Middleware

Leittechnische Schnittstellen von SPS zu SPS, zum MES- oder ERP-System werden heute ausschließlich digital über Rechner betrieben. Die Kommunikationsbasis ist ein Datennetz, in überschaubarer Zukunft immer TCP/IP auf Ethernet.

Darauf setzt man nun Middleware-Protokolle. Diese stellen eine eigene, nur für die Kommunikation gedachte und optimierte, Sprachebene mit eigenen, hierfür optimierten Übertragungsprotokollen dar.

Im SPS- und MES-Umfeld etabiliert sich dabei eindeutig OPCuA. Dies wird in naher Zukunft durch den Einsatz von "OPCuA on TSN" auch harte Echtzeitfähigkeit aufweisen.

Im Umfeld der "weißen" IT, also weg vom Prozess, in der Serverebene, haben sich XML- oder Json - basierte Webservices (REST) durchgesetzt. Hier ist Echtzeit kein Thema.

Wozu Middleware ?

In einer Industrieanlage werden Komponenten von vielen verschiedenen Herstellern eingesetzt. Diese Geräte haben unterschiedlichste Kommunikationsschnittstellen, die meist firmenspezifische Kommunikationsprotokolle benutzen.

Wenn sie nun einen Rechner betreiben wollen, der mit mehreren solchen Komponenten kommunizieren soll, müssen sie alle diese Protokolle bedienen, indem Sie z.B. Treibersoftware der jeweiligen Firmen einsetzen. Bei großen Anlagen wird das sehr aufwändig und sehr unübersichtlich.

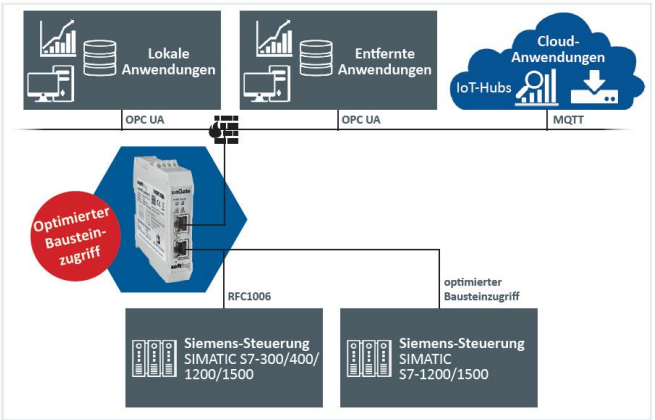
Um es anschaulicher zu machen, vergleichen wir das mit einem Fluglotsen. Wenn gerade ein Flugzeug aus Moskau, eines aus Peking, eines aus Madrid und eines aus Tokio anfliegen, deren Piloten alle ihre eigene Sprache sprechen, müßte der Lotes alle diese Sprachen beherrschen. Sie wissen, wie das gelöst wird : Man einigt sich auf eine gemeinsame “Transportsprache” : Englisch. Diese Sprache spricht im Beispiel keiner der Beteiligten im “normalen Leben”, sie wird nur für diesen Kommunikationsfall benutzt.

OPC-Server

Der Übersetzers, der die spezifische "Sprache" einer Anlagenkomponente (beim Fluglotsenbeispiel z.b. das Japanisch des Piloten aus Tokio) auf die Transportsprache "OPC-Protokoll" umsetzt, ist der OPC-Server. Jeder Teilnehmer, der Daten von Geräten in der Anlage nutzen will (lesend oder/und schreibend), bekommt auch einen Übersetzer, den OPC-Client.

Für den OPC-Server sind viele Realisierungen möglich :

Es gibt Hardwarekomponenten, die z.b. als Hutschienengerät in Schaltschränken eingebaut werden können :



(Quelle : Buxbaum/Softing)

In einer anderen Variante wird ein im Anlagennetz laufender PC (oder ein Windows-Server) als Hardwarebasis genutzt. Auf ihm installiert man einen Software-OPC-Server.
Dieser erreicht über die "eingebauten" Treiber die Geräte mit herstellerspezifischen Protokollen, und gibt als Server die Daten im Transportprotokoll OPC weiter.
Im Labornetz der Technikerschule laufen mehrere solche OPC-Server, einer davon ist der "KepwareServerEX".

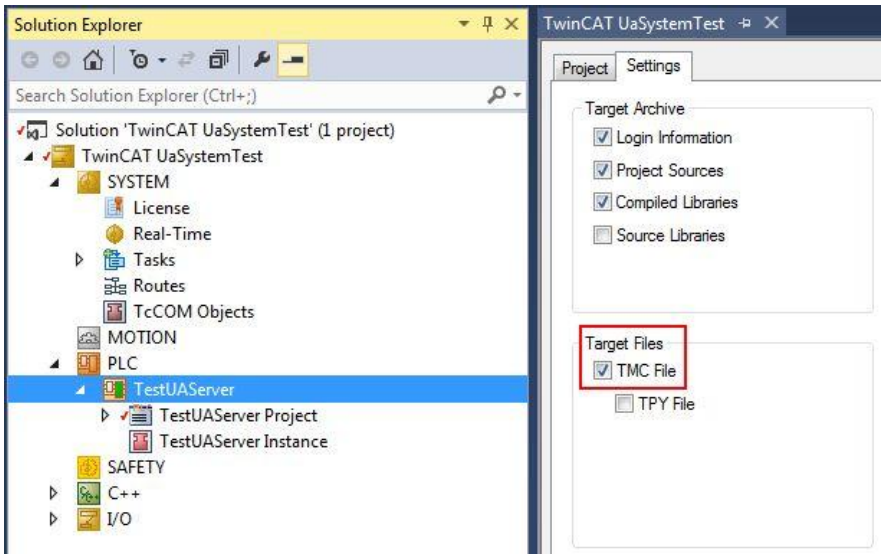
Doku dazu finden Sie hier :
https://www.kepware-opcserver.de/opc-server/#Ger%C3%A4tetreiber

Dieser Film hier erklärt die Funktion des Servers anschaulich :
https://fast.wistia.net/embed/iframe/c5016bn8nx?popover=true

Diese beiden Varianten von OPC-Server stellen aber vermutlich nur eine Übergangslösung dar. In der SPS-Welt hat sich heute schon durchgesetzt, daß jedes Gerät einen eigenen OPC-Server betreibt. Dies wird in Zukunft sicher auch für Prozessperipherie (Sensoren, Aktoren, Ventilinseln usw..) Standard werden.

Die SPS der Modellfabrik der tsm haben alle eigene integrierte OPCuA-Server, über die in der Anlage zwischen MES-Ebene und SPS kommuniziert wird.

Beispiel : Zugriff auf den OPC-Server in einem SPS-Projekt :



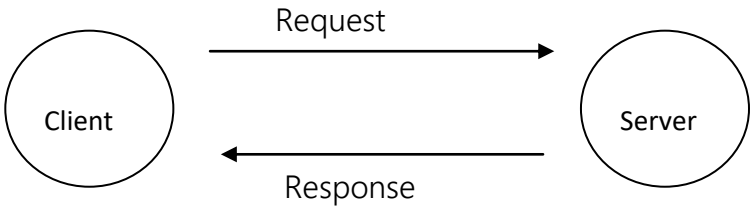
OPC-Client

Die Anlagenkomponenten (SPS, Sensoren, Aktoren…) sind nun also mit einem Übersetzer in die Kommunikationssprache OPC asugerüstet.

Jetzt kommen die Geräte, die mit den Anlagenkomponenten kommunizieren wollen. Also z.b MES-Rechner, die Abläufe koordinieren, indem Sie die SPS mit Befehlen ansteuern. Oder Rechner, die Daten lesen wollen, um Anlagenzustände ermitteln zu können. Vielleicht auch ein Techniker, der mit seinem Labtop im Wlan der Anlage auf Anlagendaten zugreift, um Fehler zu suchen.

Alle diese Anwendungen brauchen einen OPC-Client.

Um wirklich zu verstehen, welche Rolle der Server und der Client spielen, hilft es, die Grundstruktur der Client/Server-Architektur zu betrachten :



Der Server kann nützliche Sachen machen : “Dienste” ausführen. Er kann z.b. eine Webseite schicken. Oder ein Ventil schalten. Dienste heißen auf Englisch “Services”, und deshalb heißt er “Server”.

Der Client will diese Dienste jetzt nutzen. Dazu schickt er eine Dienstaufforderung, einen “Service-Request”.Der Server empfängt den Request, und antwortet, indem er den Dienst ausführt. Das ist der “Service-Response”.

Oft sind beim Response Daten dabei, z.b. eine Webseite.
Sie haben mit ihrem Client (Firefox) einen Request abgeschickt, und ein Server hat ihnen den Dienst erbracht, und eine Webseite geschickt.

Der Client wird technisch auch als Requester, der Server als Responder bezeichnet.

Also : www.google.de ist ein Responder, Firefox ist ein Requester.

Manchmal ist das gar nicht so einfach :

Eine Ventilinsel mit 8 Pneumatikventilen hängt am Ethernet.
Eine SPS, die diese Ventile steuern will, ebenfalls.
-> Wo ist der OPC-Server, wo der OPC-Client ?

Ihr MES-Rechner, den Sie im Praktikum programmieren, soll eine SPS über das OPC-Protokoll anweisen, eine Mechanikaktion auszuführen.
-> Wo ist der OPC-Server, wo der OPC-Client ?

Ein PC schickt einem Motorregler den Auftrag, einen Motor mit 4800 U/min laufen zu lassen
-> Wo ist der OPC-Server, wo der OPC-Client ?