

# Architekturen für Industrienetzwerke

von Kai Hackbarth

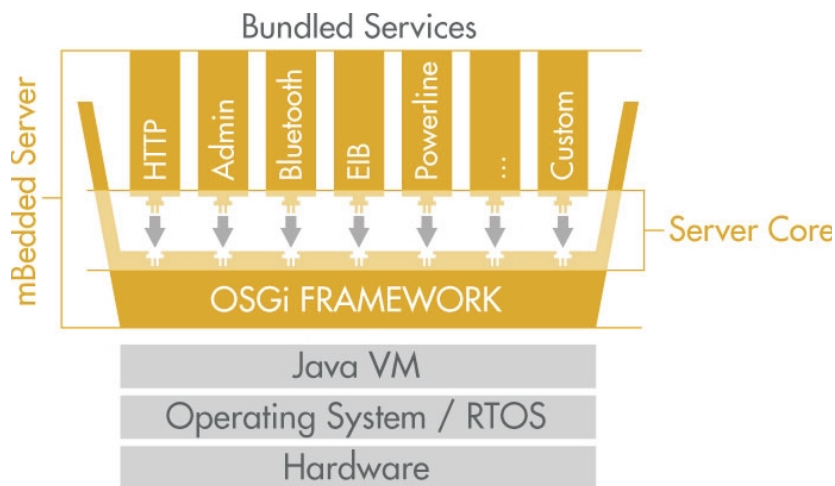


Bild 1: Open Service Gateway Software

Die untere Ebene (lower level) ist die Feldebene, auch bekannt als "Feldbus". Auf dieser Ebene wird zwischen den Endgeräten, wie z.B. Sensoren, Aktoren, und den Kontrolleinheiten, wie z.B. der SPS und der CNC, kommuniziert. Auf dieser Ebene werden sehr viele unterschiedliche Protokolle verwendet. Die höhere Ebene (upper level) wird auch als "Enterprise Ebene" bezeichnet. Auf dieser Ebene wird zwischen Feldbus und den High-End Terminals und Arbeitsplätzen kommuniziert. Üblicherweise wird hier als Protokoll TCP/IP verwendet. Die in diesen zwei Ebenen verwendeten Protokolle sind völlig unterschiedlich in Design, und somit ist eine nahtlose Verbindung sehr schwierig. Eine mögliche Lösung ist die Architektur der Open Service Gateway Initiative. Die Open Service Gateway Initiative (OSGi) ist eine unabhängige Non-profit-Organisation. Die Mission der OSGi ist die Definition und Verbreitung von offenen Spezifikationen, die die Lieferung von unterschiedlichen Services zwischen externen und lokalen Netzwerken bis hin zu den Endgeräten ermöglichen. Die OSGi Spezifikation basiert auf Java. Sie definiert ein offenes Framework, das die Übertragung und Ausführung unterschiedlicher Software-Services auf ein Service-Gateway ermöglicht. Mög-

liche Service Gateways sind z.B. eine Set-Top-Box, ein PC, ein Residential Gateway oder ein Bordcomputer in einem Auto.

## Industrienetzwerke heute

Die Enterprise-Ebene, die für Kommunikation in Firmen-Intranet zuständig ist, wird hauptsächlich für den Transfer von großen Informationsmengen, wie z.B. E-Mails, Internet-Seiten, Anfragen an Datenbanken, Produktionsreports, ESP-Daten usw. genutzt. Dagegen umfasst die Feldbus-Ebene den ganzen Informationsaustausch zwischen den Geräten, den Kontrolleinheiten und den Ingenieur-Arbeitsplätzen. Da es sich bei den Informationen um unter anderen Alarmbedingungen, Kontrollvariablen oder Setup-Punkte handelt, ist ihre Verarbeitung zeitkritisch, und sie werden in kleinen Einheiten und kurzen Zeitintervallen übertragen. Daher müssen die Transferprotokolle über eine ausreichende Sicherheit und Echtzeitfähigkeiten (Zeitresonanzen 1-20ms) verfügen. Wegen der unterschiedlichen Anforderungen an die Informationsverarbeitung in der Automatisierungstechnik werden viele Protokolle, wie z.B. Profibus, FIP, Modbus Plus, DeviceNet, Feldbus und Foundation H-1, genutzt. Die Kombination dieser vielfältigen Informationsein-

Heutige Netzwerke im Automatisierungs- und Steuerungssystemen sind im Allgemeinen in zwei Kommunikationsebenen aufgeteilt. Jede dieser Ebenen hat ihre eigenen Anforderungen und Bedürfnisse. Die Brüche zwischen den Ebenen so gering wie möglich zu halten, hat sich die unabhängige Non-profit-Organisation Open Service Gateway Initiative (OSGi) zum Ziel gesetzt. Wir zeigen was dahinter steckt.

heiten, die sowohl kleine Echtzeit-Informationen als auch große und unterschiedliche Datenmengen darstellen können, macht den Informationsaustausch auf dieser Netzwerkebene sehr kompliziert. Zusätzlich werden die Bedingungen durch proprietäre Softwarelösungen, die heute in Kontrolleinheiten und Mensch-Maschine-Interfaces (HMIs) verwendet werden, erschwert.

## Derzeitige Schwierigkeiten

### Ein Lieferant

Da die heutigen Lösungen proprietär sind, muss die Hardware auf allen Ebenen (Geräte, SPS, HMI) von ein und demselben Lieferanten gestellt werden. Dies ist eine große Einschränkung, da der Designer nicht die für seine Lösung beste Kombination von Hardwareteilen unterschiedlicher Lieferanten auswählen kann. Stattdessen muss der Designer auf einen Lieferanten zurückgreifen, welcher ihm nur eine Kompromisslösung bietet.

### Hard- und Software-Updates

Das größte Problem bei der Erneuerung von Hardware ist die Neuentwicklung neuer Kontrollalgorithmen und Benutzeroberflächen für die entsprechende Hardware. Hingegen muss bei Softwareupdates

das gesamte System gestoppt werden, was nicht nur zeit-, sondern auch kostenintensiv ist.

### Kommunikation zwischen "Feldbus" und "Enterprise"

Zum heutigen Zeitpunkt gibt es bereits einige Lösungen, die die o.g. Probleme lösen, jedoch auch Nachteile beinhalten. So konkurrieren die jungen Industriekonsortien nach wie vor darum, wessen einheitlicher Standard sich durchsetzt. Dies wiederum führt nur zu einer Verlagerung des Problems.

Dennoch kann festgehalten werden, dass einige umfassende Initiativen bereits etabliert wurden, die eine 100%ig unabhängige Plattform-Unabhängigkeit zum Ziel haben - allen voran ist hier die Initiative zu nennen, die von der OSGi (Open Service Gateway Initiative) ins Leben gerufen wurde.

### Die OSGi-Architektur

Im Jahr 1999 gründeten fünfzehn Unternehmen die OSGi, die mittlerweile mehr als 80 Unternehmen, u.a. ABB, Schneider Electric, Siemens, unterstützen. Die bereits oben erwähnte Initiative hat ein OSGi Service Framework (eine offene, modular erweiterbare Softwareplattform) spezifiziert, das die Vorteile von Java, wie z.B. Plattformunabhängigkeit oder dynamic class loading, nutzt. Hierdurch wird die Entwicklung und das dynamische Deployment von Applikationen auf Endgeräten mit geringem Speicherplatz einfacher. Durch das konsequente Programmiermodell kann ein Service Interface ohne viel Aufwand mehrere Service Implementationen beinhalten, und der Entwickler kann gegen das Service Interface codieren, ohne die bereits vorhandenen Service Implementationen zu kennen. Daraus folgt, dass das

Framework auf unterschiedlichsten Geräten ablaufen kann. Die unterschiedlichen Hardwareeigenschaften der Geräte können sich in vielen Aspekten positiv auf die Service-Implementation auswirken. Schon jetzt sichert ein Service Interface die Stabilität des gesamten Software-Systems. Ebenso stellt das Framework eine "Lifecyclemanagement"-Funktionalität zur Verfügung, die es Entwicklern erlaubt, Applikationen in kleine, selbst installierbare Komponenten aufzuteilen. Diese Komponenten werden "Bundles" genannt. Bundles können nach Belieben heruntergeladen, und, wenn sie nicht mehr benötigt werden, wieder gelöscht werden. Wenn ein Bundle auf dem Framework installiert und gestartet worden ist, kann es jegliche Services registrieren, die den anderen Bundles zur Verfügung stehen. Aufgrund der OSGi Architektur kann das Framework branchenunabhängig eingesetzt werden. Alle Kombinationen von externen und lokalen Netzwerken werden unterstützt und bestehende und zukünftige Standards können in das Framework eingebunden werden, ohne neue proprietäre Abhängigkeiten zu schaffen. Außerdem ist die Architektur für stabile Software-Anwendungen ausgelegt und erlaubt das dynamische Updaten der Software ohne dabei das ganze System anhalten zu müssen. Darüber hinaus ist das System aufgrund seiner Java-Herkunft plattform-unabhängig.

### Schlussfolgerung

Bisher sah sich die Industrieautomation mit vielen Schwierigkeiten konfrontiert, die auf proprietären Lösungen und dem Wachstum der unterschiedlichen Informationen beruhten. Durch die Einführung von Standardisierungskonzepten wurde versucht, diesen Problemen zu begegnen. Hierbei hat OSGi eines der vielversprechendsten Standardisierungskonzepte entwickelt, da es offen, modular erweiterbar und skalierbar ist. Die daraus entstehenden Spezifikationen wurden in den Software-Lösungen von ProSyst zum branchenunabhängigen Einsatz umgesetzt und stehen der Industrie zum Lösen der existierenden Barrieren zur Verfügung. Ein besonders interessanter Aspekt ist der Investitionsschutz, da Investitionsgüter im Industriebereich eine lange Lebenserwartung haben. Bei all diesen Maschinen und Anlagen wächst der Anteil der Elektronik und speziell der Software. Mit steigendem Softwareanteil erhöht sich erfahrungsgemäß auch die Fehler-, Ausfall- und Stillstandquote der Maschinen.

Hier bietet die Spezifikation der OSGi bzw. die konkreten Umsetzungen von ProSyst einen Verbesserungsansatz, denn die Software kann stets aktualisiert und Fehler aus der Ferne behoben werden. ■

5223

[www.prosyst.com](http://www.prosyst.com)

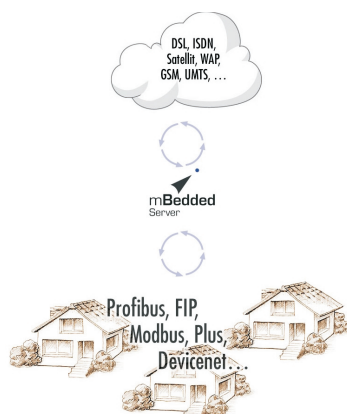
*Kai Hackbarth ist Mitarbeiter der ProSyst Software AG, Köln.*

### ProSyst und OSGi

Als Mitglied der OSGi entwickelt ProSyst Lösungen, die zu 100% OSGi-kompatibel sind. Im Vergleich zu anderen Frameworks (Softwareplattformen) hat der "mBedded Server" von ProSyst einen geringeren Speicherbedarf - ein wichtiges Kriterium für kleine Endgeräte. Ferner läuft er auf einer großen Auswahl von Java Virtual Machines (JVM), Betriebssystemen (inklusive Echtzeitbetriebssystemen wie QNX, VxWorks usw.) und sogenannten Hybriden (u.a. Jbed). Zusätzlich bietet ProSyst eine große Auswahl an schon fertigen Bundles:

- Kommunikationsprotokolle (CAN, LonWorks, EIB, EHS, X10, USB, UPnP, Jini, WAP etc.)
- Datenbank-Support
- Security (SSL, TSL, X501)
- Netzwerkmanagement (DHCP, SNMP, Jini Lookup, UPnP Control Point)

Alle Services können untereinander kommunizieren und teilen sich die Ressourcen, um die Speicheranforderungen so klein wie möglich zu halten. Um auf Enterprise Level die im Einsatz befindlichen Gateways (Frameworks) verwalten zu können, Datenfernauslese von mit dem Gateway kommunizierenden Devices zu betreiben, Fernwartung, Fernsteuerung dieser Geräte oder Anlagen durchführen zu können und diese mit Softwareupdates versorgen zu können, bietet ProSyst eine Back End Management Software an: ProSysts mPower Remote Manager ist ein Administrationstool, das hilft, den kompletten Lebenszyklus der Gateways und der Bundles zu managen. Die Aufgabe des mPower Remote Managers besteht darin, eine unbestimmte Anzahl von Gateways aus der Ferne dynamisch und automatisch zu administrieren, hierunter ist das Aufspielen und Löschen von Software, bzw. das Starten und Stoppen von Services und die Berechnung der damit verbundenen Dienstleistungen zu verstehen.



**Bild 2: Der OSGi Standard ermöglicht die Verbindung von lokalen Netzwerken mit externen Netzen**