

AUTOMATION

OPC without Windows?

Web services make it possible!

Dipl.-Ing. Frank Iwanitz is the Project Manager for OPC and Profinet at Softing AG in Haar near Munich

OPC specifications define software interfaces for exchanging data in the field of automation technology. They used to be based exclusively on DCOM, which meant that OPC applications could only be realized on Windows-based systems. No one was satisfied with this restriction, and now they don't have to be – because using Web services as the basis of future OPC specifications will change the situation entirely.



The OPC Data Access specification defines an interface between servers and clients. Servers enable transparent access to data. Clients are not aware of whether this data is being transmitted via a communication system or whether it comes from another application. Clients and servers may be made by different manufacturers. Compliance with the specification ensures interoperability. It is therefore no longer the responsibility of the manufacturer of a visualization system to provide drivers for different communication systems. Instead, product users now choose the best solutions for accessing data. There are numerous other areas of application, but the Windows operating system has always been a prerequisite. If the data is on a PLC, for instance, then it is necessary to have an additional OPC server on a PC. The same applies to clients which require process data but which cannot access OPC servers because there is no OPC support for their platform. This is also why OPC still plays only a minor role in the field of remote maintenance and diagnosis. Why can't the OPC Data Access specification also be used in the areas mentioned above to help save costs? It can be, in fact, but there are a few issues to take into account.

For instance, there are no resource problems on a PC, but there are problems on embedded platforms – one of the target groups of the OPC XML DA specification. Furthermore, though it takes effort, security settings can be specified with DCOM. But what happens when OPC applications have to work together on different platforms, or even over the Internet? Web services are the solution.

What are Web services?

As more products become available which implement the OPC XML Data Access specification, there will be a considerable increase in the participants involved in OPC applications. Web services form the technological basis of this specification. The Internet is currently changing from a collection of information into a world of distributed applications. In this context, components which provide modular, clearly defined, encapsulated functionality on the Internet are called Web services. The same transformation is taking place in automation. Over the past years, a number of solutions have been offered which provide access to data via Web servers. But in the future, more interaction will be needed. Users don't want to have to view prefabricated websites; instead, they want to be able to visualize data according to their individual requirements. Applications on the Internet run in a heterogeneous environment. They are implemented on a wide variety of platforms (Web phones to application servers), written in a wide variety of languages (Python to C++) and use a variety of object models (DCOM, CORBA, J2EE). How can they communicate with one another? HTML was sufficient as long as only static information was being exchanged: Web servers supplied the structured information, and any client could display the information accordingly.

Flexible data structuring

This approach was developed further. When an application uses the functionality of a Web service, it calls a method, transmits request parameters and receives response parameters in return. Spatial separation leads to a time delay between the request and response. The application sends a request message and receives a response message in return. The use of HTTP as the transport protocol and XML as a flexible way of structuring messages means that this approach is independent of platforms, languages and object models. This solution also goes by the name of SOAP.

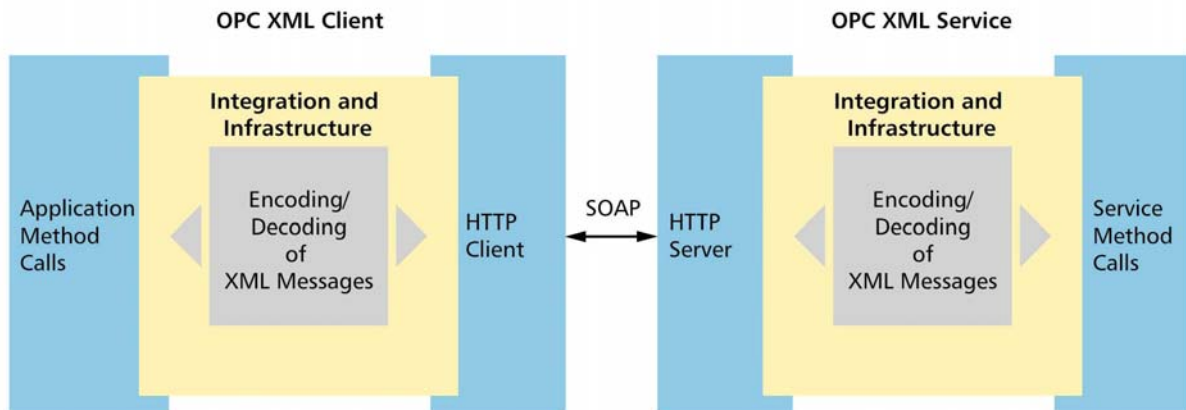
Web services provide the functionality required by an application. The application calls a method (e.g., read), and the request is then mapped to a defined XML message and sent as an HTTP request. The server receives the request, unpacks the XML message and passes the request parameters to the corresponding method depending on the required functionality.

The same components are used when the resulting values are sent back. Three elements are therefore required for Web services:

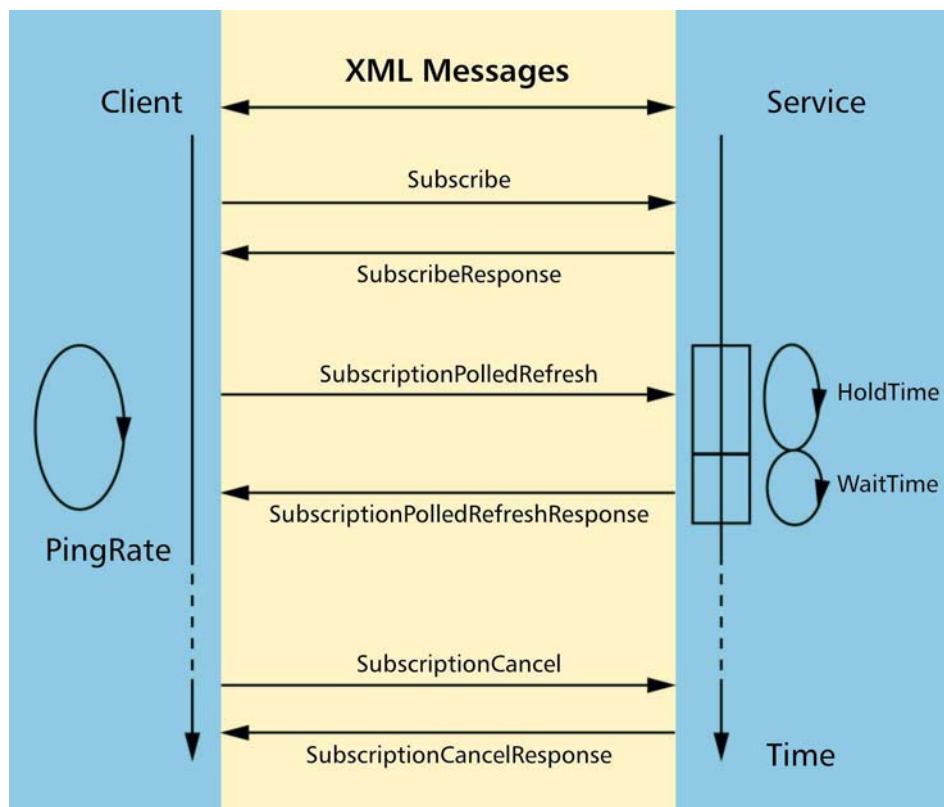
- the encoding and decoding of XML messages and mapping to HTTP requests
- the implementation of HTTP
- integration between the protocol and the application

Besides decoupling applications at runtime, Web services also make it possible to develop components independently. The advantage of this is that applications can communicate with one another regardless of how they are implemented. The only prerequisite is the availability of HTTP and XML in each environment. This applies both to an intranet and the Internet. Furthermore, Web services are "firewall friendly". The OPC Foundation defined a special Web service for this in the OPC XML DA specification and described its interface with a WSDL (Web Service Description

Language) file. The adoption of the Data Access specification model creates a namespace containing all data to which the service provides transparent access.



Figur 1: Communication process between a client and a Web service following the SOAP approach



Figur 2: Delayed response instead of polling: With a subscribe message, the client tells the service what data it wants, how quickly this data must be acquired and what the result of the transmission should be.

Widespread support for implementation

What opportunities are available for implementing OPC XML DA solutions? Tools already exist for implementing such solutions in various environments. These tools help generate the interface between the protocol and the application, and they support the creation of XML messages and their mapping to HTTP. These toolkits are divided into:

- System-specific toolkits for particular platforms and languages, such as those from Microsoft, Sun or IBM.

- Portable toolkits with the entire range of implementation support mentioned above, but which can be used on different platforms.
- Low-level support which does not take the form of a toolkit in the sense described above. HTTP and XML are supported here; a connection must be made to the application, and XML messages must be generated and mapped to HTTP requests. This approach makes it possible to realize very compact, resource-saving solutions for embedded platforms, for example.

What aspects of using OPC XML DA are significant to users? First of all, OPC XML DA offers much more freedom as regards acquiring and using interesting data. The next aspect requires us to expand our scope somewhat. Applications always want to have current data. An OPC client can continually poll a server for this data, but this is not efficient. DCOM offers the possibility of implementing callbacks. In this case, the server records values, checks them and transmits them to the client if they have changed. This is not possible with a Web service. The client can request data from the server, but it cannot send out data without being requested to do so. After all, who would want their Web client to suddenly start displaying random images? This behavior cannot be changed. But while working on the specification, the members of the working group hit upon a more efficient way to transmit data. At the client's request, the service will only return data which has changed, and it can delay this response. This is known as a "delayed response".

Efficient alternative to polling

This method is considerably more efficient than polling. Data is only sent if it has changed. In the worst case, an empty response message will be returned. Users can therefore acquire data efficiently, but they must ensure that monitoring parameters for keeping the connection open are also set in the protocols under HTTP. Another advantage of OPC XML DA is that non-Microsoft platforms can now be components of OPC applications as well. The question arises, though, as to where the respective counterparts in these exchanges are. Where are the clients and servers? Microsoft's .NET framework already includes support for Web services. Client and server applications on non-Microsoft systems can be created using existing general toolkits. But there will also be specific toolkits which only cover the OPC XML DA specification and which enable very compact, low-resource solutions. Other factors which may interest users include the ability to access information over the Internet and the security aspects associated with this. Just as we can access current technical articles using our Web browser, we can access Web services – assuming that we are authorized to do so. Work has already begun on a security architecture for Web services. In light of so many advantages, the performance limitations are negligible. Though there is greater overhead with XML, milliseconds are not likely to play a decisive role when exchanging data over the Internet.

Compact

Web services form the technological basis of the OPC XML Data Access specification. Because of this, even non-Microsoft platforms, such as embedded systems, can be used as a component in OPC applications. This makes it possible to overcome the resource problems which arise particularly with embedded systems. OPC XML DA offers significantly more freedom when it comes to acquiring data. For example, the delayed response method is a more efficient alternative to polling. A variety of toolkits and tools for developing solutions are already available.