Zeitschrift: Bulletin Electrosuisse
Herausgeber: Electrosuisse, Verband für Elektro-, Energie- und Informationstechnik
Band: 94 (2003)
Heft: 17

Artikel: Distributed Automation Systems
Autor: Felser, Max
DOI: https://doi.org/10.5169/seals-857578

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: 12.07.2025

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

Distributed Automation Systems

Herstellerübergreifende Programmierung von verteilten Steuerungen im Anlagenbau

Die aktuelle wirtschaftliche Situation und der enorme Kostendruck in der Automatisierungsbranche erfordern neue Lösungen, die weitere Kostensenkungen erlauben. Aber nur wenn mit neuen Technologien die Kosten tatsächlich gesenkt werden können, werden diese auch eingesetzt. Eine Möglichkeit zur Kostenreduktion bieten die in IEC 61499 definierten Funktionsblöcke: Sie erlauben die herstellerübergreifende Programmierung von Steuerungen in verteilten Anlagen. Mit PROFInet ist nun ein erstes Produkt auf dem Markt, das es ermöglicht, Komponenten unterschiedlicher Hersteller miteinander zu einer Anlage zu verknüpfen.

Die erste Revolution in der Automatisierungstechnik war der Einsatz der Feldbusse zur Reduktion der Verkabelungs- und Installationskosten vor 10 Jahren. Eingespart wurden vor allem Material,

Max Felser

Platz und Installationskosten. Nachkalkulationen realisierter Anwendungen zeigen, dass Einsparungen von bis zu 40% der ursprünglich veranschlagten Kosten erreicht werden können. Wer heute die Feldbustechnik nicht beherrscht, kann Automatisierungssysteme nicht mehr zu konkurrenzfähigen Preisen erstellen.

Ein zweiter Schritt der Kostenoptimierung wird mit der vertikalen Integration erreicht. Dabei wird das eingesetzte digitale Kommunikationsmittel – der Feldbus – noch für einen weiteren Nutzen herangezogen: Über die Feldbusse können nämlich nicht nur die Prozessdaten in Echtzeit übertragen, sondern auch zusätzliche Parameter und Diagnosedaten gesendet werden.

Der Nutzen besteht dabei in der zentralen Verwaltung der Konfigurationen und der Parameter sämtlicher Feldgeräte[1)] einer Anlage und der zentralen Erfassung der Diagnose und der Betriebsdaten der einzelnen Feldgeräte. Ziel einer solchen durchgängigen Information ist die Einführung bzw. Verbesserung des Asset Managements: durch die Optimierung der Wartung sollen die Stillstandszeiten einer Anlage reduziert und somit ihre mögliche Wertschöpfung erhöht werden. Dieser Schritt der Optimierung sollte eigentlich bereits mit der vorhandenen Infrastruktur mit minimalem Aufwand und einer klaren Struktur, die auf Grund ihres logischen Aufbaus den Zugriff – oder vertikalen Durchgriff – zu den lokalen Daten ermöglicht, erreicht werden können.

Der zunehmende Einsatz komplexer Technik macht eine Anlage anfällig für technische Störungen. Diese Anfälligkeit lässt sich reduzieren, indem die Rechnerleistungen und die Funktionen wie beispielsweise ein PID-Regler gezielt verteilt werden, d.h. eine Aufgabe wird dort gelöst, wo sie anfällt. Damit entstehen teilautonome Systeme, die Teilaufgaben auch bei fehlenden Netzwerkverbindungen für eine gewisse Zeit selbständig lösen können: Ein PID-Regelkreis etwa funktioniert auch ohne zentrale Kontrolle. Mit diesem Verteilen der Funktionen kann man die Fehleranfälligkeit eines Anlageteils senken und die Verfügbarkeit einer Gesamtanlage steigern.

Die Verteilung der Aufgabe ist somit wichtig für die Verfügbarkeit der Anlage, Modularisierung ist das Thema. Alle elektrischen und mechanischen Teile, aber auch die Informationsverarbeitung, die für eine Teilaufgabe benötigt wird, sollten jeweils ein Modul bilden. Die einzelnen Module können dann für die Aufgabe mehrmals verwendet und auch in anderen Aufgaben eingesetzt werden. Man spricht dabei von Automatisierungskomponenten. Solche Komponenten sind z.B. in einem Transportsystem die einzelnen Weichen, Lifte oder Transportabschnitte. Diese Automatisierungskomponenten sollen den nächsten Schub der Rationalisierung auslösen.

Die Kostenersparnis bei Automatisierungskomponenten tritt weder beim ersten Erstellen einer Anlage noch beim direkten Betrieb auf, sondern erst dann, wenn die Anlage umgebaut oder erweitert werden muss. Dann können einfach zusätzliche Module angefügt oder die bestehenden Module zur Lösung einer neuen Aufgabe neu gruppiert werden.

Experten gehen davon aus, dass Kosteneinsparungen bis zu 15% bei den Engineeringkosten realisiert werden können.

Voraussetzungen für die Realisierung von Automatisierungsobjekten

Damit Automatisierungsobjekte realisiert werden können, braucht es eine durchgehende digitale Kommunikationsinfrastruktur, kostengünstige Rechnerleistung in den verteilten Geräten und eine Methodik, um die verteilten Programme zu erstellen und in Betrieb zu nehmen.

In der Mechanik ist man es gewohnt, mit Standards für Maschinenelemente (beispielsweise für Schraubengewinde) eine normierte Modularität festzulegen. Ebenso gibt es Richtlinien für elektrische Signale wie etwa die Norm IEC 61131-2, in welcher die Eingangs- und Ausgangssignale einer Speicher-programmierbaren Steuerung (SPS) herstellerübergreifend festgelegt sind. Entsprechend braucht es auch in der Automatisierungstechnik Richtlinien, die festlegen, wie die Software unterschiedlicher Hersteller über mehrere Feldgeräte hinweg zusammen arbeiten soll. Es genügt dabei nicht, nur das Übertragungs- und das Zugriffsprotokoll festzulegen. Wesentlich für die Zusammenarbeit unterschiedlicher Feldgeräte ist das Protokoll auf der Anwen-

dungsschicht: Welche Objekte können hier definiert werden und wie arbeiten diese verschiedenen Objekte dann über ein Netzwerk zusammen? Als Objekte können dabei einfache Variablen oder Listen von Parameter auftreten, es können komplexe virtuelle Feldgeräte definiert werden oder eine Liste von verknüpften Funktionsblöcken, die bestimmte Zustände aufweisen und Funktionen erfüllen können. Das eigentliche Protokoll auf den unteren Schichten der Kommunikation muss die Anforderungen der Anwendung erfüllen und muss nichtspezielle Zugriffsverfahren unterstützen. Da die Nutzungsdauer einer Anlagekomponente bis gegen 30 Jahre gehen kann, ist es gegebenenfalls wichtig, dass die bestehenden heutigen Lösungen unterstützt werden, um so die heute schon vorhandenen Anlageteile mit einbeziehen zu können. Diese bestehenden Komponenten sind oftmals mit den klassischen Feldbussen – und heute vermehrt auch mit Ethernet – ausgerüstet. Für zukünftige Realisierungen zeichnet sich aber ab, dass als Kommunikationsplattform vor allem Ethernet eingesetzt werden wird.

Wichtig für eine dezentrale Verarbeitung ist ferner, dass die nötige Rechnerleistung auf den verschiedenen Feldgeräten zur Verfügung steht. Dies ist heute zwar kein prinzipielles Problem mehr, doch sind Lösungen, die viele Ressourcen verlangen, immer teurer als Lösungen, die sparsam mit Speicher und Prozessorleistung umgehen können.

Die eigentliche Herausforderung besteht darin, ein Steuerungsprogramm für in Echtzeit durchzuführende Aufgaben, die auf verschiedene Knoten[2] verteilt sind, in einem Netzwerk zu realisieren. Es gibt dabei unterschiedliche Lösungsansätze, die hier nachfolgend verglichen werden sollen.

Lösungsansätze für die Verteilung der Funktionen

Bei Neuanlagen ist es am einfachsten, eine uniforme Struktur einzusetzen. Dazu kann eine Technologie verwendet werden, die für die Erstellung von verteilten Kontrollaufgaben definiert wurde, etwa JAVA oder JINI. Die Automatisierungsaufgabe wird dabei in Objekte aufgeteilt, die auf die einzelnen Knoten verteilt werden. Ein verteiltes Betriebssystem – auch als Middleware bezeichnet – sorgt dafür, dass diese Objekte ihre Aufgaben untereinander koordinieren. Ein solches homogenes System kann auch weitere Problemlösungen anbieten, wie etwa das dynamische Verteilen von Aufgaben auf Knoten mit denselben Eigenschaften oder das automatische Auffinden von Objekten, die spezifische Probleme lösen. Diese Themen werden in der aktuellen Hochschulforschung bearbeitet, und es werden Lösungen für die nächste Generation von Automatisierungssystemen erarbeitet. Beispiele findet man in [1] und [2].

In der Automatisierungstechnik kann man sehr oft nicht alle Systeme neu erstellen, sondern muss bestehende Komponenten weiter verwenden. Darum wird oftmals mit «handgestrickten» Lösungen gearbeitet. Jede Komponente wird wiederverwertet oder mit den herkömmlichen Programmiermethoden und Mitteln erstellt und von Hand mit einem einfachen Protokoll verbunden. In der Welt des Ethernet hat sich dabei der Urgrossvater der Feldbusse, der Modbus als Modbus-TCP[3] für diese Lösungen weit gehend durchgesetzt. Bei diesem System werden mit einfachen Lese- und Schreibbefehlen elementare Datenelemente, Register oder Coils[4] übertragen. Der Programmierer muss nun für jeden Anlagenteil beschreiben, welche Objekte er zur Verfügung stellt und wie diese Objekte manipuliert werden müssen, um eine bestimmte Aktion auszulösen. Beim Modbus-TCP ist dies in der Regel eine Liste von Registern mit der Bedeutung ihrer Inhalte. Die Beschreibungen sind dabei oftmals in Prosa, und es besteht die Gefahr, dass die Dokumentationen oberflächlich sind und bei Anlageerweiterungen Unklarheiten auftreten.

Wie lassen sich solche Systeme warten, unterhalten und erweitern und wie ist die Dokumentation aufgebaut? Es herrscht hierbei in der Regel eine grosse Abhängigkeit vom Wissen des Systemintegrators, und die Erweiterbarkeit ist stark abhängig von seiner systematischen Vorgehensweise.

Ein weiterer Ansatz für die Automatisierungstechnik besteht darin, die in der Norm IEC 61131-3 [3] definierten Funktionsblöcke der Steuerungstechnik auch für verteilte Systeme einzusetzen. Dies wurde mit der Norm IEC 61499 [5] bereits umgesetzt. Damit ist ein Vorgehen definiert, wie bestehende Komponenten beschrieben und in neue Anlagen integriert werden können.

Gemäss diesem Ansatz wird Software als Komponente beschrieben. Die einzelnen Komponenten können dabei mit einem beliebigen System oder Werkzeug erstellt werden, werden aber auf eine einheitliche Art beschrieben, damit die Schnittstellen[5] klar definiert werden können.

In einem zweiten Schritt werden aus den Komponenten im Baukastensystem neue Systeme zusammengebaut (Bild 1). Damit diese Komponenten zusammen arbeiten können, braucht es die Spezifikation eines gemeinsamen Kommunikationsprotokolls zwischen den Anwendungen – im vorliegenden Fall also eine Middleware, die im Idealfall auch bussystemübergreifend ist, d.h. es können Komponenten mit unterschiedlichen Bussystemen kombiniert werden.

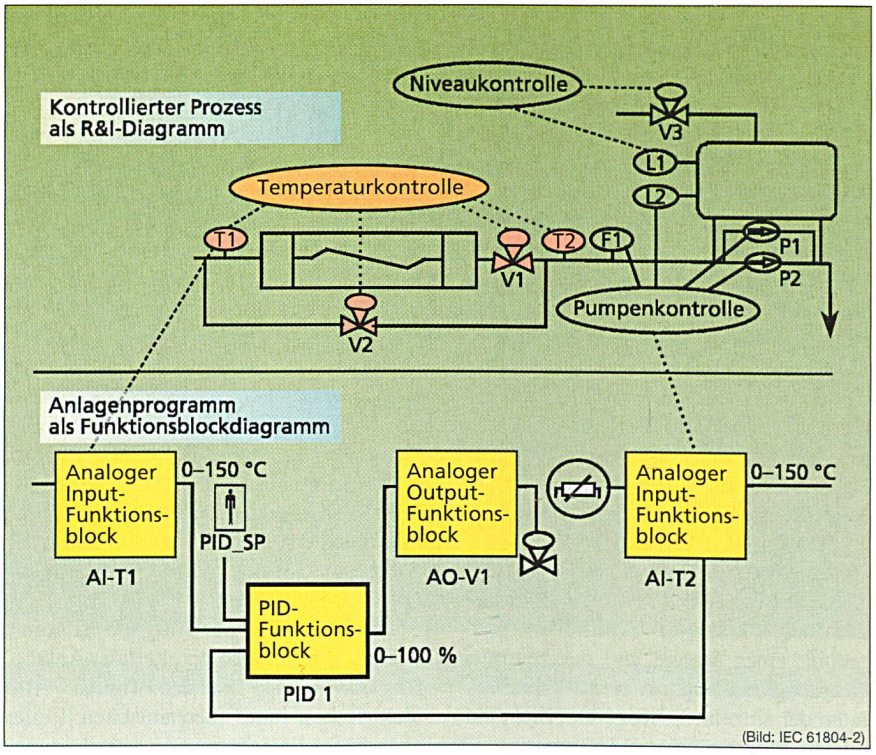


Bild 1 Einsatz von Funktionsblöcken in der Prozessautomation

Eigenschaft	Objekte	«Handgestrickt»	Funktionsblöcke
Kapselung der Daten	Ja, Objekte können auch Instanzen von anderen Objekten enthalten.	Abhängig vom Geschick des Programmierers	Ja, Funktionsblöcke können auch Instanzen von anderen Funktionsblöcken enthalten.
Externe Schnittstellen	Ja	Abhängig vom Geschick des Programmierers	Ja, es wird nicht zwischen privaten und offenen Schnittstellen unterschieden.
Aufruf	Objekte haben Methoden mit Argumenten und Rückgabewerte.	Daten zum Lesen und Schreiben	Funktionsblöcke haben Ein- und Ausgabe-Daten und -Ereignisse.
Vererbung	Ja	Nein (abhängig vom Geschick des Programmierers)	In IEC 61499 ist keine Vererbung vorgesehen.
Polymorphismus	Ja	Nein	Ein spezielles Adapterkonzept in IEC 61499 erlaubt Funktionsblöcken, gemeinsame Schnittstellen zu teilen.
Instanzierung	Objektklassen und Funktionsblock-Typen sind vergleichbare Konstrukte.	Nein	Funktionsblock-Instanzen werden von Funktionsblock Typen abgeleitet.

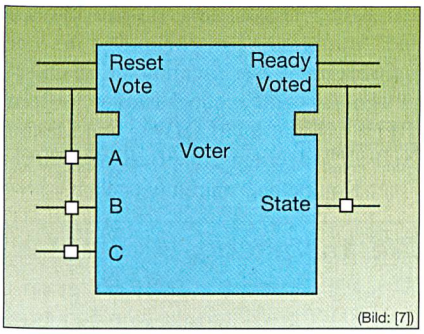
(Tabelle: erweitert aus [7])

Tabelle Vergleich der verschiedenen Methoden mit Komponenten

Die Tabelle zeigt den Vergleich der verschiedenen Verfahren nach unterschiedlichen Kriterien.

Beispiel eines Funktionsblockes nach IEC 61499

Ein Funktionsblock nach IEC 61499 kann grafisch oder als Text beschrieben werden. Dies soll nachfolgend am einfachen Beispiel eines *Voters* (Bild 2) dargestellt werden.



(Bild: [7])

Bild 2 Der Funktionsblock *Voter*

Der *Voter* hat zwei Ereigniseingänge: *Reset* und *Vote*. Der *Reset*-Eingang setzt die internen Variablen zurück und signalisiert mit dem Ereignis *Ready*, dass der Block für die Verarbeitung bereit ist.

Zuerst müssen die Eingänge A, B und C mit dem Wert *True* oder *False* belegt werden. Sobald das Ereignis *Vote* auftritt, rechnet der Funktionsblock und gibt am Ausgang *State* das Resultat *True*, wenn mindestens zwei der drei Eingänge A, B und C mit *True* belegt sind. Gleichzeitig wird mit dem Ereignis *Voted* die Gültigkeit des Wertes *State* signalisiert. Mit dem Ereignis *Reset* wird der Funktionsblock zurückgesetzt und mit *Ready* signalisiert, dass der Funktionsblock zurückgesetzt ist und eine neue Rechnung ausgelöst werden kann.

In Bild 3 ist die textuelle Beschreibung dieses Funktionsblockes wiedergegeben.

```
FUNCTION_BLOCK VOTER
EVENT_INPUT
        Reset;
        Vote WITH A, B, C;
END_EVENT
EVENT_OUTPUT
        Ready;
        Voted WITH State;
END_EVENT
VAR_INPUT
        A : BOOL;
        B : BOOL;
        C : BOOL;
END_VAR
VAR_OUTPUT
        State : BOOL;
END_VAR
EC_STATES
        Ready : ResetAlg -> Ready;
        Voted : VoteAlg -> Voted;
END_STATES
EC_TRANSITIONS
        Ready TO Voted := Vote;
        Voted TO Voted := Vote;
        Voted TO Ready := Reset;
END_TRANSITIONS

ALGORITHM ResetAlg IN ST;
        State := 0;
END_ALGORITHM

ALGORITHM VoteAlg IN ST;
        IF State = 0 THEN
                State := (A AND B) OR (B AND C ) OR (A AND C);
        END_IF;
END_ALGORITHM

END_FUNCTION_BLOCK
```

(Bild: [7])

Bild 3 Textuelle Beschreibung des in Bild 2 dargestellten Funktionsblockes *Voter*

Funktionsblöcke können auch aus anderen Funktionsblöcken zusammengesetzt sein. Dann ist die Beschreibung der Schnittstellen identisch, aber die Beschreibung der Zustände und Algorithmen wird durch eine Liste der Funktionsblöcke und der Verbindungen zusammengesetzt. Das hier vorgestellte Beispiel sowie weitere Beispiele findet man in [4].

Der Funktionsblock bzw. der zusammengesetzte Funktionsblock ist die kleinste Einheit, die einer Ressource[2)] zugeordnet werden kann. Aus Funktionsblöcken lassen sich dann Sub-Anwendungen und Anwendungen zusammensetzen, die mehreren Ressourcen in einem Gerät oder mehreren auf mehreren Geräten verteilten Ressourcen zugeordnet werden können.

Umsetzungen mit PROFInet

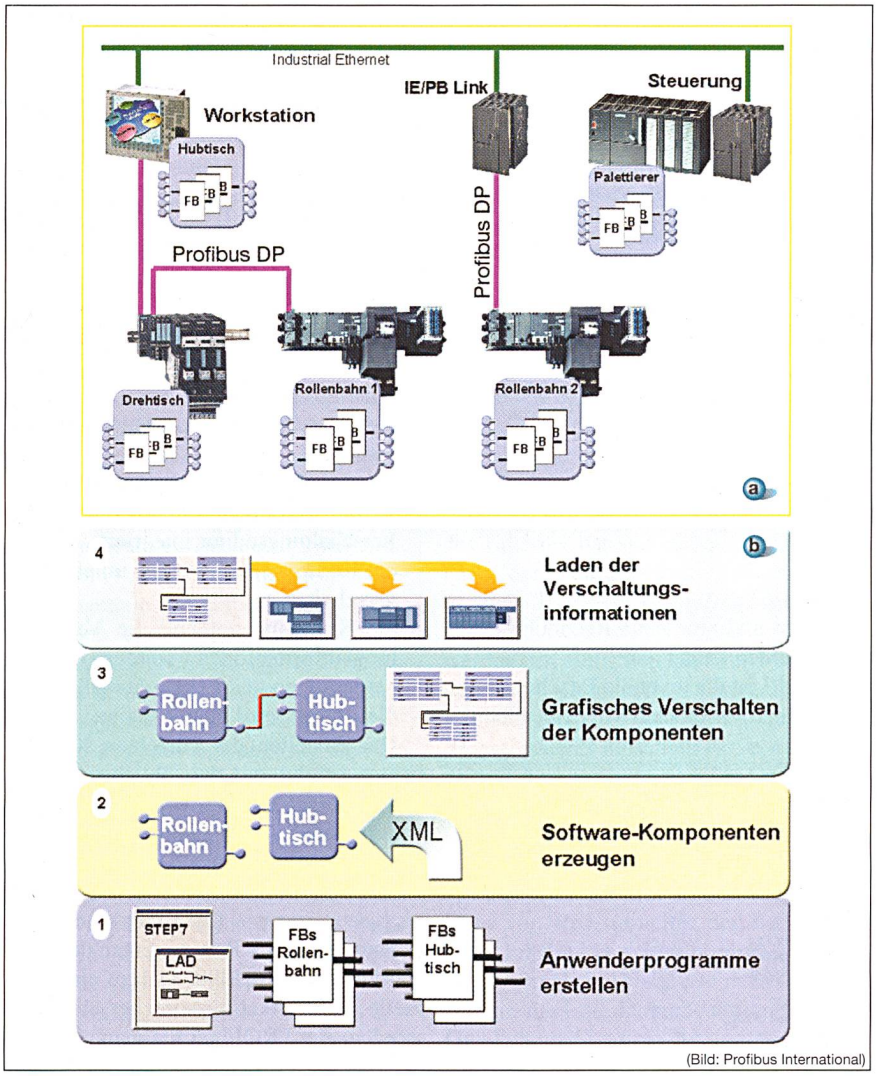
Mit PROFInet kommt nun ein erstes System auf den Markt, in welches diese Konzepte der verteilten Funktionsblöcke implementiert wurden. Eine verteilte Anlage mit PROFInet wird in folgenden Schritten realisiert (Bild 4):

- die einzelnen Komponenten werden erstellt;
- die Schnittstellen der Komponenten werden erzeugt und in einen zentralen Verschaltungseditor integriert;
- die Komponenten werden miteinander verschaltet;
- die Komponenten und die Verschaltungsinformationen werden in die Steuerungen und Feldgeräte geladen;
- die Anlage wird diagnostiziert.

Das Erstellen der einzelnen Komponenten erfolgt mit den üblichen und in der Automatisierungstechnik erprobten Programmiermitteln. Somit können ganze Anlageteile oder auch nur bestehende Funktionsblöcke weiter verwendet und eingesetzt werden. Dazu müssen nur die Schnittstellen der Komponenten definiert werden. In IEC 61499 ist dazu eine spezielle Spezifikationssprache definiert worden. Bei PROFInet hat man sich entschlossen, anstelle dieser Beschreibungssprache das aus der Internettechnologie bekannte XML zu verwenden. Dabei wird ein Style Sheet für die Beschreibung der Komponenten verwendet. Kommerzielle Programmierwerkzeuge können diese Komponentenbeschreibungen automatisch erzeugen. Ist dies nicht der Fall, besteht ein einfacher Editor, der die Erstellung der XML-Datei unterstützt. Dieses Erstellen der Komponentenbeschreibung von Hand ist aber ein fehlerträchtiger Prozess und sollte vermieden werden.

Die Beschreibungen der Komponenten in XML werden nun in Form von Bibliotheken in einen Verschaltungseditor geladen. Da diese Komponentenbeschreibungen unabhängig von Herstellern und Betriebssystemen sind, können die Werkzeuge der Komponentenerstellung und der Verschaltung beliebig gemischt werden. Im Moment sind Produkte der Firmen Siemens und Hilscher auf dem Markt verfügbar.

Es wird bei der Planung immer unterschieden zwischen der physikalischen Netzwerksicht und dem Zusammenhang der einzelnen Funktionen. Ein Gerät



(Bild: Profibus International)

Bild 4 Umsetzungsbeispiel für PROFInet

Bild a: Beispielskonfiguration einer Anlage; Bild b: Vorgehensschritte bei der Anlageplanung.

muss eine eindeutige Adresse aufweisen, kann aber mehrere Funktionsblöcke beherbergen.

Im Verschaltungseditor werden nun die Verbindungen zwischen den Komponenten geplant. Bei PROFInet ist dabei – in Abweichung zum Standard IEC 61499 – auf die explizite Definition von Ereignissen verzichtet worden. Man will damit die Bedienung der Verschaltung durch den Anlageplaner vereinfachen und ihn so bei dieser Aufgabe entlasten. Allerdings verliert der Anlageplaner damit ganz klar die Möglichkeit, die Reihenfolge der Ausführung in den einzelnen Komponenten explizit zu kontrollieren. Bei Bedarf können aber diese Ereignisein- und -ausgänge durch den Ersteller der Komponenten explizit angefügt werden, da er prinzipiell frei ist, Ereignisse zur Steuerung der Komponente als Eingang und Ausgang festzulegen, auch wenn dies in der PROFInet-Richtlinie nicht explizit verlangt wird.

Bei der Verschaltung müssen verschiedene Regeln eingehalten werden. Es ist beispielsweise zwingend notwendig, dass alle Eingänge definiert sind und entweder verschaltet oder mit Werten vorbelegt sind. Dabei darf ein Eingang nur von einem einzigen Ausgang abhängen. Müssen mehrere Ausgänge zu einem Eingang zusammengefasst werden, so ist dies mit einer getrennten Komponente zu realisieren. Ein Ausgang darf aber für mehrere Eingänge verwendet werden.

Als Ersatz für die Ausführungskontrolle der Komponenten ist bei PROFInet eine Quality of Service (QoS) für die einzelnen Verbindungen eingeführt worden. In dieser QoS kann festgelegt werden, in welchen Intervallen die einzelnen Werte über eine Verbindung durch das darunter liegende Kommunikationssystem übertragen werden sollen.

Wichtig für die ganze Funktionalität einer verteilten Automatisierung ist die Möglichkeit, die Funktionen der Komponenten in die einzelnen Geräte laden zu können. Damit wird es möglich, von zentraler Stelle die komplette Funktionalität einer Anlage auszuwechseln, ohne sich explizit zu jedem Gerät in der Anlage begeben zu müssen. Bei bestimmten Geräten macht das Auswechseln einer Funktion keinen Sinn, da auf Grund der Gegebenheiten der Hardware keine anderen Funktionen ausgeführt werden können. In diesem Fall wird das Laden der Funktionen in eine Komponente auch nicht unterstützt.

Die Verschaltungsinformation muss aber in jedem Fall geladen werden können. Nur wenn dynamisch festgelegt werden kann, welcher Ausgang mit welchem Eingang verbunden ist, kann die Anlage sinnvoll umgebaut werden.

Die Diagnose ist bei der Inbetriebnahme wichtig. Auf einfache Art und Weise müssen die Funktionen und Verbindungen der Komponenten kontrolliert und überprüft werden können. Eine allgemein verständliche Systembeschreibung findet sich in [6].

Praktische Erfahrungen

Mit dieser neuen Technologie müssen praktische Erfahrungen gesammelt werden. Das Netzwerk EDiSoN[6)] baut bis Ende 2003 mit Unterstützung des Bundesprogramms Softnet[7)] die entsprechenden Kompetenzen auf, um die kleinen und mittleren Unternehmen in der Schweiz mit dieser Technologie zu unterstützen. An den Fachhochschulen in Bern (BFH[8)]), Winterthur (ZHW[9)]) und Manno (SUPSI[10)]) werden entsprechende PROFInet-Kompetenzzentren entstehen.

Veranstaltung

Am 23. September 2003 findet an der ZHW in Winterthur eine Tagung der Fachgruppe Offene Bus-Systeme (FOBS[11)]) von Electrosuisse zu diesem Thema statt (http://www.electrosuisse.ch). Neben einer etwas ausführlicheren Darstellung der Grundlagen sollen auch erste Beispiele und Erfahrungsberichte aus der Praxis aufgezeigt werden.

Referenzen

[1] *G. Enée, C. Escazut:* A Minimal Model of Communication for Multi-Agent Systems. 8th IEEE International Conference on Emerging technologies and Factory Automation, 15.–18. Oktober 2001, Antibes – Juan les Pins, Frankreich, Seite 11.

[2] *T. Sauter, P. Massote:* Enhancement of Distributed Production Systems through Software Agents. 8th IEEE International Conference on Emerging technologies and Factory Automation, 15.–18. Oktober 2001, Antibes – Juan les Pins, Frankreich, Seite 267.

[3] IEC 61131-3: Programmable controllers – Part 3: Programming languages.
[4] IEC 61499-1:2000: Function blocks for industrial-process measurement and control systems – Part 1: Architecture.
[5] IEC 61499-2:2000: Function blocks for industrial-process measurement and control systems – Part 2: Software tools requirements.
[6] Profibus International: PROFInet Systembeschreibung. Draft Version vom Oktober 2002, kann kostenlos herunterladen werden unter www.profibus.com.
[7] *L. Robert:* Modelling control systems using IEC 61499, Applying function blocks to distributed systems. Control Engineering Series 59, The Institution of Electrical Engineers, London 2001

Angaben zum Autor

Prof. ***Max Felser*** lehrt an der Hochschule für Technik und Informatik der Berner Fachhochschule und beschäftigt sich seit über 10 Jahren speziell mit Feldbussen und der Programmierung von verteilten Systemen.
Berner Hochschule für Technik und Architektur Bern, CH-3014 Bern, max.felser@hta-be.bfh.ch

[1] Für die Definition der Strukturen eines Feldgerätes werden heute in den verschiedenen Normen und Richtlinien drei unterschiedliche Verfahren eingesetzt: Verwenden von Parameterlisten, Funktionsblöcken oder Objekten. *Parameterlisten*: Jedes Feldgerät verfügt über eine Liste von Parametern, die einen bestimmten Umfang und eine bestimmten Bedeutung haben und die Funktionen des Feldgerätes beschreiben. Sie können gelesen oder geschrieben werden. *Funktionsblöcke*: Ein Feldgerät wird in verschiedene Funktionsblöcke aufgeteilt. Jeder Funktionsblock hat Ein- und Ausgänge, einen Zustand, einen Satz von Parametern und Funktionen zur Verarbeitung der Eingangs- zu Ausgangsdaten. Die Beschreibung von Feldgeräten mit Funktionsblöcken ist vor allem in der Prozessautomation sehr beliebt. *Objekte*: Sie kapseln die Daten und die dazugehörenden Methoden. Methoden können mit Funktionen verglichen werden. Objekte haben eine Instanz, d.h. den Daten wird ein bestimmter Speicher zugewiesen. Die Eigenschaften der Objekte können von Klasse zu Klasse vererbt werden.

[2] Ein Kommunikationssystem besteht aus verschiedenen *Knoten*, die durch ein Netzwerk verbunden werden. In einer Automatisierungsanwendung sind die einzelnen Knoten z.B. Feldgeräte, Steuerungen oder Engineeringstationen. Ein Knoten kann mehrere *Ressourcen* in der Form von Prozessoren und Speicher umfassen. Die zu erfüllenden Aufgaben können in Funktionen aufgeteilt werden. Zur Ausführung müssen die einzelnen Funktionen den Ressourcen zugeordnet werden. Somit kann eine auf mehrere Ressourcen und eine in verschiedene Funktionen aufgeteilte Aufgabe auch auf mehrere Knoten in einem Netzwerk verteilt werden.

[3] TCP: Transmission Control Protocol

[4] Coil: engl. für Spule. Ein Coil stellt ein binäres Objekt dar.

[5] Im normierten Referenzmodell der offenen Schnittstellen für Kommunikationssysteme werden die Funktionen einer Schnittstelle eines Knotens in sieben hierarchisch gegliederte Schichten (engl. Layer) aufgeteilt. Jeder Layer hat dabei bestimmte Aufgaben und stellt der darüber liegenden Schicht bestimmte Dienste zur Verfügung. Für Feldbusse ist dieses Referenzmodell auf drei Schichten reduziert worden. Der Anwender (engl. User) eines solchen Kommunikationssystems verwendet somit immer nur die Dienste der obersten Schicht – der Anwendungsschicht. Bekannte Protokolle der Anwendungsschicht sind HTTP (Hyper Text Transfer Protocol) für das Lesen von Texten oder SMTP (Simple Mail Transfer Protocol) zum Versenden von elektronischer Post. Für die Automatisierung ist in der Norm ISO 9506 ein Modell von virtuellen Fertigungsgeräten als MMS (Manufacturing Message Specification) definiert worden. Diese virtuellen Fertigungsgeräte weisen verschiedene Objekte wie z.B. Variablen, Programme oder Alarme auf. Mit definierten Funktionen können solche Variablen gelesen und geschrieben, Programme gestartet oder gestoppt und Alarme gemeldet und quittiert werden. Diese Norm wird nur selten eingesetzt, diente aber praktisch allen Feldbussen als Vorlage für ihre speziellen Normen.

[6] EDiSoN: Embedded and Distributed Solutions Network. EDiSoN fördert auf dem Gebiet der verteilten eingebetteten Systeme die Vernetzung von Bildungs- und Forschungsinstitutionen auf Hochschulstufe sowie mit industriellen Unternehmen und fördert damit aktiv deren Wettbewerbsfähigkeit. http://www.edison.ch

[7] Softnet: Aktionsprogramm des Bundesamts für Berufsbildung und Technologie (BBT). Ziel des Programms ist der Aufbau einer starken Software-Szene in der Schweiz, die auch auf internationalen Märkten aktiv und erfolgreich ist. http://www.softnet.ch

[8] http://www.bfh.ch

[9] http://www.zhwin.ch

[10] SUPSI: Scuola Universitaria Professionale della Svizzera Italiana. http://www.supsi.ch

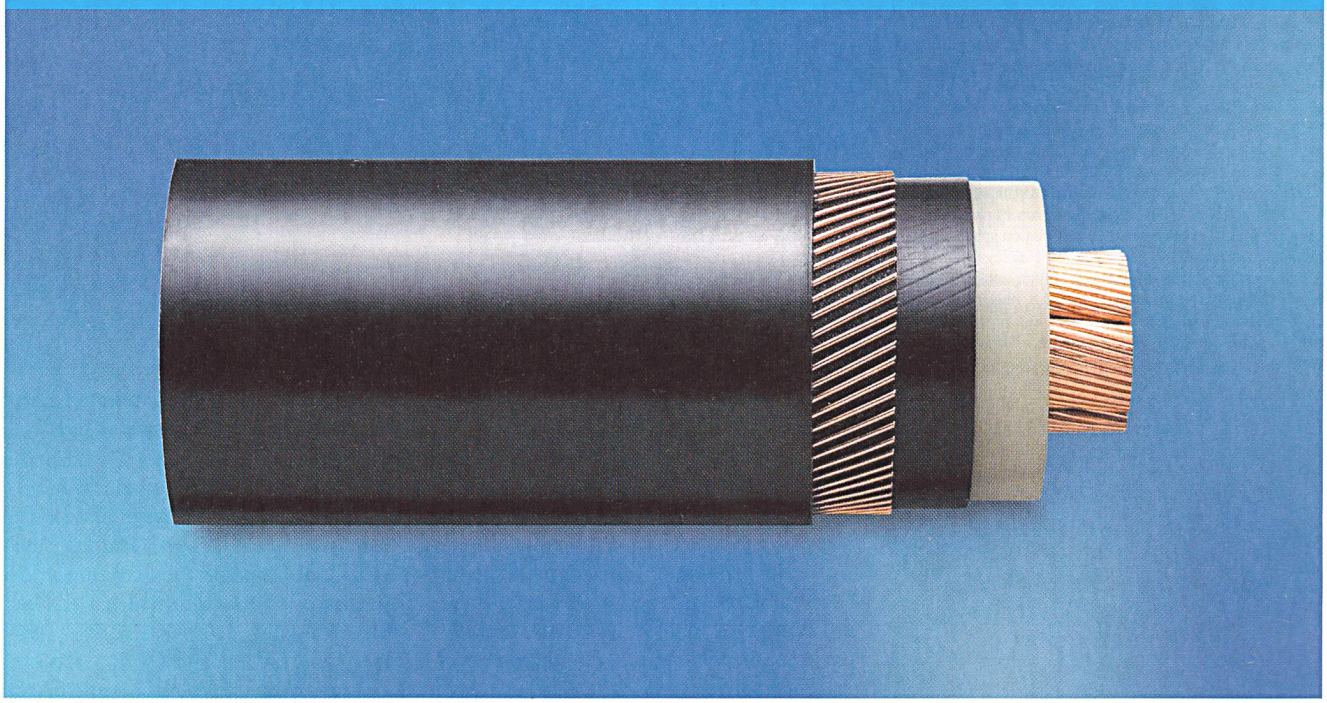
[11] FOBS: Fachgruppe Offene Bus-Systeme. Fachgruppe zur Förderung des Erfahrungsaustausches innerhalb der schweizerischen Industrie auf dem Gebiet der offenen Bus-Systeme. Der Begriff Offene Bus-Systeme wird weit gefasst, insbesondere werden nicht nur Bus-Systeme im engeren Sinn verstanden, auch andere Topologien werden behandelt.

Distributed Automation Systems

Programmation commune à différents fabricants pour pilotages séparés dans le domaine de la construction d'installations

La situation économique actuelle et l'énorme pression au niveau des coûts pesant sur la branche de l'automation requièrent de nouvelles solutions pour réduire encore davantage les coûts. Or, seules les technologies permettant des baisses de coûts substantielles sont effectivement mises en œuvre. Les blocs de fonction définis dans la norme IEC 61499 représentent une possibilité de réduire ces coûts, car ils permettent une programmation commune aux fabricants pour le pilotage de différentes installations. PROFInet est désormais le premier produit commercialisé qui offre la possibilité de réunir les composants provenant de fabricants différents dans une même installation.

Eine Erfindung von Brugg: Energiekabel, die auch 400 kV nicht aus der Ruhe bringen.



Lassen Sie sich an der Ineltec vom 2. bis 5. September in Halle 1.0, Stand D56 in die Welt von Brugg Cables entführen. Wir zeigen Ihnen gerne, dass wir nicht nur Kabel und Zubehör für die Anwendungsgebiete von 1 bis 420 kV, den Telekommunikations- und Industriebereich anbieten, sondern auch mit kundenspezifischen Lösungen und einem ausgezeichneten Service eine gute Figur machen. Nehmen Sie mit uns Kontakt auf.

Brugg Kabel AG, Klosterzelgstr. 28, CH-5201 Brugg,
Tel. +41 (0) 56 460 33 33, Fax +41 (0) 56 460 35 36,
info@brugg.com www.brugg.com

BRUGG CABLES
Well connected.