Methoden

Christian Diedrich*, Alexander Bieliaiev, Jürgen Bock, Andreas Gössling, Rolf Hänisch, Andreas Kraft, Florian Pethig, Oliver Niggemann, Johannes Reich, Friedrich Vollmar und Jörg Wende

Interaktionsmodell für Industrie 4.0 Komponenten

Interaction model for I4.0 components

DOI 10.1515/auto-2016-0118
Eingang 28. September 2016; angenommen 6. Dezember 2016

Zusammenfassung: Verwaltungsschalen bilden zusammen mit den Assets der digitalen Fabrik I4.0-Komponenten. Interaktionen zwischen den Verwaltungsschalen sind wichtige Bestandteile der Wertschöpfungsketten in den I4.0-Systemen. Dafür benötigen die Verwaltungsschalen eine gemeinsame Sprache. Auf der Basis der Festlegungen der DIN SPEC 91345, d. h. des RAMIs und der Struktur der Verwaltungsschale, wird hier das Interaktionskonzept beschrieben.

Schlüsselwörter: Industrie 4.0, I4.0-Komponente, Interagierende Automaten.

Abstract: Asset Administration Shells (AAS) and assets form I4.0 components in a digital factory. Interactions between Asset Administration Shells are important elements of process chains. The Asset Administration Shells need a common language to understand each other. This paper proposes an interaction model based on DIN SPEC 91345 (RAMI4.0).

***Korrespondenzautor: Christian Diedrich,** Otto-von-Guericke-Universität Magdeburg, Magdeburg,
E-Mail: Christian.Diedrich@ovgu.de
Alexander Bieliaiev: Otto-von-Guericke-Universität Magdeburg, Magdeburg
Jürgen Bock: KUKA Roboter GmbH, Augsburg
Andreas Gössling: Hilscher Gesellschaft für Systemautomation mbH, Hattersheim
Rolf Hänisch: Fraunhofer-Institut für Offene Kommunikationssysteme FOKUS, Berlin
Andreas Kraft: DEUTSCHE TELEKOM AGT-Labs (Research & Innovation), Berlin
Florian Pethig: Fraunhofer-Anwendungszentrum Industrial Automation (IOSB-INA), Lemgo
Oliver Niggemann: Hochschule Ostwestfalen Lippe, Lemgo
Johannes Reich: SAP, Walldorf
Friedrich Vollmar: Consultant, Frankfurt/Main
Jörg Wende: IBM Deutschland GmbH, Dresden

Keywords: Industry 4.0, I4.0 component, digital factory, interacting automata.

1 Motivation

Industrie 4.0 (I4.0) ist ein Sammelbegriff für Technologien, die eine neue Gestaltung der Wertschöpfungskette von Unternehmen ermöglichen. Zu diesen Technologien zählen beispielsweise Cyber-Physische-Systeme (CPS) und das Internet of Things (IoT) [1]. CPS sind hierbei Systeme, die durch die Verschmelzung der physischen und virtuellen Welt entstehen [2]. Physische Systeme, wie beispielsweise ein elektrischer Antrieb, erhalten eine virtuelle Repräsentation, um über das IoT mit anderen Komponenten kommunizieren und kooperieren zu können. Ein Anwendungsbeispiel ist die Smart Factory, in der der Mensch zukünftig durch über das IoT kooperierende CPS in Form kontext-sensitiver Assistenzsysteme unterstützt wird. Derartige I4.0-Systeme setzen Interaktionsfähigkeit der verwendeten Komponenten voraus und basieren auf den Entwurfsprinzipien Interoperabilität, Virtualisierung, Dezentralisierung, Echtzeitfähigkeit, Serviceorientierung und Modularität [1]. Um diese zu gewährleisten, wurde die Plattform I4.0 gegründet, um in Zusammenarbeit mit Verbände und Organisationen (ZVEI, VDI, VDE, DKE) I4.0-Standards zu entwickeln. Als erstes Ergebnis wurde das Referenzarchitekturmodell Industrie 4.0 (RAMI 4.0) [3] vorgestellt.

2 Interaktionskonzept

2.1 Ausgangspunkt

Eine der wesentlichen Charakteristika von I4.0-Systemen ist es, dass Assets als I4.0-Komponenten repräsentiert werden und direkt miteinander in Kontakt treten, um

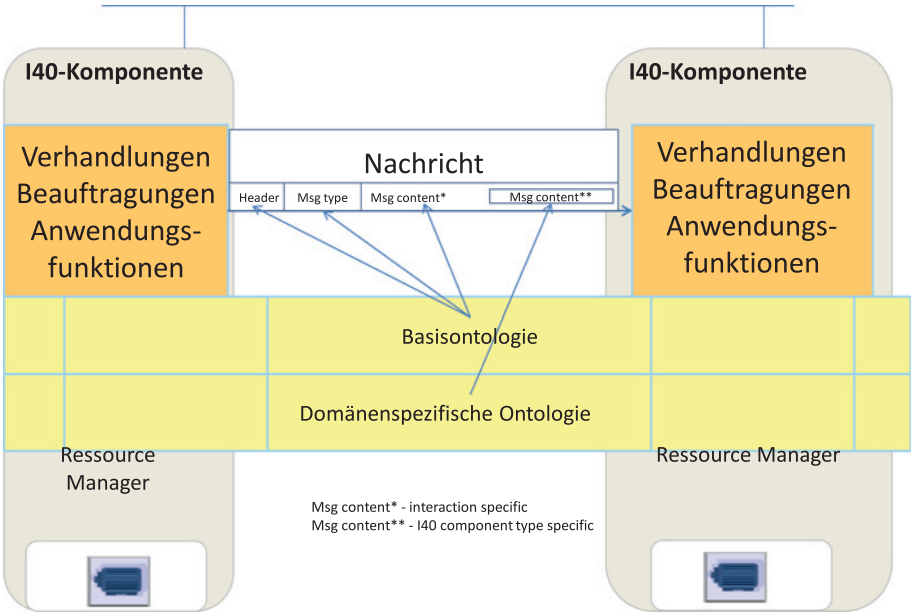

Abbildung 1: Einordnung der Interaktion in die I4.0-Komponente.

Aufgaben in Wertschöpfungsketten auszuführen. Dazu bedarf es spezieller Interaktionsmuster.

Hierfür bietet die Verwaltungsschale (Asset Administration Shell, AAS [3] Def. 3.12) von I4.0-Komponenten eine Reihe von Daten und Funktionalitäten an. Die Funktionen der I4.0-Komponenten sind sogenannten Teilmodellen zugeordnet (Asset Administration Shell, AAS [3]). Beispiele für Teilmodelle sind Bohren oder Transportieren, Verhandeln und Vereinbarungen abschließen, einen Produktionsprozess steuern (z. B. durch PackML), Diagnostizieren, Gerätetausch und viele andere mehr. Die benannten Funktionen können aus mehreren Funktionen zusammengesetzt werden.

Das prinzipielle Konzept sieht vor, dass I4.0-Komponenten Nachrichten austauschen, die auf das Verhalten der I4.0-Komponenten einwirken können (Abbildung 1). Die Nachrichtenelemente sind Teil einer, für Basisinteraktionen notwendigen Basisontologie und den Domänenontologien, die für die einzelnen Typen von I4.0-Komponenten existieren oder aus den Domänen entstehen. Die Ontologieinhalte sind im Manifest verzeichnet. Die Ontologien sind im I4.0-System bekannt und eindeutig. Ontologien können auf Taxonomien, bzw. Merkmalskatalogen wie z. B. ecl@ss basieren oder bei Notwendigkeit weitere technologische Konzepte nutzen.

Die Verwaltungsschale (Asset Administration Shell, AAS [3] Def. 3.12) von I4.0-Komponenten bietet eine Reihe von Daten und Funktionalitäten an. Die Funktionen der I4.0-Komponenten sind sogenannten Teilmodellen zugeordnet (Asset Administration Shell, AAS [3]). Beispiele für Teilmodelle sind Bohren oder Transportieren, Verhandeln und Vereinbarungen abschließen, einen Produktionsprozess steuern (z. B. durch PackML), Diagnostizieren, Gerätetausch und viele andere mehr. Die benannten Funktionen können aus mehreren Funktionen zusammengesetzt werden.

2.2 Einordnung des Interaktionsmodells

Die Funktionen der Assets der I4.0-Komponente werden über das Interaktionsmodell zugreifbar gemacht. Für RAMI 4.0 kommen objekt- und protokollorientierte Architekturen in Frage. Beide Technologien können die Grundlage für die Interaktion zwischen I4.0-Komponenten bilden. Die SOA-Architektur und insbesondere ihr Vertreter OPC UA hat bereits breite Akzeptanz in der IT und Automatisierungstechnik gefunden. Dieser Architekturansatz wird in der GMA 7.21 bearbeitet, siehe hierzu „Industrie 4.0 Service Architecture“ [13]. Zur gleichberechtigten Zusammenarbeit zwischen I4.0-Komponenten, bei der eine ergebnisoffene Verhandlung, erforderlich ist, wird eine protokollorientierte Interaktion benötigt. Diese beiden Prinzipien bilden die Modellgrundlage für die Interaktionen (Abschnitt 4.2), die wie folgt zu charakterisieren sind:

- objekt-/dienstorientiert: In den Verwaltungsschalen werden Objekte bereitgestellt, die die Funktionalität durch Operationen zugreifbar machen
  - Dienstsystem, z. B. SOA-basiert
    - synchron
    - hierarchisch

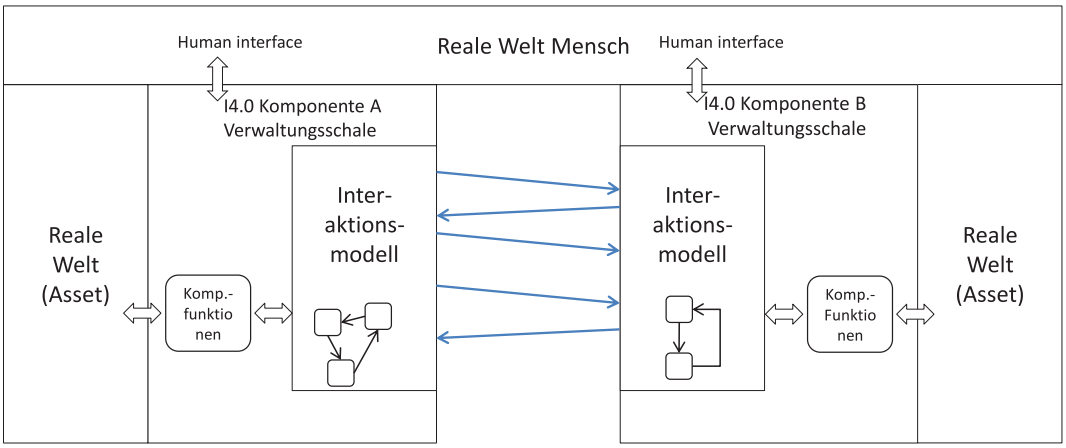

Abbildung 2: Interaktionszwischen I4.0-Komponenten.

  - Client/Server
  - enge Kopplung
- geeignet für Kommunikations- Informations- und Plattformdienste
- protokollorientiert: Abstraktion der Funktionalität durch Automaten
  - Protokollautomaten
    - asynchron
    - horizontal
    - peer-to-peer
    - lose Kopplung
  - geeignet für Anwendungsaufgaben z. B. Verhandlung, komplexere Maschinensteuerungen (z. B. nach IEC 61512-1/PackML)

Im Folgenden wird nur auf die protokollorientierte Interaktion eingegangen.

Die Assets werden durch Funktionen in der Verwaltungsschale steuerbar (Abbildung 2). Das Interaktionsmodell verwendet Nachrichten zwischen mindestens zwei I4.0-Komponenten, die Zustandsübergänge in der I4.0-Komponente hervorrufen können. Diese Zustandsmaschine ist eine Projektion des Teils der Funktionalität des Assets, der für die Zusammenarbeit mit den Partnern erforderlich ist. Das Modell beruht auf gekoppelten Automaten, das Modell ist in Abschnitt 4 beschrieben.

Die funktionalen und nicht-funktionalen Eigenschaften der I4.0-Komponenten sind für Nutzer auslesbar bereitzustellen. Dazu gibt es in jeder I4.0-Komponente ein sogenanntes Manifest, in dem die Beschreibungen der Funktionen und Daten abrufbar bereitgestellt werden. Es wird favorisiert, dass das Manifest auf der Basis von Merkmalen, Merkmalsausprägungen, Parametern, Eigenschaften und ihre Beziehungen zueinander und anderen Komponenten oder deren logische Abstraktion (beispielsweise „Transportmittel“ oder „elektrischer Verbraucher“) aufgebaut wird. Merkmalbeschreibungen haben eine festgelegte Struktur und bieten für den Nutzer einen verlässlichen Informationsumfang.

2.3 Prinzipieller Ablauf von Interaktionsmustern

Mit einer Folge von Nachrichten, die zwischen den I4.0-Komponenten ausgetauscht werden, entsteht eine Interaktion zwischen diesen. Eine Interaktion hat den Zweck, dass die handelnden Akteure, d. h. die I4.0-Komponenten aufeinander einwirken, um gemeinschaftlich eine Aufgabe zu erledigen. Die detaillierte Beschreibung der Nachrichten und der Akteure, die Abfolge der Nachrichten und die Aktivierung von Aktionen bzw. anderer verhaltensaktivierender Prozesse müssen dafür gegenseitig verabredet sein.

Ein Beispiel soll dies kurz verdeutlichen. Das Beispiel ist nur zum prinzipiellen Verständnis aufgeführt, die Festlegungen sind noch in der Diskussion. Für den Fall der Initiierung einer Aufgabe, die auf einer bereits vorhandenen Vereinbarung beruht, fragt eine I4.0-Komponente bei der gewünschten I4.0-Komponente an (Abbildung 3). In der angefragten Komponente wird geprüft, ob diese Vereinbarung existiert und aktiviert werden darf. Im Manifest sind die für diese Aufgabe relevanten Merkmale hinterlegt. Ist dies der Fall, so wird eine entsprechende positive Antwort gegeben, zusammen mit den für die Aufgabe benötigten Daten. Ist dies nicht der Fall wird eine entsprechend negative Reaktion abgesetzt mit Angabe einer Fehlerkennzeichnung.

Die Merkmale in der Anfrage und die im Manifest hinterlegten Merkmale sind gleich. Sie sind durch eindeutige

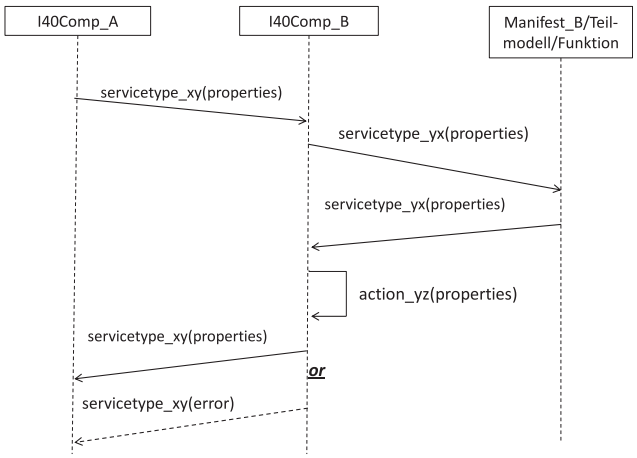

Abbildung 3: Beispielhaftes Interaktionsmuster.

Identifier benannt. Die Werte der Merkmale in der Anfrage repräsentieren Anforderungen und die im Manifest hinterlegten Werte sind Zusicherungen. Merkmale mit demselben Identifier können also unterschiedliche Ausprägungsaussagen haben. Dies muss bei einer Überprüfung berücksichtigt werden, indem es Regeln gibt, die die Erfüllung der Anforderung durch die Zusicherung überprüfen. Im einfachsten Fall muss Gleichheit festgestellt werden. Es sind aber auch Fälle vorstellbar, bei denen Gültigkeitsbereiche abgeprüft werden oder sogar logische und mathematische Beziehungen zwischen verschiedenen Merkmalen auszuwerten sind. Die Ergebnisse der Regelabarbeitung steuern die konkrete Ausprägung der Interaktionsmuster.

Abstrahiert man das benannte Beispiel, so besteht das Modell aus der Definition von interagierenden Partnern (hier I4.0-Komponenten, Manifest, Teilmodell), den Nachrichteninhalten (z. B. IDs und Merkmale von Vereinbarungen, Aufträge), den Abläufen (hier repräsentiert durch ein Sequenzdiagramm) und zusätzlichen Regeln, z. B. zur Überprüfung von Anforderung und Zusicherung. Die im Sequenzdiagramm dargestellt Abläufe sind Ausschnitte der interagierenden Automaten, wie in Abschnitt 4 beschrieben. Es entsteht eine Sprache in der Nachrichteninhalte, deren Abfolgen (Interaktionsmuster) und Regeln zur Steuerung von Alternativen im Ablauf definiert sind. Dafür werden formale und semi-formale Modelle und Methoden verwendet, wie z. B. Klassendiagramme, Sequenzdiagramme, Zustandsmaschinen, Klassifikationen und Merkmalkataloge.

In diesem Beitrag, getragen durch die UAG Ontologie der Plattform Industrie 4.0, wird das Konzept einer Sprache für Verhandlungen und Beauftragungen sowie die gegenseitige Nutzung von Anwendungsfunktionen vor I4.0-Komponenten vorgeschlagen. Die Spezifikation legt den Schwerpunkt auf Interaktionen, um Verhandlungen vorzunehmen und um anwendungsbezogene Funktionalitäten zu nutzen, die in Teilmodellen enthalten sind.

3 Verwendete Begriffswelt des Interaktionskonzept für I4.0-Komponenten

Die Beschreibungen in diesem Beitrag basieren auf der DIN SPEC 91345, der Beschreibung des RAMI und der dort im Abschnitt 6 beschriebenen I4.0-Komponente. Die in der DIN SPEC eingeführte Begriffswelt wird hier aufgegriffen und für das vorgestellten Konzepts für die Interaktionen zwischen I4.0-Komponenten verfeinert. Die Begriffswelt wird als Modell in einem Klassendiagramm dargestellt, um eine höher Eindeutigkeit zu gewinnen. Dies ist jedoch nicht mit einer Softwarespezifikation zu verwechseln.

Die Beschreibung unterscheidet prinzipiell zwischen der Typ und der Laufzeitebene (Type Layer und Runtime). Ein Charakteristikum der I4.0-Komponenten ist es, dass sowohl die Typinformationen zugreifbar sind, als auch zur Laufzeit Daten ausgetauscht werden können. I4.0-Systeme bestehen aus I4.0-Komponenten (Abbildung 4 ist das I4.0-System nicht enthalten). Die I4.0-Komponente besteht aus Assets und der Verwaltungsschale (Asset Administration Shell). Die Verwaltungsschale besteht aus Sicht dieser Betrachtung aus dem Interaktionsmanager (die Begriffe ComponentManager und ResourceManager sind synonym), dem Manifest und Teilmodellen (Submodel). Die Verwaltungsschale wird in einen Header und den Body eingeteilt. Im Header ist die Identifikation, die Administration und die Verwaltung der Sichten (View) verankert, die Bestandteil des Manifest sind. Die Merkmale (Properties) sind wesentliche Bestandteile des Manifests, und sind jeweils der Administration und Identifikation oder den Teilmodellen zugeordnet. Diese Merkmale können als Anforderung auftreten, z. B. wenn sie als Anfrage in einer Eingangsnachricht bei einer Verhandlungsanfrage ankommen, oder als Zusicherung, wenn sie einer Ausgangsnachricht mitgegeben werden. Im operativen Betrieb können auch Aktualwerte ausgetauscht werden (Parameter- und Variablenwerte in der Laufzeit). Die Merkmale sind Daten, die entsprechend der Festlegungen von IEC 61360 beschrieben sind.

Es gibt Teilmodelle, die zur Identifikation, Administration und der Verwaltung der Sichten zuständig sind, die Verhandlungen und die produktiven Funktionen der I4.0-Komponente durchführen. Der Interaktionsmanager

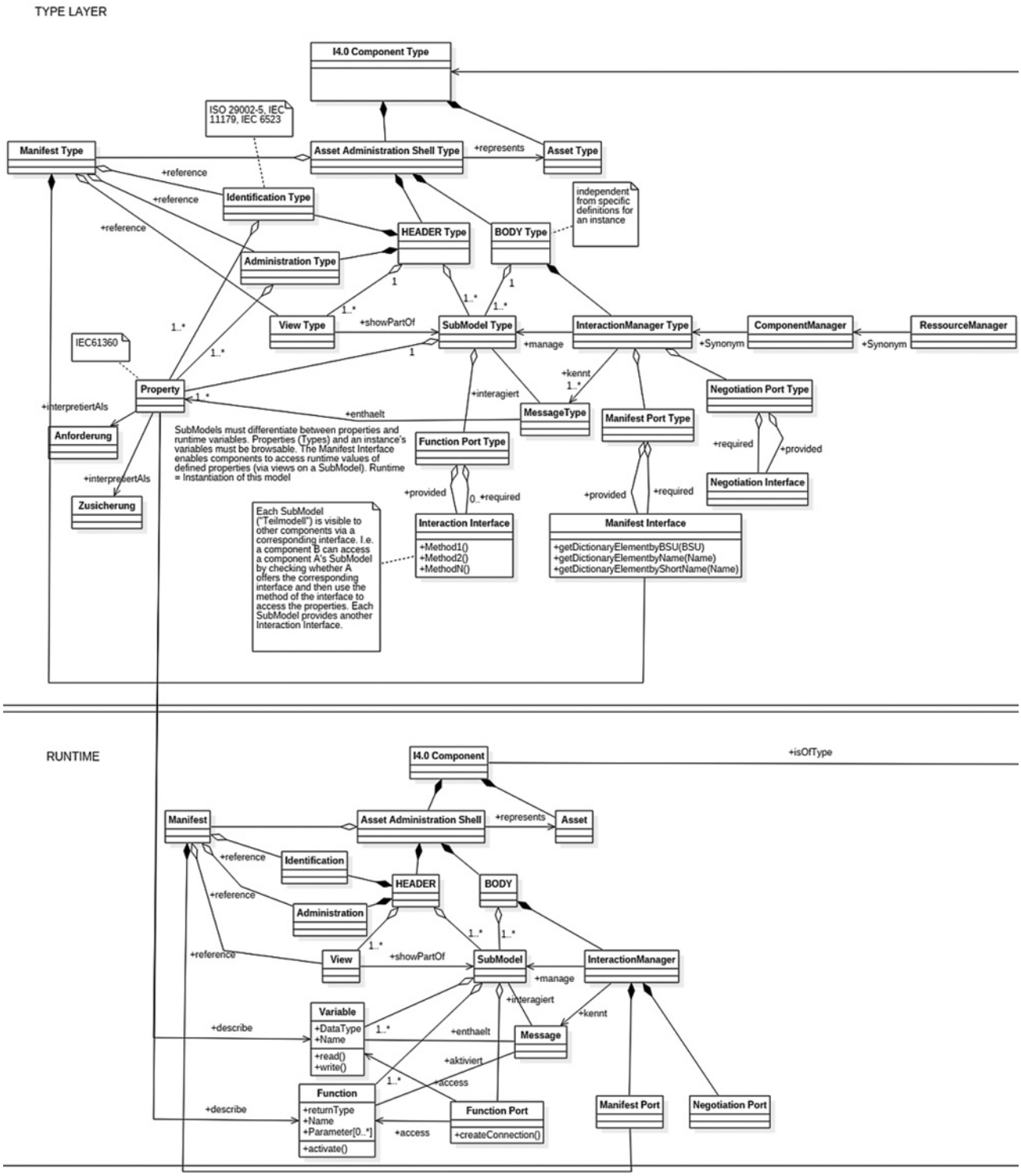

Abbildung 4: Begriffswelt der Verwaltungsschale und des Interaktionskonzepts (Auswahl).

verwaltet die Teilmodelle und ordnet die ein- und ausgehenden Nachrichten (MessageType) den Teilmodellen zu. In den Nachrichten sind die Merkmale in ihren unterschiedlichen Ausprägungen enthalten. Eine Verwaltungsschale kann mehrere Zugangspunkt (hier als Ports bezeichnet) anbieten. Beispielhaft sind hier Ports zu den Informationen des Manifests (Manifest Port Type), zur Verhandlung (Negotiation Port Type) und einen oder mehrere Zugänge zu den Funktionen (Function Port Type). Die Ports sind durch die entsprechenden Interfaces typi-

siert. Es ist möglich, dass über den Function Port Type der Zugang zu den Funktionen einer I4.0-Komponente vorhanden ist, ohne dass zur Laufzeit der Interaktionsmanager einzugreifen hat. Die Merkmale der Typebene beschreiben die Variablen und Funktionen der Laufzeitebene. So geben die I4.0-Komponenten Auskunft über ihre Fähigkeiten.

4 Allgemeines Modell für die Beschreibung der Interaktionsmuster

4.1 Begriffsbestimmung Aktion und Interaktion

Als **Aktion** wird ein Zustandsübergang eines Systems bezeichnet, der sich aus den Eingabewerten, den inneren Zustandswerten vor und nach der Transition sowie den Ausgabewerten zusammensetzt. Als **Interaktion** wird die Kombination von Zustandsübergängen wenigstens zweier Systeme bezeichnet, bei der die Ausgabewerte eines Systems („Sender“) die Eingabewerte des weiteren Systems („Empfänger“) darstellt (Abbildung 5).

Interaktionen lassen sich auf Basis der Art der Informationsverarbeitung von Sender und Empfänger klassifizieren [11]:

- Synchronizität: Der Sender wartet auf die Berechnung des Ergebnisses des Empfängers (synchrones Verhalten des Senders) oder nicht (asynchrones Verhalten).
- Determinismus: Der Zustandsübergang des Empfängers ist aus Sicht des Sender durch die von ihm gesendeten Informationen vollständig festgelegt (deterministisches Verhalten des Empfängers) oder nicht (nichtdeterministisches Verhalten).
- Zustandsbehaftung: Der Zustandsübergang des Empfängers basiert aus Sicht des Senders ausschließlich auf den zuletzt gesendeten Informationen (zustandsloses Verhalten des Empfängers) oder auf zusätzlichen inneren Zustandswerten (zustandsbehaftetes Verhalten).

Werden innerhalb von Interaktionen die Rollen von Sender und Empfänger vertauscht, lassen sich zwei grundsätzlich verschiedene Klassen von Interaktionen unterscheiden: symmetrische und asymmetrische. Bei symmetrischen Interaktionen verhalten sich alle Parteien bezogen auf die drei angeführten Dimensionen auf

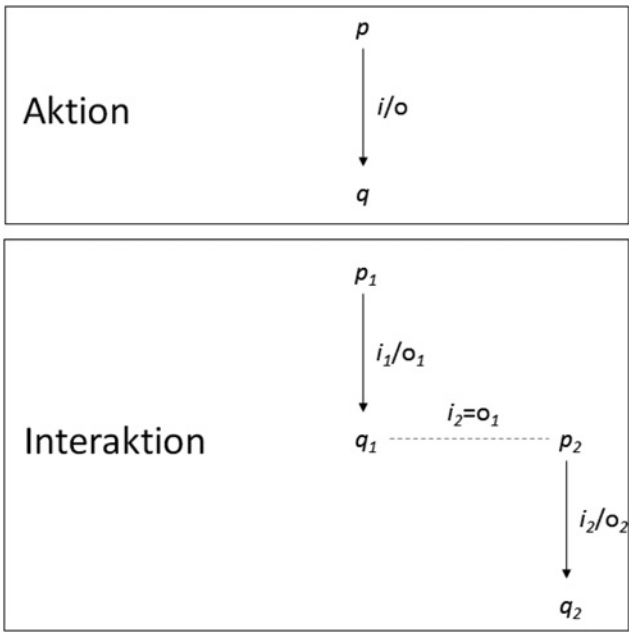

Abbildung 5: Darstellung einer Aktion eines Systems und einer Interaktion zweier Systeme. In einer Interaktion wird die Verbindung zweier Aktionen durch eine „Nachricht“ hergestellt, bei der die Ausgabe der „sendenden“ Transition die Eingabe der „empfangenden“ Transition ist.

dieselbe Art und Weise, bei asymmetrischen Interaktionen verhalten sich die Parteien unterschiedlich.

Ist die Interaktion asymmetrisch, lässt sich der Interaktion bezogen auf die drei angeführten Dimensionen eine semantische Richtung zuschreiben. Ist die Interaktion symmetrisch, lässt sich der Interaktion bezogen auf diese Dimensionen keine Richtung zuschreiben.

Die wichtigste asymmetrische Interaktionsform ist die **„Verwendung“** (Abbildung 6): Die Richtung wird durch den Determinismus vorgegeben. Der Verwender bestimmt über die verwendete Komponente. Umgekehrt ist die Beziehung nichtdeterministisch. Mittels der Verwendungsbeziehung lässt sich die Schichtung einer Anwendung im Sinne des OSI-Modells definieren. Durch sie entsteht aus logischer Sicht eine enge Kopplung im Sinne einer Teil-Ganzes-Beziehung.

Syntaktisch lässt sich der Top-Down-Anteil der Verwendung durch eine objektorientierte Operation beschreiben, d. h. als Funktion, die auf einem definierten Datenkontext operiert. Gegebenenfalls wird ein nichtdeterministischer Teil mittels Ausnahmebehandlung separiert, der dann den Kontrollfluss beeinflusst.

Das heißt auf Grund des Determinismus lässt sich dieser Anteil der Verwendung allein durch die Beschreibung des Verhaltens nur eines Interaktionspartners, des Verwendeten, vollständig erfassen!

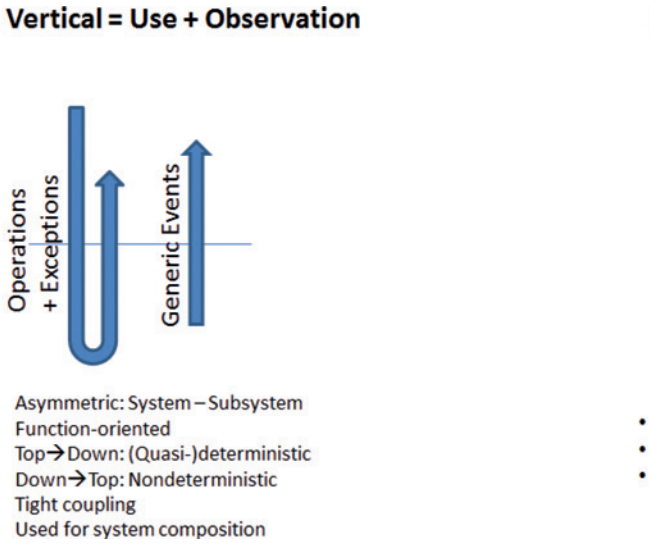

Abbildung 6: Vertikale Interaktionsbeziehung im Sinne einer „Verwendung".

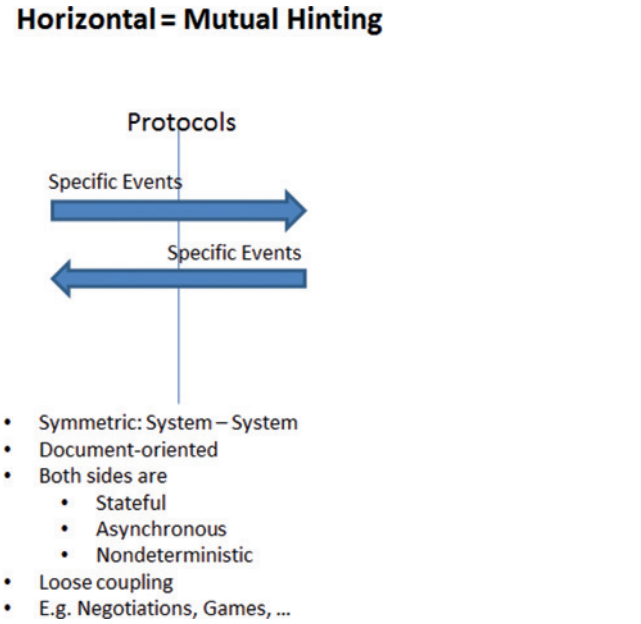

Abbildung 7: Horizontale Interaktionsbeziehung im Sinne eines „Protokolls".

Die wichtigste symmetrische Interaktionsform ist das Protokoll (Abbildung 7). Hier agiert jede Partei im Allgemeinen aus Sicht der anderen Parteien asynchron, nichtdeterministisch und zustandsbehaftet. Auf Grund des Nichtdeterminismus lässt sich den transportierten Informationen in der Regel keine imperative Semantik zuordnen und entsprechend lassen sich diese Interaktionen nicht[1] durch objektorientierte Operationen beschreiben. Deswegen beschränkt sich die Bezeichnung der transportierten Informationen auf ihren „Dokumenten"-Charakter: Der eine Interaktionspartner dokumentiert seinen Zustandsübergang auf eine Art und Weise, dass die davon betroffenen Interaktionspartner ihrerseits ihren Zustandsübergang durchführen können, usw.

Diese Interaktionsform ist im Sinne des Wortes „interaktiv". Im Gegensatz zur Verwendung kommt sie immer dann zum Einsatz, wenn die Unwägbarkeiten zwischen den Interaktionspartnern so groß werden, dass sie sich aus pragmatischer Sicht nicht mehr als Ausnahme separieren lassen, sondern einen wesentlichen Anteil der Beziehung zwischen den Akteuren darstellen. Dies ist immer dann der Fall, wenn die Akteure in weitere Interaktionen auf derselben semantischen Ebene eingebunden werden und gewisse Freiheiten benötigen, um alle ihre Interaktionen zu koordinieren. Eine dieser Interaktionen kann typischerweise auch die Interaktion mit der Umwelt sein. Von einem etwas anderen Blickwinkel kommt diese Interaktionsform zum Einsatz, wenn Applikationen „lose" gekoppelt miteinander interagieren sollen, sie also aus logischer Sicht einen (Groß-)Teil ihres inneren Zustands voreinander verbergen sollen.

Die Interaktionsform des Protokolls ist in ihren Ausdrucksmöglichkeiten einer Sprache sehr ähnlich: Innerhalb von Protokollen lassen sich gewisse Interaktionsmuster identifizieren, die sich sehr intuitiv mit den Begriffen „Bitte", „Notifikation", „Befehl", „Frage", „Auftrag", etc. bezeichnen lassen, innerhalb derer der Empfänger der Informationen in der Regel einen gewissen Interpretationsspielraum hat, die er für seine eigenständigen Entscheidungen auch benötigt.

4.2 Systembildung durch Interaktion

Beim Systembegriff wird davon ausgegangen, dass eine Folge von Eingangswerte durch eine Systemfunktion auf eine Folge von Ausgangswerten abgebildet wird, wobei dabei eine Folge innerer Zustände durchlaufen wird [11, 12].

Basierend auf der Vorstellung, dass Systeme sich durch ihre Systemfunktion konstituieren, kann man die Komposition von diskreten Systemen, die durch Interaktion zustande kommt, systematisieren. Die Kopplung

1 Genau genommen nur schlecht. Man müsste dann Eingabewerte auf Mengen von Ausgabewerten abbilden. Während das in der theoretischen Informatik an manchen Stellen seine Berechtigung hat, führt ein solches Vorgehen praktischer Weise recht schnell in unbeherrschbare Komplexität.

von Systemen geschieht dabei über die gemeinsame I/O-Zustände des Shannon-Kanals.

Die Interaktion zwischen Systemen kann einerseits zur Komposition, also zur Bildung von Supersystemen führen. Im anderen Fall der Interaktion lässt sich von einer Kooperation der Systeme in Folge einer losen Kopplung sprechen in dem Sinne, dass der Zustandsraum der Interaktion deutlich kleiner ist, als der Zustandsraum der beteiligten Aktionen der Systeme.

4.2.1 Komposition diskreter Systeme

Die Komposition von Systemen lässt sich auf die folgenden in Abbildung 8 und Abbildung 9 dargestellten Grundstrukturen zurückführen.

- Die sequentielle und die parallele Kopplung von Systemen hat immer eine Komposition mit Supersystem-Bildung und entsprechend enger Kopplung der Systemelemente zur Folge. Im Automatenbild stößt bei sequentieller Kopplung der Ausgang des sendenden Automaten eine Transition des/der empfangenden Automaten an, ohne das es eine Rückwirkung auf weitere Transitionen des sendenden Automaten gibt. Bei paralleler Kopplung findet ein gemeinsamer Übergang eines Produktautomaten mit einem gemeinsamen Eingabezeichen statt.
- In der „scheinbar rekursive Komposition“ werden drei Einzelsysteme sequenziell komponiert. Der Ausgang des sendenden Automaten stößt eine Transition des empfangenden Automaten an, und dieser wirkt auf ein anderes Subsystem des sendenden Automaten zurück. Dies ist ein sehr wichtiger Fall, weil er die Regel ist für den Methodenaufruf von Objekten.

Hier findet keine „Interaktion“ zwischen dem System „S'“ und S_2 statt (da „S'“ gar kein System ist), sondern S_1, S_2 und S_3 sind jeweils Subsysteme eines Systems $S = S_1 \circ S_2 \circ S_3$, das gar nicht eingezeichnet ist. Dieser Fall macht deutlich, dass ein derartiger Methodenaufruf tatsächlich ein Subsystem des aufrufenden Kontextes adressiert.

- Neben der scheinbar rekursiven Komposition lässt auch die echte Rekursion Supersysteme entstehen. Echte Rekursion gibt es in der Welt der diskreten Funktionen und damit auch in der Welt der diskreten Systeme als Loop- und als While-Konstruktion. In der Loop-Konstruktion verwendet ein System seinen eigenen Ausgang als Eingang in Verbindung mit einem zusätzlichen Zähler. Die Anzahl der Iterationen und damit der Zeitschritt des entstehenden Supersystems ist a priori bekannt. In der While-Konstruktion wird die Abbruchbedingung während der Laufzeit bestimmt und das Ergebnis der Berechnung ist dann der Stand des Zählers. Diese Art der gegenseitigen Komposition ist für Komponenten zu vermeiden, da sie schwer zu handhaben sind. Das bedeutet, dass rekursive Elemente zur Berechnung von diskreten Funktionen in die Komponenten hinein zu verlagern sind. Die Komponenten stellen somit eine Komplexitätsschranke für die Struktur berechenbarer Funktionalität dar.

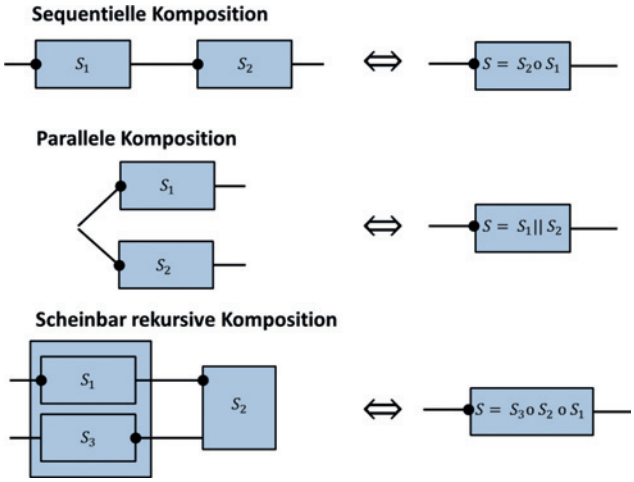

Abbildung 8: Best practice Kompositionsprinzipien.

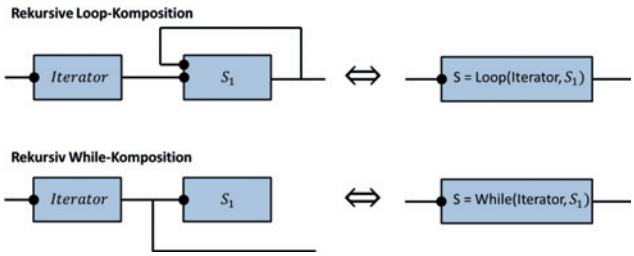

Abbildung 9: Zu vermeidende Kompositionsprinzipien.

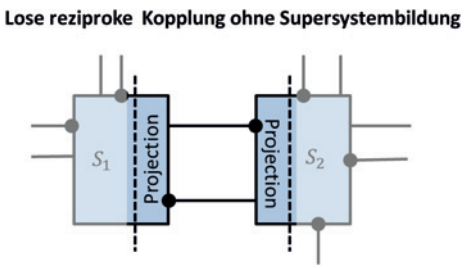

Abbildung 10: Kooperation ohne Supersystembildung.

4.2.2 Kooperation diskreter Systeme

Systeme können auch reziprok interagieren, ohne dass es zur Bildung von Supersystemen kommt, also zu Einheiten, denen man eine Systemfunktion zuordnen könnte (Abbildung 10). Die Supersystembildung lässt sich vermeiden, wenn nur Projektionen von Systemen (auch Rollen genannt) miteinander in Beziehung treten. Diese Projektionen der Systeme agieren als Stellvertreter des Systems und stellen nur einen Teil des Verhaltens der Systemelemente dar. Da das Verhalten der Projektionen tatsächlich weiterhin auch von den restlichen Systemelementen abhängig ist, erscheint das Verhalten des Systems in den Interaktionen nichtdeterministisch aus Sicht des anderen Systems.

4.2.3 Zwei Klassen von Komponenten

Die wesentliche Folgerung aus dem unterschiedlich intendierten Interaktionsverhalten von Systemen ist, dass es 2 unterschiedliche Klassen von Komponenten geben sollte. Die Komponenten der einen Klasse bietet (wiederverwendbare) Funktionalität an und komponieren hierarchisch. Die Komponenten der anderen Klasse bieten nichtdeterministische Schnittstellen an und bilden während der Interaktion keine Supersysteme, sondern interagieren „lose gekoppelt“. Sie bilden horizontale Kooperationsbeziehungen.

Wie weiter unten ausgeführt wird, lassen sich diese beiden unterschiedlichen Interaktionsklassen syntaktisch unterscheiden. Hierarchisch komponierende Komponenten lassen sich objektorientiert beschrieben. Die Interfaces von losen, horizontal koppelnden Komponenten lassen sich mit der Kooperation von Protokoll-Rollen beschreiben.

4.3 Beispiel vertikale und horizontale Interaktion

Die Interaktion löst bei dem empfangenden Interaktionspartner unterschiedliche Zustandsübergänge aus, bzw. kann in Abhängigkeit der aktuellen Zustände unterschiedlich behandelt werden. Deshalb werden als Beschreibungsmodelle Automaten verwendet. Es gibt viele verschiedene Automatenmodelle, z. B. Moore- und Mealy-Automaten und Automaten nach Harel [5]. Für die Wahl eines Automatentypen muss zunächst untersucht werden, welche Art der Kopplung und der Aktionsauslösung bei den I4.0-Interaktionsmustern vorliegen. Dazu soll das folgende Beispiel dienen.

Es sei eine I4.0-Komponente, die einen Roboter ansteuert. Der Roboter kann Transportaufträge ausführen. Der Roboter ist auch eine I4.0-Komponente. Die Anfrage lautet, ein Werkstück mit einem bestimmten Gewicht und einem passenden Greifwerkzeug zu verschiedenen Punkten im 3-dimensionalen Raum zu bewegen. In dem Beispiel wird ein KUKA-Roboter verwendet. Er soll die Aufträge entsprechend der von KUKA angebotenen mxAutomation-Befehle abwickeln können. Der Ausgangspunkt sind die Identitätsprüfung und Security-Vereinbarungen. Alle Interaktionen zur Abarbeitung von Teilaufgaben, wie die Abwicklung der Auftragsanfrage oder die Bearbeitung des Auftrages selbst werden in eigenständigen Sessions eingebettet. Jetzt wird zunächst die Prüfung der Machbarkeit des Auftrages betrachtet. Dazu bietet die I4.0-Komponente einen Dienst „Auftragsanfrage“ an. Dieser setzt sich aus Sequenzen von Einzelaktionen zusammen.

Abbildung 11 zeigt einen Ausschnitt aus dem entsprechenden Verhalten der I4.0-Komponente. Die I4.0-Komponente des Roboters muss in dem geeigneten Zustand sein (ZustandWartenAufAuftragsanfrage). Die notwendigen Vorzustände werden hier nicht weiter betrachtet. Die auslösende Nachricht „Auftragsanfrage“, die über den für den protokollbasierten Dienst vorhandenen Port ankommt, übergibt eine entsprechende Merkmalliste an die I4.0-Komponente-Roboter. Der Inhalt der eingehenden Nachricht wird auf den Transitionseingang „Eingang-Auftragsanfrage“ abgebildet. Die Merkmale werden dem Eingang als Dokument (Merkmalliste) mitgegeben. Dieses Dokument enthält die Anforderungswerte für den Auftrag. Es ist die Entscheidung von der I4.0-Komponente zu treffen, ob die angeforderten Werte (z. B. Produktmasse und anzufahrende Positionen) unter Berücksichtigung des montierten Greiferwerkzeugs erfüllt werden können. Die Entscheidung ist durch die Raute bezeichnet, wie in UML-State-Charts üblich. Das Ergebnis der Entscheidung wird durch die Bedingungen (boolean Expression in „[ ]“) beim Übergang abgefragt. Der Algorithmus sowie die zur Berechnung verfügbaren Zustandswerte dahinter sind eine interne Angelegenheit der I4.0-Komponente[2]. Im Beispiel eines KUKA-Roboters (KR 6 R700 sixx), muss aus den Anforderungsdaten (Werkstückmasse und alle Positionen)

2 Die Autonomie und die Mächtigkeit der eigenen Entscheidungsfähigkeit einer I4.0-Komponente verbergen sich in diesem Algorithmus. Eine logische Entscheidung (wahr oder falsch), d. h. ob ein Auftrag möglich ist oder nicht, kann viele Graduierungsstufen an Wissen über die eigene Komponente oder das diese umgebende System einbeziehen.

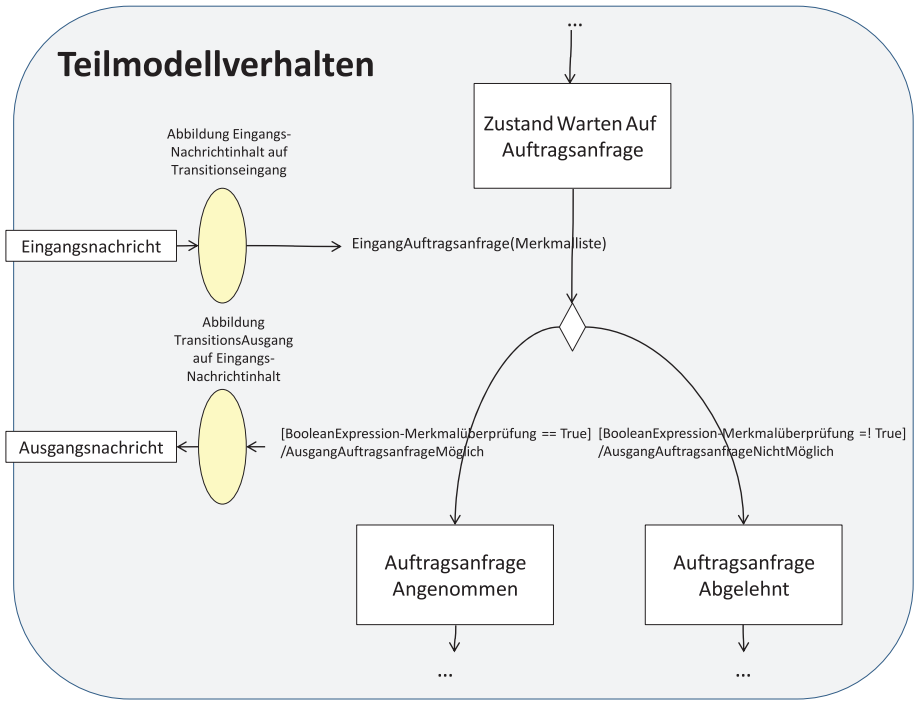

Abbildung 11: Automatenausschnitt Entscheidung über technische Möglichkeit eines Auftrags.

und den Nenndaten des Roboters (Nenn-Traglast: 3 kg, Maximale Traglast: 6 kg, Abstand des Traglastschwerpunkts Lxy: 60 mm und Abstand des Traglastschwerpunkts Lz: 80 mm) die Machbarkeit ermittelt werden. Das Ergebnis wird dem Anfragenden mitgeteilt. Dies geschieht durch den Transitionsausgang (AusgangAuftragsanfrageMöglich/***nichtMöglich). Dieser Ausgang ist in die Ausgangsnachricht des protokollbasierten Dienstes Auftragsanfrage einzubringen. Diese Art der Interaktion wäre ein Protokoll. Der Automat des Auftragsgebers ist aus Platzgründen hier nicht beschrieben.

Eine bestimmte Signatur von Nachrichten, verbunden mit einer bestimmten Struktur von Zuständen und Transitionen mit ihren Bedingungen ist das Charakteristikum für ein Interaktionsmuster.

Sind alle Vorbereitungen für die produktive Aufgabe abgeschlossen, so können direkte Befehle im Rahmen eines Teilmodells erteilt werden. Dafür ist eine neue Session einzurichten und es sind alle Voraussetzungen für die aktiven Bewegungsaufgaben zu erfüllen (z. B. Auftrag muss angenommen sein, Roboter muss im richtigen Betriebsmodus sein, das richtige Greifwerkzeug muss am Roboter vorhanden sein, die unmittelbare Umgebung muss auf die Bewegung des Roboters gefasst sein). Dann ist die I4.0-Komponente des Roboters ist in dem dafür geeigneten Zustand (MotionExecution) (Abbildung 12). In diesem Zustand können alle die Befehle an den Roboter gegeben werden, die die gewünschten Aktionen des Roboters auslösen (symbolisiert durch ExecuteCmd). Diese Zustandsmaschine ist beispielhaft in Abbildung 12 dargestellt. Die Befehle und deren Parameter sind in den eingehenden Nachrichten enthalten und sind die Eingänge an den Transitionen. Die Ausgänge der Transition sind nur intern und gehen direkt an die Steuerung des Roboters. Die Robotersteuerung aktualisiert die Zustände in der projezierten Zustandsmaschine. Diese kann Zustandsübergänge an den aufrufenden Partner weitergeben, d. h. werden Transitionen für den Partner sichtbar, die signifikant über die Auftragsabwicklung Kenntnis geben. Dazu gehören auch mögliche Fehler. Die aufrufende I4.0-Komponente kann sich außerdem über den Zustand der durch die Nachrichten initiierten Aktivitäten informieren.

Diese Art der Interaktion entspricht einem Funktions- oder Prozeduraufruf (auch als Remote Procedure Call – RPC bezeichnet). Das Verhalten dieser Interaktionen werden durch deterministische Zustandsmaschinen beschrieben. Da der Prozess der Befehlsausführung eine Zeitlang dauern kann, wird ein Zwischenzustand (Executing – Abbildung 12) eingenommen, der anzeigt, dass der Prozess nicht abgeschlossen ist. Die verschiedenen Optionen, z. B. es kann erst ein neues Kommando gegeben werden, wenn der Roboter wieder in Standby ist oder der Roboter kann einen Auftragsstapel abarbeiten ist hier nicht dargestellt. Diese Optionen können z. B. durch vorherige Konfiguration des Betriebsmodus in vielen Fällen vorgegeben werden.

Die Ausschnitte aus den beiden Interaktionsmustern des Beispiels zeigen, dass sowohl protokollartige und auch RPC-artige Zustandsmaschinen zum Einsatz

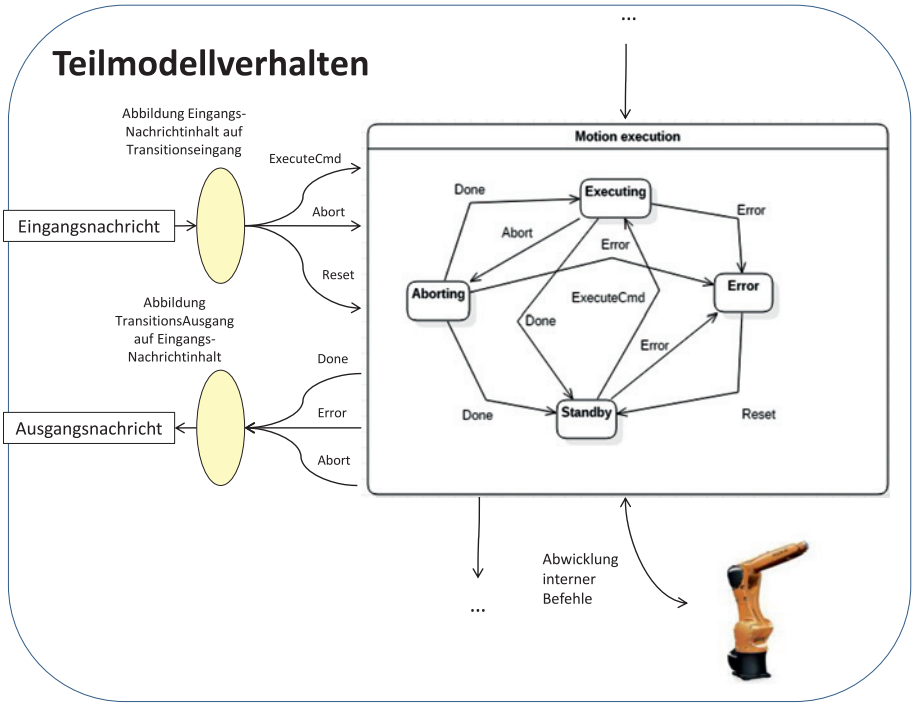

Abbildung 12: Automatenausschnitt Befehlsausführung des Roboters.

kommen müssen. Eine Darstellung der Zustandsmaschinendetails und der Komposition für die Kooperation der I4.0-Komponenten ist in [4] und [10] enthalten.

Wie hier zu erkennen ist, wird die Funktionsverarbeitung der I4.0-Komponente, wie bereits in Abbildung 2 eingeführt, auf Automaten abstrahiert. Außerdem ist ersichtlich, dass die Bedeutung der Nachrichteninhalte durch ihre Verarbeitung bestimmt wird, also aus dem Zustand der I4.0-Empfänger-Komponente, sowie seinem Transformationsverhalten. Um das noch einmal zu betonen, die zu einem bestimmten Zeitpunkt und in einer bestimmten Situation tatsächlich vorhandene Bedeutung einer Nachricht, d. h. die Ausführungsbereitschaft für eine Aktion ist nicht nur von der eindeutigen Identifizierbarkeit der eingehenden Nachricht (d. h. angefragter Aktionstyp und deren Parameter) abhängig, sondern auch von dem Zustand des aufgeforderten Systems und seinem Transformationsverhalten.

4.4 Integration einer höheren Autonomie in die Interaktionen

Eine neue Dimension der Auftragsbearbeitung gewinnen die I4.0-Komponenten durch die beiden Teilaspekte der Abbildung der Eingangsnachricht auf den Eingang an den Transitionen und die Auswertung des Aktualzustands an den Transitionen (Abbildung 13).

Eine Möglichkeit der Abbildung der Eingangsnachricht besteht darin, dass bei nicht vorhandenen direkten Abbildungsregeln durch Techniken des semantischen Webs ein verlinkten Begriffsraum nach Ähnlichkeiten durchsucht wird. Kann eine Zuordnung zu den intern verwendeten Begriffen gefunden werden, so kann die Eingangsnachricht weiter verarbeitet werden. Das erweitert das Verständnis beträchtlich. Der automatisierte Umgang mit solchen echten semantischen Unschärfen erfordert jedoch ausreichende Garantien insbesondere im Bereich der Betriebssicherheit (engl. „Safety“).

Der Zustand einer I4.0-Komponente kann im engen Sinne gefasst werden, ob sich diese selbst in einen Zustand befindet, in dem Aufträge abgearbeitet werden können. Bei einem Roboter könnte dies z. B. bedeuten, dass die Initialisierung erfolgreich abgeschlossen ist und er erwartet Fahrbefehle 1 : 1 auszuführen. Eine erweiterte Erwartungshaltung könnte aber sein, dass der Roboter auch erkennt, dass auf dem angeforderten Weg ein Hindernis zu umfahren ist, oder weitergehend, dass andere Roboter in ihrer Bewegung seinen Bewegungsraum berühren, dass Menschen in den Bewegungsraum treten und anderes mehr. Abstrakt gesprochen steigt der Handlungsspielraum mit der Verfügbarkeit eines Umgebungs- oder Weltbildes vom System, in dem die I4.0-Komponente eingebunden ist.

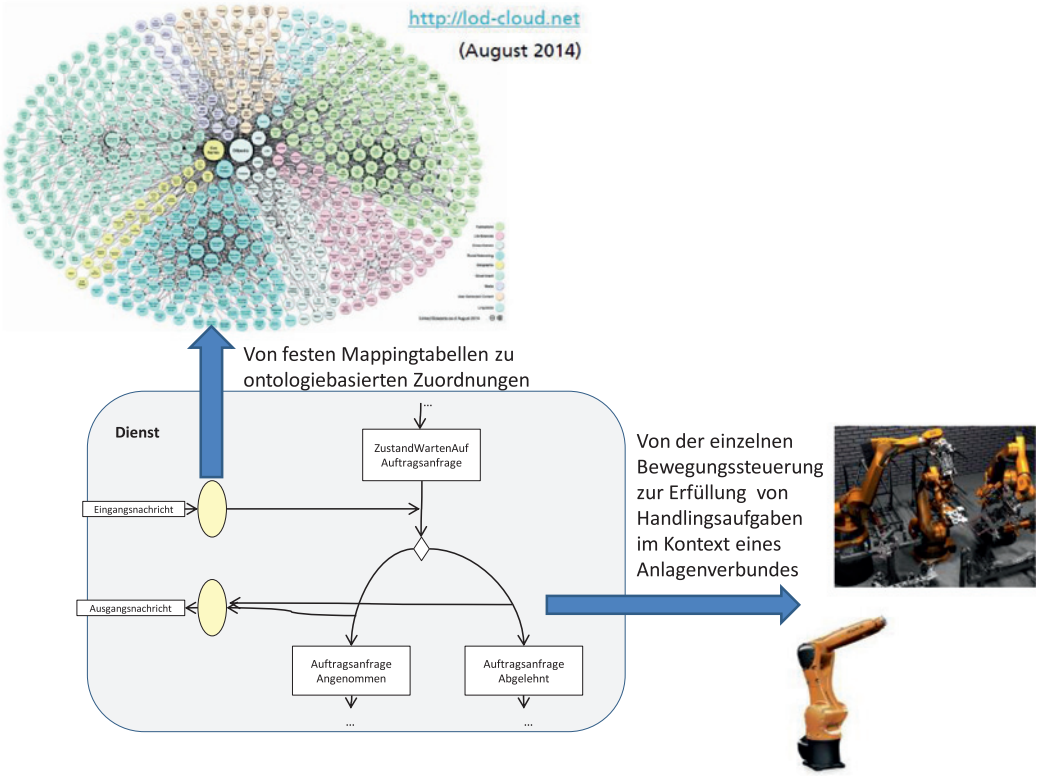

Abbildung 13: Erhöhung der Aufgabenmächtigkeit durch erweitertes Weltmodell der I4.0-Komponente.

5 Muster von Interaktionen zwischen I4.0-Komponenten

Anwendungsszenarien für I4.0-Systeme sind stets durch eine hohe Wandelbarkeit und Anpassbarkeit während des operativen Betriebs für aktuelle Aufgaben der Wertschöpfungsketten gekennzeichnet. Begleitend wird es Interaktionsmuster geben, die für die zu bearbeitenden Aufgabe abzuwickeln sind (z. B. Auskunft, Vereinbarungen, Sicherheit und Konfiguration).

Eine der ersten solcher Aufgaben ist die Feststellung der gegenseitigen Identität und die Vereinbarung der Security-Maßnahmen. Die entsprechenden Festlegungen werden zurzeit in der AG3 der Plattform entwickelt und werden zu einem späteren Zeitpunkt in die Interaktionsmuster aufgenommen.

I4.0-Komponenten sollen, anders als klassische AT-Geräte oder -Komponenten, selbst entscheiden können, ob sie eine bestimmte Aufgabe erfüllen können oder nicht. Es muss im Vorfeld also festgestellt werden, ob eine I4.0-Komponente eine Aufgabe funktional, mit den erforderlichen nicht-funktionalen Eigenschaften (z. B. Qualität) und zum geforderten Zeitpunkt ausführen kann. Man kann sich auch vorstellen, nicht-funktionale Eigenschaften, beispielsweise aus dem Business-Layer von RAMI 4.0, wie z. B. den Preis einzubeziehen. Dies erfordert Interaktionsmuster, die zum Beispiel folgende Aufgaben erfüllen können:

- Feststellung und Verifizierung der Identität und Vereinbarung der Security-Maßnahmen – vor Beginn der gegenseitigen Aktivitäten müssen die Sicherheitsaspekte geklärt sein
- Initiierung einer Aufgabe in einer Wertschöpfungskette – Bezug auf eine vorhandene Vereinbarung
- Anbahnung einer Aufgabe in einer Wertschöpfungskette – Anfrage für eine Zusammenarbeit (zwischen 1..n I4.0-Komponenten, orchestriert durch die initiierende I4.0-Komponente)
- Aushandlung einer Aufgabe in einer Wertschöpfungskette – Verhandlung der Details (funktionale und nicht-funktionale Eigenschaften) für eine Zusammenarbeit, einschließlich der Klärung von Zugriffsrechten der anfragenden I4.0-Komponente (Authentifizierung und Autorisierung)
- Beauftragung einer Aufgabe an eine I4.0-Komponente – Freigabe einer Zusammenarbeit (im einfachsten Fall startet die Aufgabe sofort, die Aufgabe kann aber in einen Batchpuffer (Auftragsliste) eingeordnet werden)

– Durchführung einer Aufgabe – die Zusammenarbeit wird zwischen dem Auftraggeber und den entsprechenden Teilmodellen abgewickelt. Laufzeitinteraktionen orchestrieren und steuern (mehrere) I4.0-Komponenten zur Durchführung der Aufgabe.
– Beenden einer Aufgabe in einer Wertschöpfungskette – Beenden des Auftragsverhältnisses nach Erfüllung
– Melden von Störungen bei der Verhandlung und Auftragsabarbeitung – Während der gesamten Anbahnung und Abwicklung einer Aufgabe können nicht erwünschte oder unvorhergesehene Ereignisse eintreten, die behandelt werden müssen

Jede I4.0-Komponente beherbergt einen bestimmten Satz an diesen Interaktionsmustern, die die Aufgaben abdecken, die die I4.0-Komponente anbietet. Es ist davon auszugehen, dass alle I4.0-Komponenten Sicherheitsaspekte abprüfen bzw. einstellen, bevor mit der Zusammenarbeit begonnen werden kann. Einfache I4.0-Komponenten werden auf vorgefertigte Vereinbarungen zurückgreifen, die eindeutig identifizierbar und damit abprüfbar sein müssen und zeitlich nacheinander die angebotenen Teilmodelle aktivieren können. Über verschiedenste Kombinationen und Ausbaustufen kann man sich auch I4.0-Komponenten vorstellen (z. B. ein Bearbeitungszentrum in der Fertigungstechnik oder eine modulare Station in der Verfahrenstechnik), die den Inhalt und die Abwicklung eines Auftrages verhandeln. Diese I4.0-Komponenten könnten z. B. Batch-Funktionalitäten, die heute typischerweise in MES-Systemen beherbergt sind, als Teilmodelle anbieten.

6 Zusammenfassung und Ausblick

Der Beitrag gibt einen Überblick über das Interaktionsmodell für I4.0-Komponenten. Es wird angenommen, dass es Interaktionsmuster geben wird, die auf dem Austausch von Nachrichten basiert, in denen Merkmale die Informationsträger sind. Merkmale sind die wesentlichen Bestandteile des I4.0-Komponentenmanifests. Die Nachrichten beeinflussen die Zustände der I4.0-Komponenten, die in der Verwaltungsschale in Form von protokollorientierten und/oder RPC-orientierten Zustandsmaschinen beschrieben werden. Die Nachrichteninhalte und die Zustandsübergänge beschreiben zusammen die Semantik einer Interaktion. Der Beitrag verfeinert die Definition der I4.0-Komponente wie diese in der DIN SPEC 91345 [3] definiert wurden und bildet beispielhaft das Interaktionsmodell auf OPC UA ab. Der Beitrag stellt aktuelle Diskussionen dar, wie sie zurzeit in der AG1 UAG Ontologie/Grammatik geführt werden.

Literatur

1. M. Hermann, T. Pentek, and B. Otto: „Design principles for industrie 4.0 scenarios: A literature review“, Working Paper, Audi Stiftungslehrstuhl Supply Net Order Management, Technische Universität Dortmund, 2015.
2. J. Jasperneite: „Was hinter begriffen wie Industrie 4.0 steckt“, Computer & Automation, 2012.
3. DIN SPEC 91345 „Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)“, April 2016. DIN ICS 03.100.01; 25.040.01; 35.240.50.
4. F. Bangemann, J. Reich, and Ch. Diedrich: A Grammar for the Semantics of Component Interaction of Cyber-Physical Systems. IEEE Interantional Symposium on Industrial Electronics, June 8–10 2016. SF-006319 proceedings.
5. D. Harel: „Statecharts: A Visual Formalism for Complex Systems“. In: Science of Computer Programming. 8/1987, North Holland, S. 231–274.
6. eCl@ss: www.eclass.org
7. S. Gruner, J. Pfrommer, and F. Palm: „RESTful Industrial Communication with OPC UA,“ In: IEEE Transactions on Industrial Informatics, 2016, no. 99, pp. 1–1.
8. G. Candido, F. Jammes, J. B. de Oliveira, and A. W. Colombo: „SOA at device level in the industrial domain: Assessment of OPC UA and DPWS specifications,“ In: Industrial Informatics (INDIN), 8th IEEE International Conference on. IEEE, 2010, pp. 598–603.
9. DPWS v1.1, OASIS Devices Profile for Web Services (DPWS) Version 1.1, OASIS Standard, 2009, Online: http://docs.oasis-open.org/ws-dd/dpws/1.1/os/wsdd-dpws-1.1-spec-os.pdf
10. J. Reich: Composition, Cooperation, and Coordination of Computational Systems. arXiv:1602.07065v1 [cs.SE].
11. J. Reich: Eine semantische Klassifikation von Systeminteraktionen, In: D. Cunningham, P. Hofstedt, K. Meer, and I. Schmitt (Hrsg.): INFORMATIK 2015 Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2015.
12. J. Lunze: Ereignisdiskrete Systeme. Auflage: 2. De Gruyter Oldenbourg; (10. Oktober 2012), ISBN-10: 3486718851, ISBN-13: 978-3486718850.
13. GMA 7.21: Industrie 4.0 Service Architecture – Basic Concepts for Interoperability. VDI/VDE 2016.

Autoreninformationen

Christian Diedrich
Otto-von-Guericke-Universität Magdeburg, Magdeburg
Christian.Diedrich@ovgu.de

Alexander Bieliaiev
Otto-von-Guericke-Universität Magdeburg, Magdeburg
oleksandr.bieliaiev@ovgu.de

Jürgen Bock
KUKA Roboter GmbH, Augsburg
juergen.bock@kuka.com

Andreas Gössling
Hilscher Gesellschaft für Systemautomation mbH, Hattersheim
agoessling@hilscher.com

Rolf Hänisch
Fraunhofer-Institut für Offene Kommunikationssysteme FOKUS, Berlin
Rolf.Haenisch@fokus.fraunhofer.de

Andreas Kraft
DEUTSCHE TELEKOM AGT-Labs (Research & Innovation), Berlin
A.Kraft@telekom.de

Florian Pethig
Fraunhofer-Anwendungszentrum Industrial Automation (IOSB-INA), Lemgo
florian.pethig@iosb-ina.fraunhofer.de

Oliver Niggemann
Hochschule Ostwestfalen Lippe, Lemgo
oliver.niggemann@hs-owl.de

Johannes Reich
SAP, Walldorf
johannes.reich@sap.com

Friedrich Vollmar
Consultant, Frankfurt/Main
friedrich.vollmar@web.de

Jörg Wende
IBM Deutschland GmbH, Dresden
joerg.wende@de.ibm.com