Christian Mauser
christian.mauser@gmx.at
Matr.Nr.: 0625688
Kennzahl: 066 938

183.168 Dezentrale Automation, VO

OPC Unified Architecture

Service Sets and Profiles

5. Mai 2010

Inhaltsverzeichnis

1 Einleitung

Die OPC Foundation (*http://www.opcfoundation.org/*) bringt mit der *Unified Architecture (UA)* einen neuen Standard für die Kommunikation zwischen verschiedensten Automationssystemen auf den Markt. Wie schon bei den Vorgängerversionen dieses Standards, die in der englischsprachigen Literatur (WM09) als *Classic OPC* bezeichnet werden, liegt der große Vorteil für den Kunden bei der Herstellerunabhängigkeit. Durch festgelegte Verifikationsmechanismen soll sichergestellt werden, dass Produkte unterschiedlicher Hersteller in beliebiger Zusammenstellung einwandfrei funktionieren.

Der große Nachteil von *Classic OPC* ist die Abhängigkeit von Windows-Plattformen, die durch die Verwendung von Microsofts Component Object Model (COM) bzw. Distributed COM (DCOM) für die Definition der Programmierschnittstellen (*application programming interfaces*, APIs) gegeben ist. Der erste Versuch der OPC Foundation in Richtung Plattformunabhängigkeit nennt sich *OPC XML-DA* und basiert auf dem *Simple Object Access Protocol* (SOAP) über HTTP. Aufgrund der schlechteren Performance von XML-DA im Vergleich zur COM-Lösung war dieses Projekt allerdings zum Scheitern verurteilt.

Mit UA soll das Problem der Plattformabhängigkeit nun allerdings der Vergangenheit angehören. Es soll im Vergleich zu Classic OPC keine Nachteile in Bezug auf Funktionalität und Performance geben und außerdem soll UA zusätzlich die Verwendung von komplexeren Datenstrukturen erlauben. Eine weitere Anforderung ist die Möglichkeit der Integration von alten COM-abhängigen Komponenten in ein UA-System, welche von UA ebenfalls erfüllt wird.

Die nächsten beiden Kapitel befassen sich nun mit zwei Teilen der umfangreichen Spezifikation von OPC UA (DIN EN 62541), nämlich den sogenannten *Service Sets* und den Profilen, siehe (DIN08c) resp. (DIN08d).

2 Service Sets

Die sogenannten UA *Services* bilden die Schnittstelle zwischen Server und Client und definieren den Ablauf der Datenkommunikation zwischen den beiden in abstrakter Weise in der Applikationsschicht, wodurch die Unabhängigkeit der Service-Definition vom darunterliegenden Transportprotokoll bzw. vom für die UA Applikation verwendeten Datenmodell sichergestellt ist. Abbildung 1 zeigt die Services in der UA Architektur.

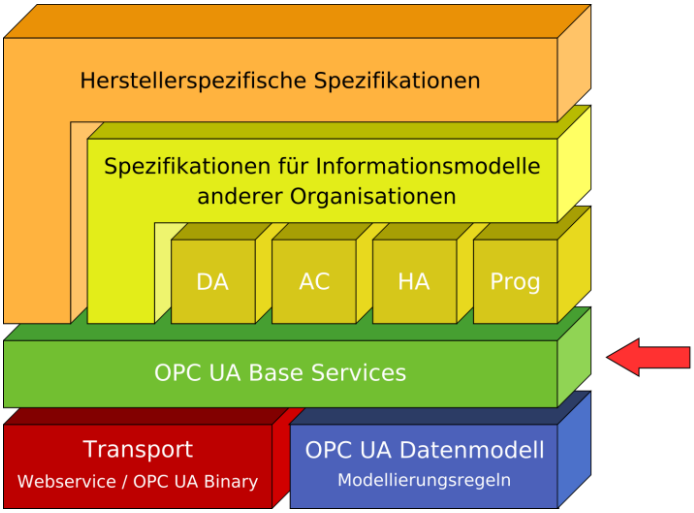

Abbildung 1: Services in der UA Architektur, aus (Wik10)

In der Spezifikation von UA sind verwandte Services zusammengefasst zu sogenannten *Service Sets*. Dabei handelt es sich um eine rein logische Einteilung in der Spezifikation, die auf die Implementierung keinen Einfluss hat. Die verschiedenen Service Sets werden nun in den folgenden Unterkapiteln etwas genauer betrachtet.

2.1 Discovery Service Set

Das Discovery Service Set beinhaltet alle Services, die mit Informationen über im System vorhandene Server arbeiten. Dazu zählen die Services:

- *FindServers*
- *GetEndpoints*
- *RegisterServer*

Das *FindServers*-Service wird von allen Servern, insbesondere vom sogenannten *Discovery Server* implementiert. Der Discovery Server antwortet auf eine derartige Anfrage mit einer Liste, die alle registrierten Server enthält, alle anderen UA Server liefern Informationen über die eigenen Services.

Für eine FindServers-Anfrage wird keine sichere Verbindung benötigt und es muss keine Session zwischen Client und Server aufgebaut werden.

GetEndpoints liefert Informationen über die Endpoints, die von den jeweiligen Servern zur Verfügung gestellt werden. Endpoints enthalten dabei alle Informationen, die für den Aufbau eines sicheren Kanals bzw. von Sessions notwendig sind. Dazu zählen u.a. die Netzwerkadresse des Servers, die Security-Einstellungen und die Art der Benutzer-Authentifizierung für den Aufbau einer Session. Auch für dieses Service wird keine sichere Verbindung und keine Session benötigt.

Das dritte und letzte Service im Discovery Service Set heißt *RegisterServer* und wie der Name schon sagt, können sich Server mit diesem Service bei einem Discovery Server anmelden. Dafür wird natürlich eine gesicherte Verbindung benötigt, damit die Vertrauenswürdigkeit des anfragenden Servers sichergestellt ist. Durch das periodische Senden einer solchen Anfrage kann durch den Discovery Server außerdem noch laufend die Verfügbarkeit des anfragenden Servers überprüft werden.

2.2 SecureChannel Service Set

Die Services im SecureChannel Service Set sind im Gegensatz zu allen anderen Services nicht auf Applikationsebene definiert, sondern im UA Kommunikations-Stack implementiert. Die Services in diesem Service Set sind:

- *OpenSecureChannel*
- *CloseSecureChannel*

Das Service *OpenSecureChannel* wird verwendet um einen sicheren Kanal zwischen Client und Server aufzubauen, das Service *CloseSecureChannel* beendet diesen Kanal wieder.

2.3 Session Service Set

Auch dieses Service Set dient dazu den Kommunikationskanal zwischen Client und Server sicher und vertrauenswürdig zu machen. Hier passiert dies allerdings auf Applikationsebene in Form von Sessions anstatt den SecureChannels auf der Ebene des UA Kommunikations-Stacks. Abbildung 2 zeigt schematisch den Unterschied zwischen einer Session und einem Secure-Channel.

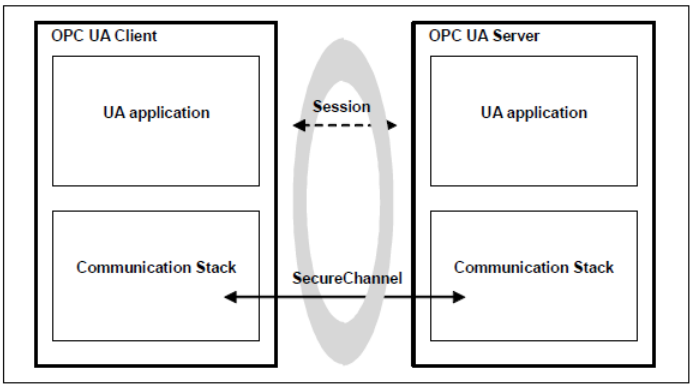

Abbildung 2: Unterschied zwischen SecureChannel und Session, aus (DIN08a)

Zu den Services in diesem Set zählen:

- *CreateSession*
- *ActivateSession*
- *CloseSession*
- *Cancel*

Die Servicenamen *CreateSession* und *CloseSession* sind im Prinzip selbsterklärend, sie werden dazu verwendet eine Session aufzubauen bzw. zu beenden. *ActivateSession* ist ebenfalls noch Teil des Handshakes zum Aufbau einer Session und wird außerdem verwendet um gewisse Parameter zu ändern, so z.B. den Benutzer der Session oder den verwendeten SecureChannel. Mit *Cancel* können noch ausständige Requests abgebrochen werden.

2.4 NodeManagement Service Set

UA Server verwenden den sogenannten Adressraum (engl. AddressSpace) um Objekte, die aus Variablen und Methoden bestehen können, für die Clients zugänglich zu machen. Diese Objekte werden im Rahmen des Adressraums als Knoten bzw. Nodes bezeichnet. Nähere Informationen zum UA Adressraum findet man im entsprechenden Teil der Spezifikation, siehe (DIN08b).

Das NodeManagement Service Set enthält Services zur Verwaltung solcher Knoten, dazu zählen:

- *AddNodes*
- *AddReferences*
- *DeleteNodes*

- *DeleteReferences*

Mittels *AddNodes* kann man solche Knoten in die Adressraum-Hierarchie einfügen. Weiters gibt es noch die Möglichkeit einem bestehenden Knoten Referenzen auf andere Knoten zuzuweisen, dies geschieht mit *AddReferences*. Analog dazu werden die Services *DeleteNodes* und *DeleteReferences* zum Entfernen von Knoten resp. Referenzen aus der Adressraum-Hierarchie verwendet.

2.5 View Service Set

Dieses Service Set vereinigt Services, die sich grob gesagt mit der Navigation durch den Adressraum beschäftigen. Dazu zählen:

- *Browse*
- *BrowseNext*
- *TranslateBrowsePathsToNodeIds*
- *RegisterNodes*
- *UnregisterNodes*

Mittels *Browse* können Clients eine Liste mit allen Referenzen, die von einem bestimmten Startknoten ausgehen, bei einem Server erfragen. Diese Liste kann zusätzlich durch Views oder einfache Filter eingeschränkt werden. Mit diesem Service wird dem Client die Navigation durch den Adressraum ermöglicht. *BrowseNext* wird verwendet, falls die Server-Antwort auf eine *Browse*-Anfrage, also die Liste der referenzierten Knoten, zu lange ist um weitere Teile der Liste zu übertragen. Die Länge dieser Liste kann sowohl clientseitig als auch serverseitig eingeschränkt werden.

Das Service *TranslateBrowsePathstoNodeIds* liefert, wie der Name schon sagt, die entsprechenden NodeIDs zu einem BrowsePath. Die NodeIDs werden benötigt um Zugriff auf die von den Knoten bereitgestellten Informationen zu erhalten. Mit dem *RegisterNodes*-Service können Clients gewisse Knoten, die sie wiederholt verwenden, beim Server registrieren, damit die Abwicklung des Zugriffes durch den Server effizienter gestaltet werden kann. Analog dazu hebt *UnregisterNodes* diese Registrierung wieder auf.

2.6 Query Service Set

Ein UA Query ermöglicht den Clients den Zugriff auf Daten, die von einem Server bereitgestellt werden, ohne dabei Wissen über den logischen Aufbau der Speicherung der Daten zu benötigen. In diesem Fall ist es für die Clients ausreichend, Informationen über den Adressraum zu haben. Die Services in diesem Service Set sind:

- *QueryFirst*
- *QueryNext*

Das *QueryFirst*-Service schickt eine *Query*-Anfrage an einen Server. Falls die Datenmenge der Antwort des Servers zu groß ist, kann man *QueryNext* verwenden um die noch ausständigen Daten abzufragen.

2.7 Attribute Service Set

Das Attribute Service Set beinhaltet Services für den Zugriff auf die Attribute eines Knotens. Die entsprechenden Services heißen:

- *Read*
- *HistoryRead*
- *Write*
- *HistoryUpdate*

Read wird verwendet um aktuelle Werte eines Attributs bzw. mehrerer Attribute von einem oder mehreren unterschiedlichen Knoten abzufragen. Im Gegensatz dazu liefert *HistoryRead* alte, aufgezeichnete Werte von Attributen. Das Service *Write* wird verwendet um aktuelle Werte von Attributen zu schreiben bzw. zu ändern und *HistoryUpdate* ändert Werte in der History eines Attributs.

2.8 Method Service Set

Dieses Service Set beinhaltet nur ein Service, nämlich:

- *Call*

Call wird von den Clients verwendet um die Methoden der Objekte aufrufen zu können.

2.9 MonitoredItem Service Set

Clients können sogenannte *MonitoredItems* definieren, d.h. sie können Attribute von Knoten mit Events verknüpfen so dass bei Auftreten eines Events eine Benachrichtigung an den Client gesendet wird. In diesem Service Set befinden sich dementsprechend alle Services, die mit der Verwaltung von MonitoredItems zu tun haben, nämlich:

- *CreateMonitoredItems*
- *ModifyMonitoredItems*
- *SetMonitoringMode*
- *SetTriggering*
- *DeleteMonitoredItems*

Mit *CreateMonitoredItems* können Clients solche MonitoredItems erzeugen und analog dazu mit *DeleteMonitoredItems* wieder löschen. Es besteht natürlich auch die Möglichkeit einer nachträglichen Änderung der Eigenschaften von MonitoredItems mittels *ModifyMonitoredItems*.

Mittels *SetMonitoringMode* kann man definieren, ob ein MonitoredItem überhaupt überwacht wird und wenn ja, ob die Clients über Events benachrichtigt werden sollen oder ob die Daten ohne Benachrichtigung in einer Queue gespeichert werden sollen. Mittels *SetTriggering* kann man ein Attribut überwachen indem man ein anderes Attribut als Trigger verwendet. D.h. nur wenn der Trigger auslöst wird die Benachrichtigung mit den Daten des eigentlich überwachten Attributs erzeugt.

2.10 Subscription Service Set

Subscriptions fassen mehrere MonitoredItems zusammen und sind dafür zuständig, dass Benachrichtigungen der MonitoredItems an den Client übertragen werden. Die Services in diesem Set sind:

- *CreateSubscription*
- *ModifySubscription*
- *SetPublishingMode*
- *Publish*
- *Republish*
- *TransferSubscriptions*
- *DeleteSubscriptions*

Die Services *CreateSubscription*, *ModifySubscription* und *DeleteSubscriptions* sind im Prinzip selbsterklärend. Das Service *SetPublishingMode* wird verwendet um das Senden der Benachrichtigungen für eine oder mehrere Subscriptions zu aktivieren. Das Service *Publish* wird einerseits dazu verwendet um empfangene Benachrichtigungen zu quittieren und andererseits um vom Server eine Benachrichtigung bzw. ein Lebenszeichen abzufragen. Für eine erneute Anfrage um eine Benachrichtigung wird *Republish* verwendet. *TransferSubscriptions* wird verwendet, um Subscriptions zwischen verschiedenen Sessions zu verschieben.

3 Profile

OPC UA vereint eine ganze Menge von unterschiedlichen Funktionalitäten, dazu zählen die klassischen Standards von OPC, nämlich OPC Data Access, OPC Historical Data Access, OPC Alarm & Events, aber auch neue Funktionalitäten. Da es in der Praxis nicht vorkommen wird, dass jedes am Markt verfügbare OPC UA Gerät alle diese Funktionalitäten unterstützt hat OPC das Konzept der sogenannten *Profile* eingeführt, die in Teil 7 der UA Spezifikation definiert sind, siehe (DIN08d). Solche Profile beschreiben genau die verwendete Untermenge der Funktionalitäten, die eine spezielle UA Applikation anbietet. Dazu gibt es auch ein Verifikationsverfahren, das die korrekte Implementierung des angegebenen Profils prüft. Bei positivem Abschluss wird für die Applikation ein signiertes Software Zertifikat ausgestellt.

Diese UA Profile gibt es auch in maschinen-lesbarer Form, damit UA Applikationen untereinander prüfen können ob die jeweilige Gegenstelle die benötigten Funktionalitäten implementiert. Falls nicht kann eine Applikation von sich aus die Kommunikation abbrechen.

Es besteht auch die Möglichkeit für eine UA Applikation mehrere Profile zu unterstützen bzw. dass Profile auch noch andere Profile enthalten. Aufgebaut sind die Profile üblicherweise aus mehreren kleineren, *testbaren* Einheiten, die *Conformance Units* genannt werden und zu denen es festgelegte Testfälle gibt. Abbildung 3 gibt einen Überblick über den Aufbau einer Beispielapplikation mit mehreren Profilen.

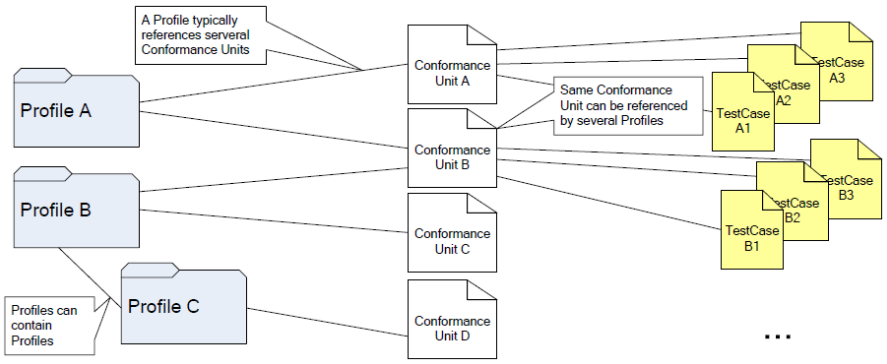

Abbildung 3: Profile und Conformance Units, aus (WM09)

3.1 Profilkategorien

UA Profile lassen sich in die folgenden Kategorien einteilen:

- **Server-Profile**
  Server-Profile beschreiben die funktionalen Eigenschaften eines Servers. Man unterscheidet zwischen voll-unterstützten Profilen und sogenannten *Facets*, wobei Facets Profile sind, die entweder Teil eines anderen, größeren Profils sind oder Profile, die gemeinsam mit anderen die Funktion des Servers beschreiben.

- **Client-Profile**
  Die Client-Profile sind das Analogon zu den Server-Profilen im Clientbereich, d.h. sie beschreiben die Funktion eines Clients. Solche Profile gibt es nur in Form von Facets.

- **Security-Profile**
  Die Security-Profile definieren verschiedene Algorithmen und Schlüssellängen für die Verschlüsselung bzw. Signatur von Nachrichten.

- **Übertragungsprofile**
  Mit den verschiedenen Übertragungsprofilen kann das gewünschte Kommunikationsprotokoll festgelegt werden.

Literatur

[DIN08a] DIN/VDE, DKE. *OPC Unified Architecture - Teil 1: Übersicht und Konzepte (IEC 65E/92/CDV:2008); Englische Fassung FprEN 62541-1:2008.* 2008

[DIN08b] DIN/VDE, DKE. *OPC Unified Architecture - Teil 3: Adressraummodell (IEC 65E/94/CDV:2008); Englische Fassung FprEN 62541-3:2008.* 12 2008

[DIN08c] DIN/VDE, DKE. *OPC Unified Architecture - Teil 4: Dienste (IEC 65E/95/CDV:2008); Englische Fassung FprEN 62541-4:2008.* 12 2008

[DIN08d] DIN/VDE, DKE. *OPC Unified Architecture - Teil 7: Profile; Englische Fassung IEC 65E/101/CD:2008.* 12 2008

[Wik10] Wikipedia. *http://de.wikipedia.org/wiki/OPC_Unified_Architecture* . 05.05.2010

[WM09] Wolfgang Mahnke, Matthias D.: *OPC Unified Architecture.* Springer-Verlag Berlin Heidelberg, 2009