our own implementations to realize the full potential of OPC.

###