Zeitschrift: bulletin.ch / Electrosuisse
Herausgeber: Electrosuisse
Band: 95 (2004)
Heft: 7

Artikel: Hersteller-, Plattform- und Ortsunabhängigkeit mit OPC XML
Autor: Lange, Jürgen
DOI: https://doi.org/10.5169/seals-857925

Nutzungsbedingungen
Die ETH-Bibliothek ist die Anbieterin der digitalisierten Zeitschriften auf E-Periodica. Sie besitzt keine Urheberrechte an den Zeitschriften und ist nicht verantwortlich für deren Inhalte. Die Rechte liegen in der Regel bei den Herausgebern beziehungsweise den externen Rechteinhabern. Das Veröffentlichen von Bildern in Print- und Online-Publikationen sowie auf Social Media-Kanälen oder Webseiten ist nur mit vorheriger Genehmigung der Rechteinhaber erlaubt. Mehr erfahren

Conditions d'utilisation
L'ETH Library est le fournisseur des revues numérisées. Elle ne détient aucun droit d'auteur sur les revues et n'est pas responsable de leur contenu. En règle générale, les droits sont détenus par les éditeurs ou les détenteurs de droits externes. La reproduction d'images dans des publications imprimées ou en ligne ainsi que sur des canaux de médias sociaux ou des sites web n'est autorisée qu'avec l'accord préalable des détenteurs des droits. En savoir plus

Terms of use
The ETH Library is the provider of the digitised journals. It does not own any copyrights to the journals and is not responsible for their content. The rights usually lie with the publishers or the external rights holders. Publishing images in print and online publications, as well as on social media channels or websites, is only permitted with the prior consent of the rights holders. Find out more

Download PDF: 04.02.2026

ETH-Bibliothek Zürich, E-Periodica, https://www.e-periodica.ch

Hersteller-, Plattform- und Ortsunabhängigkeit mit OPC XML

Erfolgsgeschichte OPC – neue Spezifikationen bedienen die Bedürfnisse der Automatisierung von heute

Über 3000 OPC-Produkte sind heute in mehreren hunderttausend Installationen weltweit in den verschiedensten Industriezweigen und Branchen im Einsatz. Doch der Siegeszug von OPC macht nicht Halt. Die Vision von der globalen Interoperabilität, d.h. der Kommunikation zwischen Softwareanwendungen, unabhängig davon, von welchem Hersteller sie stammen, in welcher Programmiersprache sie entwickelt wurden, auf welchem Betriebssystem sie laufen oder an welchem Ort sie sich befinden, ist dabei, Realität zu werden. Die Vorteile der OPC-Schnittstellendefinitionen in Kombination mit den neuen, Internet-fähigen Technologien XML[1], SOAP[2] und Web Services eröffnen neue Realisierungsmöglichkeiten, deren Ausmass heute noch gar nicht abzusehen ist.

Ein Kennzeichen unserer Zeit ist der rasante Wandel: laut verschiedener Studien verdoppelt sich die weltweit verfügbare Information rund alle vier Jahre, ist die Bedeutung der Informationstechnologie (IT) über alle Branchen hinweg am Wachsen und dringt das Internet immer mehr und immer tiefer in das private und berufliche Leben ein.

Dieser Trend und das Bedürfnis, dass benötigte Information heute für jede berechtigte Person an jedem Ort und zu jeder Zeit aktuell verfügbar sein soll, sind auch in der Industrieautomation deutlich zu erkennen. Der Softwareanteil in den Geräten und Anlagen ist in den vergangenen Jahren um ein Vielfaches gestiegen. Der Trend geht zu immer mehr Intelligenz in den verteilten Automatisierungskomponenten. Einst simple Sensoren oder Aktoren werden zu komplexen Geräten, deren Onboard-Funktionalität bereits einen grossen Teil der zugewiesenen Automatisierungsaufgabe abdeckt. Einfache Antriebe sind heute mit Steuerungslogik ausgestattet und können so neben der eigentlichen Funktion des Antriebs auch die Aufgabe einer SPS[3] übernehmen. Die Informationstechnologie hat längst Einzug gehalten in die Automatisierung, beispielsweise in Feldgeräte mit eigenem Internetanschluss, die von beliebigen Orten den Zugriff auf die elektronische Bedienungsanleitung ebenso ermöglichen wie die Überprüfung interner Status- und Diagnoseinformationen. Neueste Entwicklungen gehen – bezüglich der Software – in Richtung sich selbst installierender, selbst konfigurierender und selbst wartenden Automatisierungssysteme. Die Grundvoraussetzung für diese Ansätze und Konzepte ist der Einsatz offener Standards mit folgenden Eigenschaften:

- Verwendung einheitlicher Modelle für die Darstellung und Integration der einzelnen Komponenten;
- Bereitstellung offener Schnittstellen für die Interaktion von Automatisierungsbausteinen verschiedener Hersteller;
- Verwenden einer standardisierten Kommunikation zwischen verteilten Systemen;
- Bereitstellung definierter Mechanismen für das selbstständige Auffinden von Komponenten und das Erfassen der zur Verfügung gestellten Funktionalität.

Microsoft Windows und damit verbundene Basistechnologien haben massgeb-

Jürgen Lange

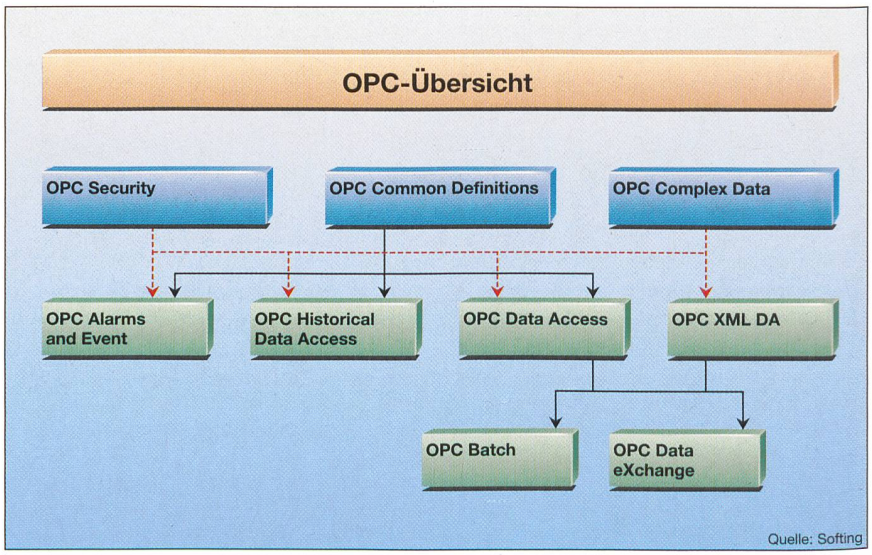


Bild 1 Verfügbare OPC-Spezifikationen

lich zur Entwicklung und Verbreitung solcher Standards beigetragen. Sowohl auf der IT-Ebene mit ERP[4)]-, Produktplanungs- oder Datenhaltungssystemen als auch auf der Automatisierungsebene oder in der Verbindung dieser beiden Welten – der so genannten vertikalen Integration – sind Microsoft-Basistechnologien wesentliche Grundsteine.

OPC ist der Standard in der Automatisierung

Auch OPC profitierte von der starken Verbreitung Windows-basierter Systeme. Der Windowsrechner wurde zu einer akzeptierten Automatisierungskomponente mit den Vorteilen des Internetanschlusses und der Verfügbarkeit einfach bedienbarer Office-Programme. In den Anfängen nur für die Erfassung und Visualisierung von Peripheriewerten sowie für die Bedienung zeitlich unkritischer Parameter genutzt, kam mehr und mehr Funktionalität zur Steuerung von Prozessen mit harten Zeitanforderungen hinzu. Basierend auf der Microsoft-Basistechnologie DCOM[5)] kann mit OPC die immer wiederkehrende Aufgabe gelöst werden, Programme für die Datenerfassung, die Visualisierung und die Steuerung auf einheitliche Weise mit der Aussenwelt zu verbinden.

OPC-Server für sämtliche Feldbusse, für PC-Einsteckkarten, für proprietäre Kommunikationsprotokolle, aber auch einfach nur OPC-Server für die Kapselung bestehender Softwareanwendungen mit sehr speziellen Programmierschnittstellen wuchsen wie Pilze aus dem Boden. SCADA[6)]-Systeme und Visualisierungsprogramme, Prozessleitsysteme, PC-basierte Steuerungen und selbst ERP-Systeme sind heute ohne OPC-Schnittstelle gar nicht mehr denkbar. Über diese Schnittstelle werden heute längst nicht mehr nur Prozessdaten oder einzelne Parameter übertragen. Ganze Warenwirtschaftsdokumente, Parametersätze, Steuerungssequenzen, Videosignale oder Antriebsprogramme werden über OPC transportiert. Innerhalb nur weniger Jahre hat OPC die Industrie erobert. Ob in der Fertigungsindustrie, Prozess- und Verfahrenstechnik oder Gebäudeautomation – OPC ist der Standard.

OPC heute – ein kurzer Überblick

OPC ist die technologische Grundlage für die komfortable und leistungsfähige Verbindung von Automatisierungskomponenten mit Steuerungshardware und Feldgeräten sowie für die Integration von Office-Produkten und Informationssystemen der Unternehmensebene wie ERP und Manufacturing Execution Systems (MES). Prozessdaten auf der Feldebene können in einem Excel-Sheet dargestellt werden. Zustands- und Produktionsdaten auf der Leitebene lassen sich über OPC mühelos in einer Datenbank archivieren oder in einem Produktionsplanungssystem unter anderem weiterverarbeiten.

Die OPC-Standards sind frei zugängliche technische Spezifikationen, die Standardschnittstellen für verschiedene Einsatzbereiche in der Automatisierungstechnik definieren wie beispielsweise für den Austausch von Prozessdaten, für die Verfolgung und Quittierung von Alarmen, für die Auswertung historischer Daten. Diese Schnittstellen ermöglichen einen hoch effizienten Datenaustausch zwischen Softwarekomponenten verschiedener Hersteller. Bild 1 zeigt die gegenwärtig verfügbaren Spezifikationen.

OPC-Produkte, wie sie heute in grosser Anzahl installiert sind, basieren auf DCOM von Microsoft, welches ein Objektmodell für das Implementieren verteilter Anwendungen entsprechend dem Client-Server-Paradigma beschreibt. Ein Client kann gleichzeitig mehrere Server nutzen und ein Server kann seine Funktionalität mehreren Clients gleichzeitig zur Verfügung stellen.

Da OPC auf DCOM basiert, existieren jedoch einige technologische und administrative Grenzen beim Einsatz der Produkte. So gibt es Probleme beim Zugriff über das Internet, da DCOM nicht über Firewalls läuft, was eine direkte Adressierung von Rechnern über die Firewall verhindert. DCOM benötigt aber gerade diese Möglichkeit, um einen internen Check durchführen zu können. Ausserdem ist kein Zugriff auf Systeme möglich, die DCOM nicht unterstützen. Viele Anwendungen im Bereich ERP, MES als Datenkonsumenten oder auch Embedded Devices als Datenlieferanten laufen nicht auf Microsoft Systemen ab und scheiden damit mangels DCOM für den Einsatz von OPC aus.

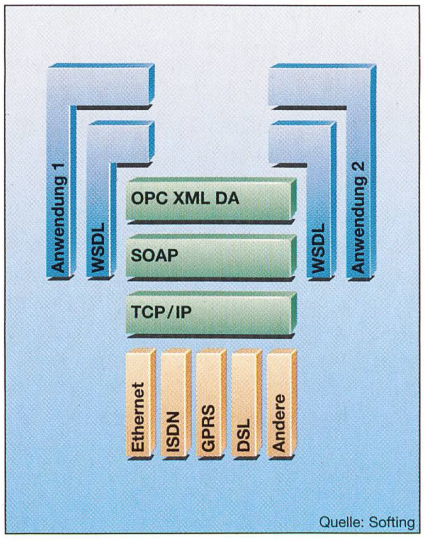


Bild 2 Infrastruktur für OPC XML-DA Produkte

Der Sprung in die Plattformunabhängigkeit und Kommunikation über das Internet mit OPC XML

Wie so oft prägte ein neuer Trend in der Informationstechnologie sehr bald die Automatisierungsindustrie: die Beschreibung von Daten in XML und der Austausch von Informationen in XML-Dokumenten über das Internet.

Frühzeitig erkannte die OPC Foundation[7)], dass XML bei der technologischen Weiterentwicklung eine bedeutende Rolle spielen würde, und startete Überlegungen, wie XML bei der OPC-basierten Kommunikation angewendet werden könnte. Arbeitsgruppen wurden ins Leben gerufen, die unter Einbeziehung von XML eine Nachrichten-orientierte Kommunikation zwischen Client und Server definierten bzw. festlegten sollten, wie komplexe Inhalte dieser Nachrichten in XML beschrieben werden.

Basierend auf Data-Access-Festlegungen begann eine Arbeitsgruppe mit der Definition der neuen «OPC XML-DA»-Spezifikation. In dieser Phase zeigte sich, dass sich sowohl Microsoft mit seinem neuen .NET-Framework als auch die Unix-Welt auf XML, Web Services und das Internet Protokoll SOAP festlegten. Aus diesem Grund entwickelte die «OPC XML-DA»-Arbeitsgruppe eine neue, auf Web Services basierende Spezifikation, die OPC von der direkten Bindung an Microsoft befreite. OPC-Clients und -Server können nun sowohl auf Microsoft- als auch auf Unix-Plattformen eingesetzt werden, aber auch auf allen anderen Plattformen, die das Implementieren von Web Services ermöglichen. OPC-Kommunikation durch den Austausch von XML-Nachrichten über SOAP kann innerhalb eines Intranets, aber auch im Internet über Firewall-Grenzen hinweg stattfinden.

Die neue Technologiebasis: XML, SOAP und Web Services

XML ist eine flexible, leicht verständliche und leicht erlernbare Datenbeschreibungssprache. Information wird mit Hilfe von lesbaren XML-Dokumen-

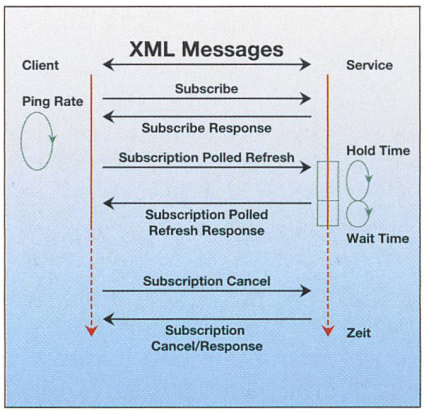


Bild 3 Ablauf der automatischen Datenübertragung basierend auf der «OPC XML-DA»-Spezifikation

Für den Aufbau einer Subscription sendet der Client einen Request, in welchem angegeben ist, an welchen Variablen der Client interessiert ist. Es können unter anderem Werte für *RequestedSamplingRate*, *Deadband* und *EnablingBuffering* vorgegeben werden. Die *RequestedSamplingRate* entspricht der *UpdateRate* von OPC DCOM-DA. Die Bedeutung des Parameters *EnableBuffering* besteht in Folgendem: der Client kann die Werte des Callbacks seltener abholen, als dies durch *RequestedSamplingRate* möglich wäre. Das heisst, der Server erfasst Werte öfter, als der Client sie abholt. Ändern sich die vom Server neu erfassten Werte und ist *EnableBuffering* = TRUE, dann wird der Server die Werte speichern und dem Client beim nächsten Aufruf mitgeben. Die *SubscriptionPingRate* ist ein vom Client vorgegebener Timeout, der durch den Service überwacht wird. Erfolgen die Aufrufe zum Abholen der Werte durch den Client nicht schneller als die *SubscriptionPingRate*, wird der Server die Subscription abbauen. Mit der *Callbackrate* informiert der Client den Server über die Frequenz, mit der erfasste Werte durch den Client angefragt werden. Der Service antwortet mit einer *SubscriptionResponse*. Sie enthält das Handle für den Callback und die unterstützte *Callbackrate* sowie verfügbare Itemwerte. Eine Subscription wird vom Client mit einem Aufruf von *SubscriptionCancelRequest* abgebaut. Der Service antwortet mit *SubscriptionCancelResponse*. Das Abholen der Daten für eine Subscription wird vom Client initiiert. Er ruft *SubscriptionPolledRefresh* auf und erhält die Daten mit *SubscriptionPolled RefreshResponse*. Im Request kann der Client dem Service mitteilen, wie lange er auf die Antwort wartet und so das Offenhalten der Verbindung für einen bestimmten Zeitraum avisieren. Durch die Vorgabe der Werte von *HoldTime* und *WaitTime* kann das Verhalten beim automatischen Übertragen der Werte vom Server zum Client nachgebildet werden. Der Service wartet mindestens *HoldTime*, bevor er die Response schickt. Ergeben sich innerhalb *HoldTime* keine Änderungen an den Werten, wartet der Service weiter. Jetzt wird *WaitTime* berücksichtigt. Hier verhält es sich aber so, dass der Service die Response sofort zum Client schickt, wenn eine Änderung festgestellt wurde. Damit ist der Zyklus abgeschlossen. Tritt auch innerhalb von *WaitTime* keine Änderung auf, wird nach Ablauf dieser Zeitspanne eine leere Response geschickt.

ten ausgetauscht. Diese enthalten mit Strukturierungsinformation angereicherten Fliesstext. Ein XML-Dokument besteht aus Elementen und Attributen, wobei Elemente weitere Elemente enthalten können. Attribute sind Elementen zugeordnet. Durch die Wahl von Elementen und Attributen erfolgt eine Strukturierung. XML legt fest, dass Elemente durch frei definierbare Tags gekennzeichnet werden.

XML-Dokumente können auf verschiedenen Wegen zwischen Anwendungen ausgetauscht werden, beispielsweise per Diskette, per E-Mail, unter Verwendung von HTTP[8)] oder unter Verwendung von TCP/IP[9)]. Die Unterstützung von XML ist heute auf praktisch allen Systemen gewährleistet. Damit können auch heterogene Systeme durch Austausch von XML-Dokumenten sehr leicht miteinander interagieren.

SOAP ist ein Interaktionsprotokoll, das zwei Technologien – XML und HTTP – verbindet. HTTP dient dabei als Transportprotokoll. Die Parameter der Interaktionen werden mit XML beschrieben. Damit ist SOAP besonders für das Internet prädestiniert. Ein SOAP-Telegramm besteht aus einem HTTP-Header, der den HTTP-Aufruf beschreibt und in allen HTTP-Telegrammen enthalten ist. Der HTTP-Header enthält Angaben zu Request/Response, Host/Rechner, Inhaltstyp (Content-Type), Länge (Content-Length) und eine sogenannte SOAP-Action, die eine Funktion beschreibt, mit der auf einen XML-DA-Service zugegriffen wird (z.B. *Read*, *Write*). Der XML-DA-Service wird mit einer URL identifiziert, die ebenfalls im HTTP-Header angegeben ist. Die Methodenparameter (z.B. für *Read*, *Write*) werden als XML übertragen. Die Abbildung des SOAP-Protokolls auf eine konkrete Implementierung ist Sache des Programmierers.

Basierend auf den vorgestellten Technologien lassen sich schon verteilte Anwendungen aufbauen, die unabhängig von Betriebssystem und Rechnerplattform miteinander über SOAP interagieren. Aber es fehlt noch etwas: ein Weg, die zur Verfügung gestellte Schnittstelle einer Anwendung zu beschreiben und aus dieser Beschreibung Programmteile zu generieren, die einerseits mit der vorhandenen Infrastruktur (HTTP usw.) zusammenwirken und andererseits in vorhandene Programme eingebaut werden können. Web Services helfen hier weiter. Sie bezeichnen die Schnittstellen, die durch Anwendungskomponenten zur Verfügung gestellt werden. Als Interaktionsprotokoll zwischen den Komponenten wird SOAP genutzt.

Web Services werden unter Verwendung von XML beschrieben. Die entsprechende Sprache wird als Web Services Description Language (WSDL) bezeichnet und wie SOAP im W3C[10)] standardisiert. Es gibt ein Schema, das beschreibt, wie eine Schnittstellenbeschreibung mit WSDL aussehen muss. Darauf aufbauend werden dann Schemata für konkrete Schnittstellen definiert. Das kann durch einen einzelnen Hersteller geschehen oder durch eine Organisation, wie beispielsweise die OPC Foundation. Möchte jetzt eine Anwendung auf einen Web Service zugreifen, so wird es eine XML-Message übermitteln, die dem Schema für die Schnittstelle entspricht. Das heisst, der Funktionsaufruf wird als XML-Message übertragen. Das trifft auch für die Antwort und eventuell zu übertragende Fehlernachrichten zu. Komponenten, die Web Services unterstützen, können auf allen Systemen implementiert werden, die XML unterstützen.

Bild 2 zeigt die Infrastruktur, auf die «OPC XML-DA»-Anwendungen zurückgreifen. Der konkrete Web Service ist innerhalb der «OPC XML-DA»-Spezifikation definiert und mit WSDL beschrieben. SOAP dient als Interaktionsprotokoll. Die verschiedenen Systeme müssen eine Unterstützung des Sendens und Empfangens von Aufrufen zur Verfügung stellen. Diese Software-Teile sind aus der WSDL-Beschreibung abgeleitet. TCP/IP, das wiederum über verschiedene Netzwerkprotokolle laufen kann, kann als Transportprotokoll unter HTTP verwendet werden.

Die «OPC XML-DA»-Spezifikation

Die Arbeiten zu OPC XML-DA begannen im Jahr 2000 mit dem Ziel, Microsofts XML-Aktivitäten in OPC zu integrieren. Diese Zielsetzung wurde dann dahingehend erweitert, dass die zu erstellende Spezifikation unabhängig von Rechnerplattform und Rechnerort eingesetzt werden kann. «OPC XML-DA»-Produkte können sowohl im Intranet als auch im Internet verwendet werden. Gleichzeitig sind Implementierungen auf verschiedenen Umgebungen möglich. Voraussetzung ist die Verfügbarkeit der Unterstützung von HTTP und XML.

Bei Betrachten dieser Anforderungen war bald klar, dass eine exakte Übernahme des Modells der «OPC DCOM-

Methode	Bedeutung
GetStatus/GetStatus Response	Abfrage des Serverstatus
Browse/ Browse-Response	Abfrage des Namensraums
GetProperties/Get-PropertiesResponse	Abfrage der Werte von Properties
Read/ ReadResponse	Lesen von Daten
Write/ WriteResponse	Schreiben von Daten
Subscribe (Client)/ SubscribeResponse (Server)	Aufbau der Subscription durch den Client
SubscriptionCancel/ Response	Abbau der Subscription durch den Client
SubscriptionPolled-Refresh(Client)/ SubscriptionPolled-RefreshResponse (Server)	Durch Client initiierte Abfrage, der für die Subscription bereitgestellten Werte

Tabelle I Die Methoden der «OPC XML-DA»-Spezifikation

DA»-Spezifikation nicht möglich war, obwohl das Ziel verfolgt wurde, eine möglichst hohe Übereinstimmung zwischen der DCOM- und der Web-Service-basierten DA-Spezifikation zu erreichen. Gründe dafür sind:

- Objektmodelle zur Realisierung der «OPC DCOM-DA»-Objekthierarchie (OPCServer-, OPCGroup-, OPCItem-Objekt) sind nicht auf allen Plattformen implementierbar.
- Das Übertragen von Daten mit XML ist weniger effizient als die Übertragung binärer Daten, wie es DCOM ermöglicht. Dies muss durch eine Wahlmöglichkeit des Clients zwischen Informationsmenge und Übertragungsperformance berücksichtigt werden. An den «OPC DCOM-DA»-Interfaces gibt es eine grosse Zahl von Methoden. Im Sinne der Effizienzsteigerung ist es sinnvoll, bei XML-DA mehrere Methoden zusammenzufassen.
- OPC DCOM-DA definiert Callbacks. Auch diese Definition ist an Voraussetzungen gebunden, die im Umfeld von Web Services so nicht existieren müssen. HTTP ist ein zustandsloses Protokoll. Der Server hat kein dauerhaftes Wissen über den Client. Es gibt zwar HTTP-Erweiterungen, die es ermöglichen, entsprechendes Wissen zu speichern, aber diese Erweiterungen in HTTP müssen nicht immer unterstützt werden. Wie oben schon erläutert, kann nicht davon ausgegangen werden, dass Server (und Clients) immer Objektmodelle implementieren. Dementsprechend wären auch die Endpunkte für Callbacks nicht vorhanden.
- In der DCOM-basierten OPC-Welt gibt es den Server Enumerator, den ein Client verwenden kann, um sich einen Überblick über die vorhandenen OPC-Server zu verschaffen. Eine ähnliche Infrastruktur ist für das Suchen nach «OPC XML-DA»-Services (noch) nicht verfügbar.

Tabelle I zeigt sämtliche Methoden der «OPC XML-DA»-Spezifikation.

Der Aufruf *GetStatus* kann zum Überprüfen der Verfügbarkeit des «OPC XML-DA»-Service[11)] verwendet werden. Erhält der Client eine entsprechende Rückantwort, so weiss er, dass der Service verfügbar ist. Andererseits kann der Client diesen Aufruf verwenden, um Information über den Service abzufragen.

Auch bei einem «OPC XML-DA»-Service enthält der Namensraum alle Variablen, auf die der Client lesend und/oder schreibend zugreifen kann. Die Vielzahl der Methoden aus OPC DCOM-DA für das Durchsuchen des Namensraums wurde in einer Methode zusammengefasst. Es gibt einen Aufruf für den Client: *Browse*, auf welchen der Service mit einer *BrowseResponse* antwortet. Der Client kann Filter, die maximale Anzahl der zurückzugebenden *Items* und bei wiederholten Aufrufen den Start- bzw. Fortsetzungspunkt für das Browsen angeben.

Mit dem Aufruf *GetProperties* kann der Client Werte für die Properties erfragen. Die Information, welche Properties vorhanden sind, erhält der Client in der *BrowseResponse*.

Nachdem der Client den Namensraum durchsucht hat, kann er Werte lesen und schreiben. Er übergibt unmittelbar den vollständigen *ItemIdentifier* für die benötigte Variable.

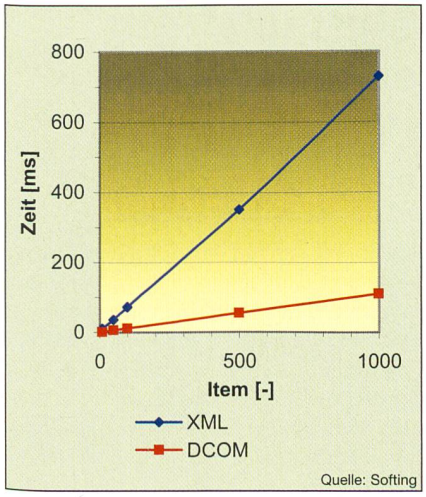


Bild 4 Ergebnisse der Messung mit OPC DCOM-DA und OPC XML-DA Server

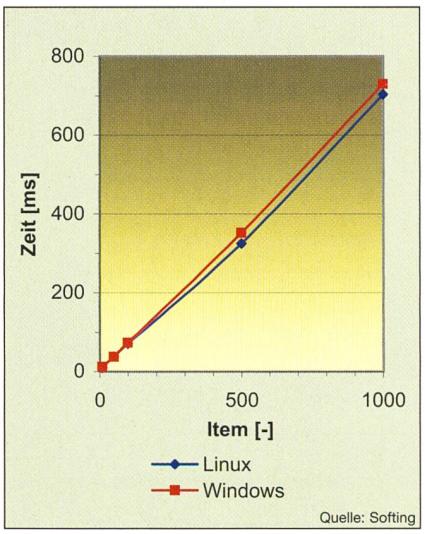


Bild 5 Ergebnisse der Messung mit OPC XML-DA Server auf Windows und Linux-Plattform

Mit Angabe von *MaxAge* gibt der Client im *ReadRequest* vor, wie alt der Wert höchstens sein darf. Ist der Wert ausreichend aktuell, wird der Service einen gespeicherten Wert zurückgeben. Ist der Wert älter als *MaxAge*, wird er einen neuen Wert anfordern und an den Client übergeben.

Das Verhalten beim Schreiben von Daten ist mit dem Verhalten beim Lesen vergleichbar. Der Client übergibt den vollständigen *ItemIdentifier* und den zu schreibenden Wert. Der Service übermittelt den Wert und gibt das Ergebnis an den Client zurück, der neben diesem Wert auch den Zeitstempel und die Qualitätsinformation schreiben sowie den Wert zurücklesen kann.

Die Spezifikation sieht auch die Möglichkeit der Subscription vor. Da Web Services keine Callbacks unterstützen, musste eine andere Möglichkeit gefunden werden (Bild 3).

Leistungsuntersuchungen von «OPC XML-DA»-Anwendungen

Eine häufig gestellte Frage bezieht sich auf die Leistungsfähigkeit der OPC-Technologie. In Bezug auf die neue «OPC XML-DA»-Spezifikation stellt sich die Frage, wie sich die Performance im Vergleich zu der auf DCOM basierten OPC-Kommunikation verhält. Ein weiterer Gesichtspunkt ist der Einfluss verschiedener Betriebssysteme auf die Geschwindigkeit des Datenaustausches. Interessant ist ebenso, wie sich das Zeitverhalten in Bezug auf die Verbindung zwischen einem «OPC XML-DA»-Client und einem «OPC XML-DA»-Server über ein Intranet oder über das Internet verhält.

Item	Data snd Client [byte]	Data rcv Client [byte]
1	739	795
10	1082	1958
50	2643	7138
100	4594	13640
1000	40596	131571

Tabelle II Zwischen Client und Service ausgetauschte Datenmenge

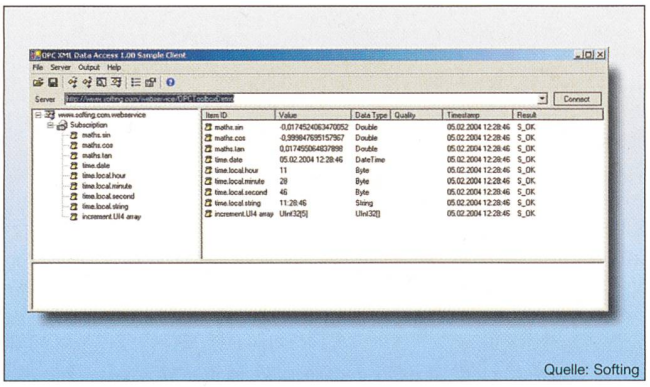


Bild 6 Zugriff eines OPC XML-DA Clients auf einen OPC XML-DA Service über das Internet

In einer Untersuchung wurden Rechner mit Intel-Prozessoren und den Betriebssystemen Windows und Linux verwendet. Ausserdem kam ein Rechner mit einem PowerPC 855T (Motorola) mit 50 MHz und Linux zum Einsatz. Client- und Serverrechner befanden sich bei den meisten Messungen im Intranet. Für eine Messung waren die Rechner über das Internet miteinander verbunden. Für die Leistungsmessungen wurden immer OPC-Items mit Datentyp VT_UI4 gelesen. Es wurde die gesamte Dauer eines synchronen *Read* für verschiedene Szenarien ermittelt.

Das Bild 4 zeigt die gemessenen Werte für eine «OPC XML-DA»- und eine «OPC DCOM-DA»-Anwendung. Server und Client liefen auf verschiedenen Rechnern mit Windows-Betriebssystem. Der Client war in .NET implementiert und der Server wurde unter Verwendung eines Toolkits in C++ programmiert. Der Datenaustausch über OPC XML-DA ist um den Faktor 6 langsamer als der Datenaustausch mit OPC DCOM-DA. Die Werte, beispielsweise 730 ms für das Lesen von 1000-UI_4-Werten, sind sicher für viele Anwendungen akzeptabel.

In einer zweiten Messreihe wurde der Einfluss des Betriebssystems auf die Performance untersucht. Bild 5 zeigt den Vergleich einer Messung mit «OPC XML-DA»-Server auf Windows und Linux Red Hat. Es wurde immer der gleiche Rechner verwendet. In diesem Fall ist praktisch kein Unterschied bei der Performance festzustellen.

Die letzte Untersuchung diente dazu, festzustellen, welche Auswirkung es auf die Performance hat, ob die «OPC XML-DA»-Anwendungen im Intranet oder im Internet laufen. Dabei erfolgte der Zugriff über das Internet von einem PC mit Windows 2000 und 866-MHz-Prozessor und DSL[12)]-Anschluss. Der Server lief auf einem PowerPC 855T (Motorola) mit 50 MHz. Der Rechner befindet sich an einer Stichleitung hinter der Firewall und ist nicht mit dem Intranet direkt verbunden. Der Zugriff auf den Rechner erfolgte von einem Client im Intranet auf den Web Service im Internet (Bild 6). Da die Firewall aber erkannte, dass Empfänger und Sender in der gleichen Domäne sind, ging der Aufruf gleich wieder zurück. Die absoluten Werte für den Intranet-Zugriff dürfen nicht mit obigen absoluten Werten verglichen werden, da eine andere Prozessorarchitektur und besonders ein deutlich langsamerer Prozessor verwendet wurde. Wichtig sind die absoluten Werte für den Zugriff über das Internet und die Differenz zwischen dem Zugriff über Internet oder Intranet (Bild 7).

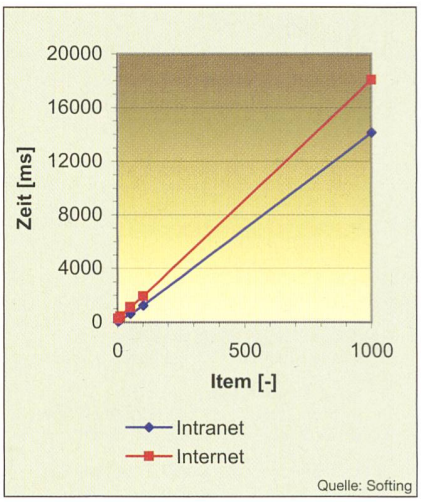


Bild 7 Zugriff auf OPC XML-DA Service über das Intranet und das Internet

Die relative Differenz nimmt mit zunehmender Anzahl von Items – also grösserer Datenmenge – ab. Etwas überraschend sind die Werte für das Internet aber schon. 10 Werte in weniger als 1 s, 50 Werte in etwas über 2 s und selbst 1000 Werte sind noch 18 s verfügbar.

Tabelle II zeigt die Datenmenge, die zwischen Client und Service und umgekehrt ausgetauscht wurde. Interessant ist, dass sich die relative Differenz zwischen Intranet- und Internet-Zugriff für grössere Datenmengen verringert.

Fazit

Seit vielen Jahren ist OPC der Standard für den Austausch von Daten in der Automation. Mit den bis heute geschaffenen Spezifikationen können vielfältigste

L'indépendance vis-à-vis des fabricants, des plates-formes et des emplacements avec OPC XML

L'histoire de succès OPC – de nouvelles spécifications répondent aux besoins de l'automatisation actuelle

Plus de 3000 produits OPC sont actuellement en service dans plusieurs centaines de milliers d'installations du monde entier et dans les secteurs industriels les plus divers. Mais le triomphe de OPC continue et la vision de l'interopérabilité mondiale, soit de la communication entre les applications logicielles, indépendamment des fabricants, du langage de programmation, du système d'exploitation et de l'emplacement, est en train de devenir réalité. Les avantages des définitions d'interface OPC en combinaison avec de nouvelles technologies aptes à Internet XML, SOAP et des services Web ouvrent de nouvelles possibilités de réalisation d'une portée encore insoupçonnée actuellement.

Anwendungsgebiete von der schnellen Datenübertragung (Data Access) über die Aufbereitung grosser historischer Datenmengen (Historical Data Access) bis zur Erfassung und Quittierung flüchtiger und kritischer Ereignisse (Alarms and Events) bedient werden.

Mit OPC XML-DA und der neu auf Web Service basierten Kommunikation hat OPC den Sprung in die Plattformunabhängigkeit geschafft und ermöglicht die OPC-Kommunikation über das Internet zukünftig auch über Firewalls. Die Wege zu einer globalen Interoperabilität sind beschritten, bei der Softwareanwendungen unabhängig von Hersteller, Programmiersprache, Betriebssystem oder Ort miteinander kommunizieren können. Mit den neuen Möglichkeiten von OPC XML-DA werden noch vielfältige weitere Anwendungsgebiete erschlossen werden.

Weiterführende Literatur

OPC Overview, Version 1.0, October 27, 1998, OPC Foundation, www.opcfoundation.org

OPC Data Access Specification, Version 2.05, Dezember 2001, OPC Foundation, www.opcfoundation.org

OPC XML-DA Specification, Version 1.00, Juli 2003, OPC Foundation, www.opcfoundation.org

Frank Iwanitz, Jürgen Lange: OPC Grundlagen, Implementierung und Anwendung. 2. Auflage, 2002, Hüthig Verlag, ISBN 3-7785-2866-1, www.softing.com/de/communications/produkte/opc/fachbuch.htm

Angaben zum Autor

Dipl. Ing. ***Jürgen Lange***, geboren 1961, studierte Elektrotechnik an der TU München. Seit 1986 arbeitet er bei der Firma Softing im Bereich Automatisierungstechnik. Er ist heute im Produkt-Marketing tätig, wo er unter anderem für OPC-Produkte verantwortlich ist. Jürgen Lange ist Mitglied des European Steering Committees der OPC Foundation.

Softing AG, D-85540 Haar/München

juergen.lange@softing.com

[1] XML: Extensible Markup Language, Metasprache für das Definieren von Dokumenttypen.
[2] SOAP: Simple Object Access Protocol. Der Standard definiert, wie Transaktionen via Internet und XML getätigt sowie dynamische Web Services über verteilte Netzwerke genutzt werden können.
[3] SPS: Speicherprogrammierbare Steuerung.
[4] ERP: Enterprise Resource Planning.
[5] DCOM: Distributed Component Object Model. Protokoll, welches es mit Software bestückten Komponenten ermöglicht, direkt oder über ein Netzwerk zu kommunizieren.
[6] SCADA: Supervisory Control and Data Acquisition.
[7] OPC Foundation: www.opcfoundation.org
[8] HTTP: Hypertext Transfer Protocol.
[9] TCP/IP: Transmission Control Protocol over Internet Protocol.
[10] W3C: Word Wide Web Consortium.
[11] OPC XML-DA basiert auf Web Services.
[12] DSL: Digital Subscriber Line.

Die Fachgruppe FOBS der ITG veranstaltet am *6. Mai in Winterthur* zu diesem Thema eine Tagung unter dem Titel

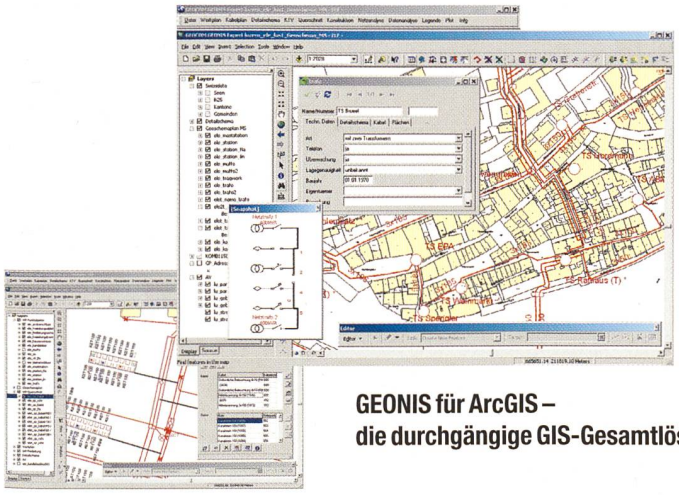
OPC – die Schnittstelle zwischen Feldbus und Informatikwelt

(siehe Seite 56 in dieser Ausgabe)

GEOCOM INFORMATIK AG

GIS von morgen schon heute

GEONIS, das moderne Netzinformationssystem für die Verwaltung von Stromversorgungsnetzen



GEONIS ermöglicht:

- die prozessorientierte Erfassung von Trassen, Kabeln, Verteilkabinen, Trafostationen, Unterwerken, für unterschiedlichste Spannungsebenen;
- die Generierung von Querschnitten mit Lage der Rohre und Kabel;
- das Erstellen von geoschematischen Kabelplänen, orthogonalen Blockschemaplänen;
- Netzverfolgungen unter Berücksichtigung von Elementattributen.

GEONIS für ArcGIS – die durchgängige GIS-Gesamtlösung vom Feldsensor bis ins Internet.

Software-Entwicklungen • Geografische Informationssysteme GIS • Hardware/Software/Netzwerke • Beratung/Schulung vor Ort/Support

Bernstrasse 21, 3400 Burgdorf, Telefon 034 428 30 30, Fax 034 428 30 32, Info@geocom.ch, www.geocom.ch