REST (Webservices)

Schauen wir nochmal zurück :

Wir hatten Anlagen, die im Feld mit Feldbussystemen vernetzt waren, irgendwas proprietäres, vielleicht Profibus DP.

Dann kam der Wunsch, die Feldebene (“die Halle”) mit der IT der Firma zu vernetzen (vertikale Kommunikation).

Dazu passt nur Ethernet : also Ethernet “in die Halle”.

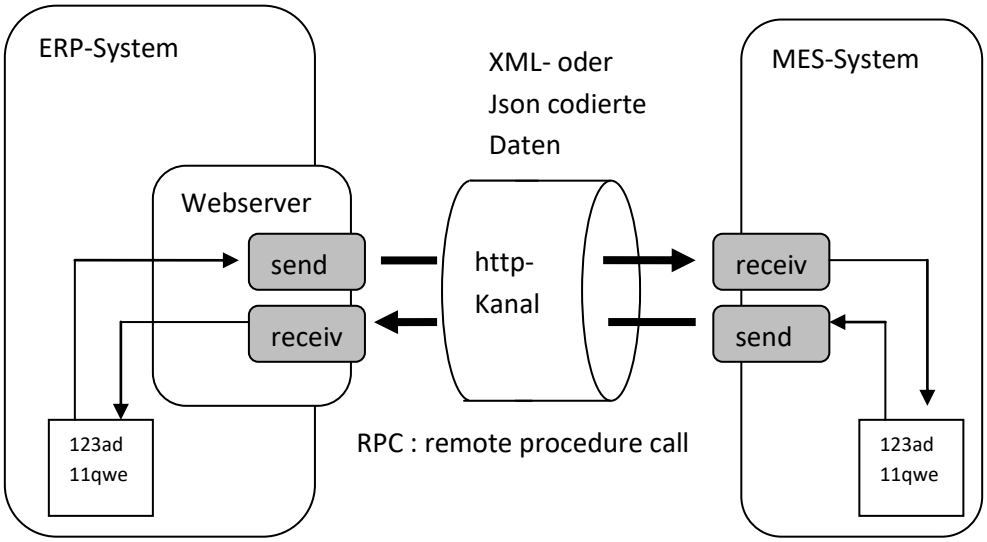
Damit die ganze schöne alte Struktur nicht weggeworfen werden muß, “simuliert” man für die SPS auf diesem Ethernet die alte Feldbusstruktur : Profinet. (Nochmal deutlich gesagt : Profinet “ist nicht Ethernet”, es ist ein Protokoll, das in Layer1/2 auf Ethernet läuft). Damit kann man aber nicht, wie gewünscht, mit der IT kommunizieren.

Man könnte jetzt aus dem reichlich vorhanden Angebot der in der IT üblichen Protokolle irgendwelche auswählen. Zum Teil geschieht das auch, z.b. der “TCP Socket” (bei und der RFID-Controller), oder ssh oder irgend sowas. (Zu vorher nochmal : auch das sind alles Protokolle, die auf Ethernet basieren)

Es ist aber anders gelaufen. Wohl um sich ein weiteres Mal eine unüberschaubare Protokollvielfalt zu ersparen, hat man sich im Kern auf zwei Protokolltypen geeinigt, mit denen die Kommunikation im Feld und zwischen Feld und MES/ERP-Ebene abgewickelt wird :

-> Erstens OPC, zweitens die “Webservices”.

„Ein Webservice ist ein Dienst, der über einen URL aufgerufen wird und XML-Daten liefert“ (Wikipedia) . In jüngster Vergangenheit werden Webservices auch vermehrt als Restful Services oder kurz Rest-Services bezeichnet.



Als Kommunikationskanal wird HTTP (wieder : auf Ethernet natürlich !) benutzt, weil es weit verbreitet ist und wegen der Nutzung von Port 80 auch einfach von Firewalls zu handeln ist.

Die Kommunikations – Endpunkte werden Stubs genannt. Sie sind beim Webserver meist in PHP programmiert, und werden vom Client aufgerufen : Remote procedure call (RPC).

Die Stubs wandeln lokale Formate (z.b. CSV, character separated values) in das Transportformat XML oder Json um. Dieser Vorgang wird Serialisierung oder Marshalling genannt, bzw. auf der Gegenseite Deserialisierung oder Demarshalling.

Hört sich alles irre kompliziert an. Also Schritt für Schritt :

Das MES-System möchte einen Wert aus einer Stückliste, die im ERP in der Datenbank steht.

Das Programm in MES ruft über den send-Stub (ein spezieller Programmteil) ein Programm (PHP wahrscheinlich) im Webserver auf. Wenn Sie bei Amazon was bestellen, machen sie das ja auch.

Dieser Aufruf (http-Request) läuft über das http-Protokoll, das geht auch übers Internet gut. Der Aufruf des Skripts im Server wird “remote procedure call” genannt.

Das Programm (Skript) das jetzt im Server anläuft hat beim Aufruf Paramter bekommen, nämlich im Beispiel die Info, welche Daten benötigt werden. Damit greift es jetzt auf die Datenbank zu und holt die gewünschten Werte.

Über den send-Stub werden die Daten nun erst codiert (in XML oder Json) und dann zum Requester (MES) zurückgeschickt (wie das geht besprechen wir in DVT sehr ausführlich).

URL

Wie schon bei OPC besprochen (Paket 6, Seite 6) erfolgt der Aufruf einer service-orientierten Kommunikation im Allgemeinen über einen Request in Form einer URL.

Das Format nochmal :

Protokoll : // Serveradresse : Port / Pfad zur Resource

Beispiele für industrieübliche Protokolle sind :

das binäre OPCuA-Protokoll :	opc.tcp
das http-Protokoll :	http
das verschlüsselte http :	https
das file-transfer-Protokoll :	ftp

Die Standard-Ports für diese Protokolle sind als "well known ports" leicht im Internet zu finden. In vielen Praxis-Anwendungen werden die Server aber auf nicht-Standard Ports konfiguriert, um Hackern mit automatisierten Portscannern die Suche zu erschweren.

Der Pfad zur gesuchten resource wird wie in einem Betriebssystem so angegeben :
/Verzeichnis/Verzeichnis/Dateiname.Erweiterung

Beispiel :
https://brunello.ts-muenchen.de:443/modellfabrik/shop.html