Plug&Produce durch Software-defined Networking

Thomas Kobzan[1], Alexander Boschmann[2], Simon Althoff[2], Immanuel Blöcher[3], Sebastian Schriegel[1], Jan Stefan Michels[2], Jürgen Jasperneite[1,4]

[1]Fraunhofer IOSB-INA, 32657 Lemgo
[2]Weidmüller Interface GmbH & Co. KG, 32758 Detmold
[3]Hilscher Gesellschaft für Systemautomation mbH, 65795 Hattersheim
[4]Institut für Industrielle Informationstechnik (inIT), 32657 Lemgo

[1]{thomas.kobzan, sebastian.schriegel, juergen.jasperneite}@iosb-ina.fraunhofer.de
[2]{alexander.boschmann, simon.althoff, janstefan.michels}@weidmueller.com
[3]IBloecher@hilscher.com
[4]juergen.jasperneite@hs-owl.de

Abstract: Dieser Beitrag behandelt das Potential von softwaredefinierten Netzwerken für das Industrie 4.0 Anwendungsszenario Plug&Produce. Um dies umzusetzen, müssen Autokonfigurationsmethoden für das industrielle Netzwerk entwickelt und untersucht werden. Software-definierte Netzwerke eignen sich aufgrund ihrer zentralen Administration dafür, jedoch müssen Mechanismen umgesetzt werden, welche die Autokonfiguration vornehmen. Es wird dargestellt, dass die Netzwerkteilnehmer ihre Selbstbeschreibung an die zentrale Administration kommunizieren müssen, damit diese das Netzwerk verwalten und konfigurieren kann. Anhand eines industriellen Use-case, in dem eine Kamera zwischen Produktionslinien ausgetauscht wird und ohne Konfigurationsaufwand reintegriert werden soll, wird eine erste Implementierung der Mechanismen dargestellt. Ergebnis hieraus ist, dass sich die Kombination aus softwaredefinierter Vernetzung und OPC UA sich dafür eignet. Hierbei werden auf den Netzwerkteilnehmern OPC UA-Server installiert. Die Netzwerkadministration bzw. der erstellte Integrierungsmechanismus verfügt über eine OPC UA-Client Implementierung und kann so die Eigenschaften und Anforderungen der Netzwerkteilnehmer auslesen. Anschließend kann er die Informationen verarbeiten und das Netzwerk durch die Installation von SDN-Flows auf den Netzwerkinfrastrukturkomponenten konfigurieren.

1 Einleitung

Industrie 4.0 stellt neue Anforderungen an die Administration industrieller Netzwerke. Eine hohe Flexibilität und robuste Methoden für die Gewährleistung der Informationssicherheit werden in Zukunft vorausgesetzt. Dabei soll dies integraler Bestandteil der Architektur werden, was bedeutet, dass die Konfiguration automatisch geschehen soll [BE17]. Der Anwender soll lediglich die verwaltungstechnischen Rahmenbedingungen setzen müssen. Zu diesem Zwecke werden Möglichkeiten gesucht, die das Paradigma der softwaredefinierten Vernetzung (engl. Software-defined networking, kurz: SDN) hierfür nutzbar zu machen. SDN zeichnet sich dadurch aus, dass der Konfigurationsaspekt in logisch zentralisierten Steuerungskomponenten erfolgt und die Netzwerkinfrastruktur lediglich die Weiterleitung der Daten übernimmt [KA14]. Da alle Informationen des Netzwerks an einer Stelle zusammenlaufen, ergeben sich Möglichkeiten das Netzwerk effektiv und flexibel zu programmieren, was ein großes Potential für künftige industrielle Netzwerke birgt [HE16].

Anhand eines Use-case aus dem Bereich Spritzguss wird der Mehrwert dieser SDN-Netzwerkarchitektur exemplarisch gezeigt. In Abbildung 1 sind zwei Fertigungslinien abgebildet, die jeweils aus einer Spritzgießmaschine mit Steuerung (SPS), einem Förderband und einer Montagestation bestehen. Es wird hierbei nur die SPS als vernetzte Komponente gezeigt. Das Kunststoffgranulat wird erhitzt und unter hohem Druck in die Form gepresst. Nach Erkalten wird das Formteil ausgestoßen und auf einem Förderband zu einem Montageautomaten transportiert. Anschließend findet die lagegerechte Montage mit entsprechenden Metallteilen statt. Zur Sicherstellung der elektrischen Leistungsfähigkeit erfolgt eine Hochspannungsprüfung der Klemmenträger. Bei sicherheitsrelevanten Klemmen findet zusätzlich eine Überprüfung mit

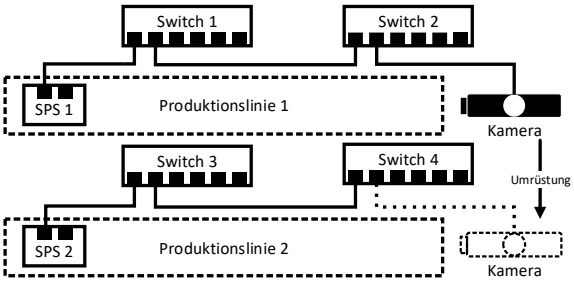

Abbildung 1: Industrieller Use-case

einer hochauflösenden Kamera statt. Das erlaubt eine zusätzliche ganzheitliche optische Analyse des Bauteils. Die Kamera kann direkt mit der Steuerung kommunizieren und notfalls den Prozess anhalten. Um einen flexiblen und dynamischen Fertigungsablauf zu gewährleisten, muss der Umrüstvorgang der Kamera möglichst schnell erfolgen. Im vorgestellten Beispiel soll die Kamera, wie in Abbildung 1 gezeigt, die Produktionslinie wechseln. Ein SDN-Netzwerkmanagement könnte anhand der explorierten Netzwerkstruktur erkennen, welchem Fertigungsprozess die Kamera zugeordnet ist und mit welcher SPS sie kommunizieren muss. Ein SDN-Netzwerkmanagement ermöglicht die benötigte Flexibilität im Sinne von Plug&Produce, doch müssen stets auch weitere Aspekte wie Datensicherheit und Sicherstellung notwendiger Quality-of-Service-Anforderungen mit betrachtet werden. Das bedeutet, dass das Netzwerkmanagement einen Mechanismus benötigt, der sowohl entscheidet, wie das Netzwerk auf eine derartige Zustandsänderung reagieren soll, als auch die entsprechenden Managementbefehle erteilt.

Der Beitrag ist im Weiteren wie folgt strukturiert. Abschnitt 2 gewährt einen Überblick über den Stand der Technik. Das Konzept wird in Abschnitt 3 dargestellt und in Abschnitt 4 in eine erste Implementierung überführt. Abschnitt 5 fasst den Beitrag zusammen und stellt in einem Ausblick sowohl die nächsten Schritte der Implementierung als auch Möglichkeiten für Forschungsaktivitäten dar.

2 Stand der Technik

Nachfolgend werden die aktuellen Bestrebungen im Bereich der Autokonfiguration mit Fokus auf SDN dargestellt. Des Weiteren wird das Konzept SDN erläutert.

2.1 Autokonfiguration

Das Plug&Produce Anwendungsszenario spielt künftig vor allem in der dynamischen Rekonfiguration von Produktionskomponenten eine Rolle [HY18]. Hierfür werden Komponenten benötigt, die über automatische Konfigurationsmechanismen verfügen. Eine wichtige Grundlage sind die von Diedrich et. al [DI17, DI18] beschriebenen Mechanismen. Es werden hier Prinzipien beschrieben, wie durch eine semantische Beschreibung mittels z. B. OPC UA aus Assets Industrie 4.0 Komponenten entstehen, die anschließend miteinander interagieren können. OPC UA-basierte Plug&Produce Methoden werden in [PA18] sowie [PR17] vorgestellt. Auch wenn diese Konzepte und Mechanismen grundsätzlich breit anwendbar sind, fokussieren sie sich in den Beschreibungen auf die Interaktion von Produktionskomponenten horizontal untereinander, vertikal mit übergeordneten MES und ERP IT-Systemen sowie als I4.0 Verbundkomponente. Eine sehr komplexe Komponente in einer vernetzten Fabrik, die für eine automatische Konfiguration mit betrachtet werden muss, ist jedoch das Netzwerk bzw. die Kommunikation selber. Rauchhaupt et al. beschreiben in [RA16] die SDN-basierte industrielle Netzwerkkommunikation als RAMI4.0 konforme Industrie 4.0-Komponente mit einer Verwaltungsschicht. Diese Konzepte schaffen die Grundlage, dass die I4.0 Komponenten der Produktion und der übergeordneten IT-Systeme mit der I4.0 Komponente(n) „Netzwerk“ interagieren können.

Die grundlegenden Interoperabilitätsmechanismen und Implementierungen können so erweitert werden, dass die Komponenten Informationen an ein Netzwerkmanagement übertragen können. Dadurch wird das Netzwerk befähigt, auf physische und logische Veränderungen der Produktionsumgebung zu reagieren. Henneke et al. stellen in [HE18] den praktisch ersten Schritt zum Plug&Produce in Verbindung mit SDN vor, indem sie OPC UA Discovery Mechanismen in eine SDN Plattform integrieren.

Die konsequente Erweiterung dieses Verfahrens ist die Extraktion von Informationen, die für den Datenverkehr wichtig sind und die automatisierte Konfiguration des SDN durch erstellte Routinen. Um ein besseres Verständnis wie das durch SDN möglich werden kann zu erhalten, werden nachfolgend grundlegende Eigenschaften und Mechanismen eines SDN beschrieben.

2.2 Software-defined Networking

Ein softwaredefiniertes Netzwerk kann in drei Ebenen unterteilt werden, was Abbildung 2 veranschaulicht. In den Ebene sind verschiedene Funktionen für das Netzwerk abstrahiert. In der Datenebene (engl. Data-Plane) findet der Nutzdatenverkehr statt und auf ihr befinden sich die Komponenten, die für die Weiterleitung der Daten verantwortlich sind. Anders als bei herkömmlichen Netzwerkinfrastrukturkomponenten besitzen diese in einem SDN keine Steuerungsfunktionen. Diese befinden sich in dedizierten Steuerungen, die sich auf der Steuerungsebene (engl. Control-Plane) befinden [KR15, KA16].

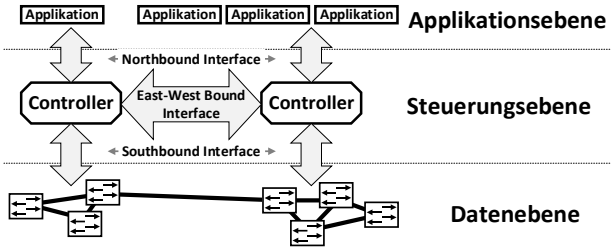

Abbildung 2: Grundsätzliches Konzept eines softwaredefinierten Netzwerks

Die oberste Ebene, genannt Applikationsebene, beinhaltet Applikationen, die zur Administration aber auch Unterstützung des Netzwerkbetriebs eingesetzt werden können. Hierbei ist es implementierungsabhängig, in welcher Art und Weise die Applikationsebene auf die darunterliegende Steuerungsebene einwirkt. SDN-Lösungen wie die Plattformen ODL (OpenDaylight) und ONOS lassen es zu, dass über das sogenannte Northbound-Interface direkt auf die Konfiguration der Netzwerkinfrastruktur-Komponenten eingewirkt werden kann. Dadurch lässt sich auf das Netzwerk sehr feingranular administrieren. Die Applikationsebene bietet somit die Möglichkeit basierend auf Autokonfigurationsmechanismen Plug&Produce umzusetzen. Jedoch muss darauf geachtet werden, keine zusätzlichen Sicherheitslücken oder andere Nachteile für den laufenden Netzwerkbetrieb zu generieren. Für das Northbound-Interface kommen z. B. Interfaces zum Einsatz, die REST (Representational State Transfer) verwenden.

Auf der darunterliegenden Steuerungsebene befinden sich Mechanismen, die das Netzwerk verwalten und hierbei eine Basisfunktionalität sicherstellen. Da in einem softwaredefinierten Netzwerk mehrere Netzwerksteuerungen bzw. -controller eingesetzt werden können, ist eine Schnittstelle für die Interoperabilität von mehreren Steuerungen als East-West-Bound Interface definiert [SC18]. Aus dieser Ebene werden die Weiterleitungsanweisungen in die Datenebene versendet.

Auf der Datenebene befinden sich die Komponenten, welche die Daten nach Anweisungen der Steuerungsebene weiterleiten sowie der Nutzdatenverkehr. Die Verbindung von Steuerungs- zu Datenebene stellen sogenannte Southbound-Interfaces sowie -Protokolle sicher. Es existieren zwar mehrere Southbound-Protokolle, der de facto Standard ist jedoch das OpenFlow-Protokoll [CO17]. Durch seine Verbreitung ist es wichtig seine grundsätzliche Mechanik nachfolgend zu erläutern.

2.3 OpenFlow

OpenFlow-Protokoll mit der aktuellen Version 1.5.1 [OF15] wird von der Open Networking Foundation verwaltet und spezifiziert. Es beschreibt wie ein Controller und ein Switch miteinander kommunizieren und in welcher Art und Weise die Switche Datagramme weiterleiten. Diese spezielle Mechanik ist in Abbildung 3 grundlegend aufgezeigt. Hierbei muss angemerkt werden, dass die Funktion eines SDN-Switches bzw. die des OpenFlow-Protokolls über die Schicht 2 hinausgeht. Dem OpenFlow-basierten Switch ist es u. a. auch möglich, Datagramme nach z. B. IPv4-/IPv6-Adressen oder TCP/UDP-Ports auszuwerten.

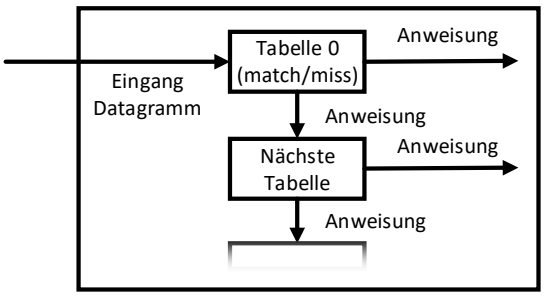

Abbildung 3: Grundlegende Funktionsweise der Flow-Anweisungen des OpenFlow-Protokolls

Das Datagramm kommt initial in der Verarbeitung des Switches an. Hier wird es mit Einträgen in einer sogenannten Flow-Tabelle verglichen. Entsprechen Informationen des Datagramms einem Eintrag, ist eine Übereinstimmung („Match“) gefunden. Fälle von mehreren Übereinstimmungen handhabt eine Priorisierung der Einträge. Findet sich keine Überreinstimmung, handelt es sich um ein „Miss“. Jedes Match und jedes Miss haben gewisse Anweisungen zur Folge. Diese Anweisungen können u. a. sein, die Datagramme zu verwerfen oder sie an spezifizierte Hardware-Ports weiterzuleiten. Des

Weiteren können sie an weitere Flow-Tabellen weitergeleitet werden, in denen sie wiederum auf Übereinstimmungen zu Tabelleneinträgen geprüft werden.

3 Theoretisches Konzept

In diesem Abschnitt wird zunächst das theoretische Konzept des Integrierungsmechanismus näher erläutert. Anschließend werden die Rollen der Netzwerkteilnehmer und auszutauschende Informationen sowie die Anforderungen an eine zu verwendende SDN-Lösung beschrieben.

3.1 Gesamtarchitektur

Die Konzept der Gesamtarchitektur ist in Abbildung 4 dargestellt. Die dreischichtige SDN-Architektur ist gegeben. Für die beschriebene Plug&Produce-Fähigkeit muss eine Applikation erstellt werden, welche die Einpassung vornimmt. Diesem Integrierungsmechanismus müssen die dafür notwendigen Informationen zur Verfügung gestellt werden. Dies sind der Netzwerkstatus, wie z. B. Informationen über die Topologie sowie die bereits eingestellten Weiterleitungsbefehle. Zusätzlich benötigt es die Charakteristika der Netzwerkteilnehmer, die in Eigenschaften (z. B. MAC-Adresse, IP-Adresse) sowie Anforderungen an das Netzwerk getrennt sind. Anforderungen an das Netzwerk müssen semantisch so beschrieben sein, dass das Netzwerkmanagement sie verstehen und in entsprechende Befehle transformieren kann. Der Integrierungsmechanismus überführt die Informationen in SDN-Einstellungsparameter, mit denen er anschließend das SDN konfiguriert.

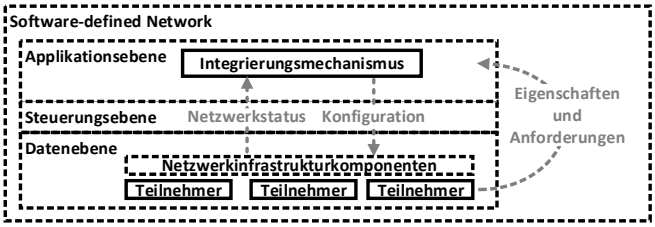

Abbildung 4: Konzept des Integrierungsmechanismus

Nachfolgend werden in Abschnitt 3.2 die Rollen der Netzwerkteilnehmer sowie die auszutauschenden Informationen beschrieben. In Abschnitt 3.3 wird auf die Anforderungen an die benötigten SDN-Komponenten genauer eingegangen.

3.2 Rollen der Netzwerkteilnehmer und auszutauschende Informationen

Rauchhaupt et al. beschreiben in [RA16], dass ein Netzwerkteilnehmer Informationen über seine Fähigkeiten im Kontext der Kommunikation (z. B. Quality-of-Service-Charakteristiken) beschreibt. Diese Fähigkeiten gilt es um weitere semantische Beschreibungen ihrer Eigenschaften und v. a. um ihre Anforderungen an das Netzwerk zu erweitern. Unter die Eigenschaften fallen allgemeine Informationen zur Identifikation des Geräts (z. B. eindeutige Identifikation, Seriennummer) und netzwerkrelevante Eigenschaften wie Netzwerkschnittstellen und -adressen [DI17, RA40].

Im Kontext von SDN beinhaltet ein Netzwerkteilnehmer die semantischen Beschreibungen von Anforderungen, die er an das Netzwerk hat. Anforderungen können beispielsweise Verbindungsanfragen sein, mit der ein Teilnehmer mitteilt, mit welchen Teilnehmern und/oder in welcher Art und Weise (Verwendung von Portfilter, Black- oder Whitelisting von Teilnehmern und Adressen) er kommunizieren möchte.

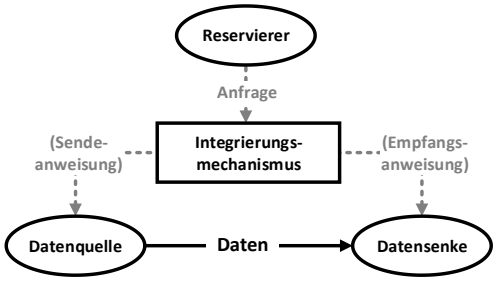

Abbildung 5: Rollen der Teilnehmer im Netzwerk und Interaktion mit dem Integrierungsmechanismus

Es werden drei Rollen für Akteure in einem Netzwerk definiert, was Abbildung 5 verdeutlicht. Die Fähigkeit Anforderungen an das Netzwerk zu stellen gibt einem Teilnehmer die Rolle eines Reservierers. Zudem existiert die Rolle der Datenquelle sowie die der Datensenke. Teilnehmer können auch mehrere Rollen gleichzeitig einnehmen.

Die semantischen Beschreibungen der Eigenschaften der Netzwerkteilnehmer und deren Anforderungen werden an den Integrierungsmechanismus übermittelt. Mittels bestimmter Eigenschaften (z. B. MAC-Adresse) wird ein Netzwerkteilnehmer eindeutig identifiziert. Durch die in der Anforderung enthaltenen Verbindungsanfragen kann das SDN die Geräte im Netzwerk durch SDN-Flows zusammenschalten. Anschließend sind die Netzwerkteilnehmer befähigt, entsprechend ihrer Verbindungsanfragen Daten auszutauschen. Die in der Abbildung gezeigten Sende- und Empfangsanweisungen sind optional. Eine Datenquelle kann so konfiguriert sein, dass sie ohne Anweisung permanent ihre Daten kommuniziert und eine Datensenke kann für einen stetigen Datenempfang konfiguriert sein.

Zwei Beispiele sollen das Beschriebene verdeutlichen:

1. Ein industrielles Managementsystem stellt zum Datenmonitoring eine Verbindung zwischen einem Sensor und einem Analysetool in einer Cloud her. Das Managementsystem agiert als Reservierer, der Sensor als Datenquelle und das Analysetool als Datensenke. Dies ist an sich eine administrative Handlung, aber passt dennoch in die hier gezeigte Rollenverteilung, da das Managementsystem als Reservierer mit einer möglicherweise sehr hohen Priorität seine Anfrage kommuniziert.

2. Im Falle des in Abschnitt 1 beschriebenen industriellen Use-case ist die Kamera die Datenquelle. Eine SPS soll die Bilddaten der Kamera erhalten und tritt als Datensenke auf. Die Kamera ist so konfiguriert, dass sie die Anfrage stellt, ihre Daten an die SPS zu leiten, sobald sie dem Netzwerk hinzugefügt wird. Somit tritt die Kamera sowohl in der Rolle der Datenquelle als auch in der des Reservierers auf.

3.3 Anforderungen an die SDN-Komponenten

Da es sich bei SDN um ein Paradigma bzw. Architekturkonzept handelt, existiert keine Standard-SDN Lösung. Die grundsätzliche Funktionalität wurde bereits in Verbindung mit Abbildung 4 beschrieben.

Nachfolgend werden die benötigten SDN-Komponenten und die Anforderungen an sie näher erläutert. Die Implementierung des vorgeschlagenen Lösungskonzepts soll vorhandene Lösungen verwenden.

- **Datenebene:**
  Netzwerkinfrastrukturkomponenten, die für die Weiterleitung des Nutzdatenverkehrs verwendet werden, sollen das OpenFlow-Protokoll verarbeiten können.
- **Steuerungsebene:**
  Da OpenFlow verwendet wird, muss der Controller ebenfalls dieses Protokoll verarbeiten können. Controllerimplementierungen mit East-Westbound-Funktionalität sind anzustreben, da in späteren Implementierungen auch größere und komplexere Netzwerke konfiguriert werden sollen.
- **Applikationsebene:**
  Dem Integrierungsservice muss es möglich sein, über ein Interface sowohl die Netzwerkstatusinformationen auszulesen als auch die Netzwerkparameter anzupassen. Der Integrierungsservice muss eigene Weiterleitungskonfigurationen für die Datenebene erstellen können.
- **Netzwerkstatusinformationen:**
  Zwingend erforderlich sind grundlegende Informationen über die Topologie. Es muss ersichtlich sein, an welchen physikalisch Ports der Switches welche Komponenten angeschlossen sind. Des Weiteren müssen die bereits konfigurierten Weiterleitungsbefehle auslesbar und editierbar (auch entfernbar) sein.

4 Umsetzung

Die beschriebene Lösung wird angelehnt an den industriellen Use-case implementiert. Dieser Abschnitt beschreibt den aktuellen Stand der Implementierung und zeigt Erfahrungen sowie offene Punkte auf. Abbildung 6 beschreibt die Hard- und Software-Komponenten und die Verbindung untereinander. Als SDN Plattform mit Controller wird OpenDaylight (ODL) Version 4.0.10 verwendet. ODL verfügt über mehrere Southbound-Protokolle und -Controllerimplementierungen, u. a. OpenFlow. Der Integrierungsmechanismus wird in Python Version 3.6.4 umgesetzt, da Python alle benötigten APIs unterstützt. Beides ist installiert auf einem Desktop PC mit Intel® Core™ i7-3770 CPU @ 3.40 GHz und 8 GB RAM. Betriebssystem ist Windows 10 64-Bit. Die SDN-Plattform ist southbound via einem 5-Port unmanaged Switch an vier SDN-Switches angeschlossen. Diese sind vom Modell Zodiac FX von der Firma Northbound Networks und besitzen einen Ethernet Port für den Controller und drei Ports für den Netzwerkdatenverkehr. Für eine erste Proof-of-Concept-Implementierung werden stellvertretend für die SPS zwei Hilscher netPIs verwendet. Als Kamera kommt ein

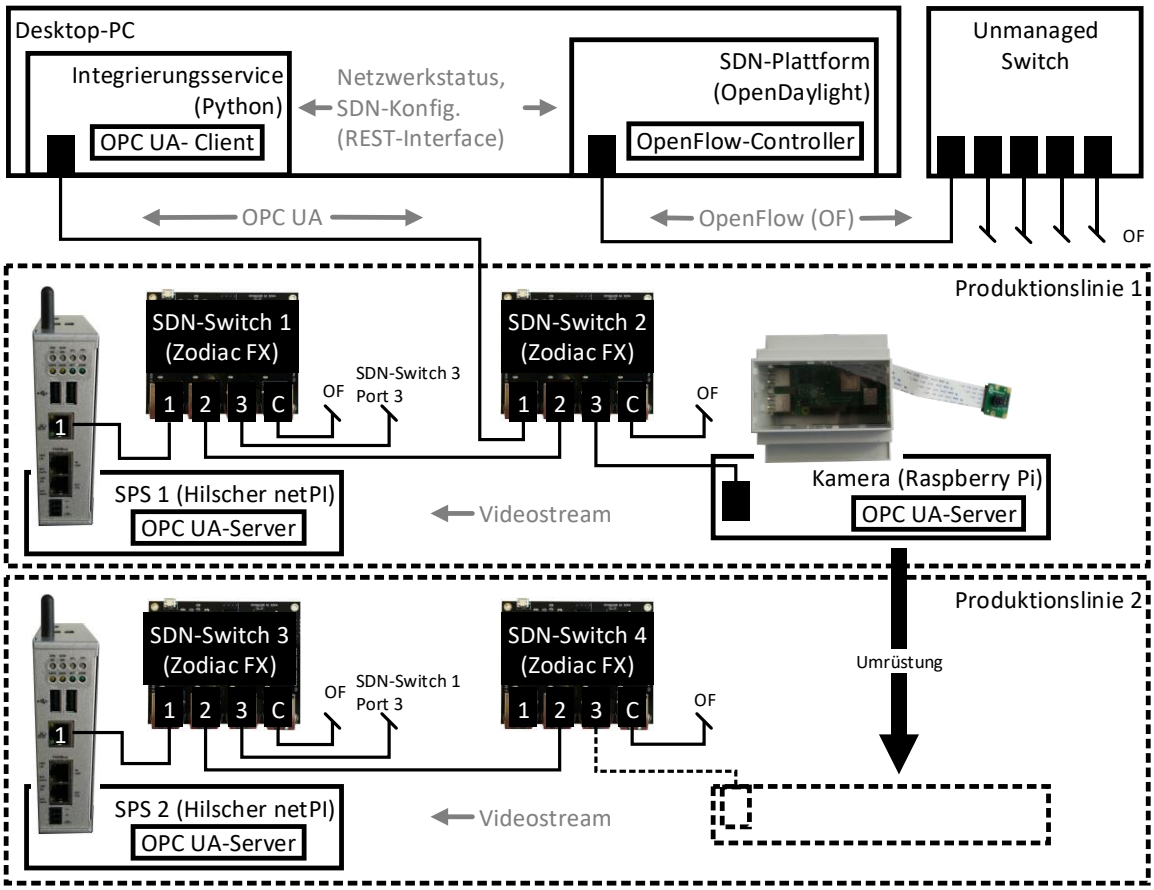

Abbildung 6: Architektur der Implementierung

Raspberry Pi 3 (RPi) mit einem angeschlossenem RPi Camera Module zum Einsatz. Die miteinander verbundenen Switches 1 und 2 repräsentieren Produktionslinie 1, gekennzeichnet durch den Anschluss der SPS 1. Die miteinander verbundenen Switches 3 und 4 repräsentieren Produktionslinie 2, gekennzeichnet durch den Anschluss der SPS 2. Der Integrierungsmechanismus ist über einen extra Ethernet Port mit dem Datenverkehr der Netzwerkteilnehmer verbunden. Dieser ist über Switch 2 mit Produktionslinie 1 verbunden und über eine Verbindung von Switch 1 und 3 mit Produktionslinie 2.

Dadurch ist es dem Mechanismus möglich, über OPC UA die Charakteristika der Netzwerkteilnehmer auszulesen (siehe Abschnitt 4.1). Gleichzeitig kann er über das von ODL zur Verfügung gestellte Northbound Interface (REST basiert) auf die SDN-Plattform einwirken. Somit kann er den Netzwerkstatus, z. B. die von ODL bereit gestellten Topologieinformationen auslesen. Des Weiteren ist es ihm möglich, neue Flows auf den Netzwerkinfrastrukturkomponenten zu installieren.

Die Kamera bzw. der RPi stellt einen Videostream zur Verfügung, der über eine Browserumgebung per TCP-Port 8081 (HTTP) ausgegeben werden kann.

4.1 Informationsmodell

Die Netzwerkteilnehmer müssen in ihrem Informationsmodell Eigenschaften und Anforderungen an das Netzwerk hinterlegen. Die Umsetzung des Informationsmodells erfolgt mit Hilfe von OPC UA. Der Vorteil hier liegt darin, dass OPC UA einen Server-Client-Kommunikationsmechanismus mit sich bringt, mit dem das Informationsmodell von den Servern durch einen Client ausgelesen werden kann. Die herkömmliche Server-Client-Option ist hier ausreichend. Da nur der Integrierungsmechanismus über einen Client verfügen muss, um die Informationen der Netzwerkteilnehmer auszulesen, ist die alternative Publisher-Subscriber Architektur nicht notwendig.

Das Informationsmodell, dargestellt in Abbildung 7, enthält die zwei Sektionen „ConnectionSet“ und „DeviceInformationSet“. Das ConnectionSet enthält die Anforderungen an das Netzwerk sowie den Teil der Netzwerkteilnehmereigenschaften, der die Netzwerk-Interfaces beschreibt. Die Verbindungsanfragen an den Integrierungsservice werden mittels einer „ConnectionRequest“-Liste, die einzelne Verbindungsanfragen enthält, im Informationshaushalt des Servers angegeben. Eine Anfrage wiederum besteht aus zwei Adressen, die mittels SDN miteinander verbunden werden sollen. Im Anwendungsfall der Kamera besteht die Anfrage aus der eigenen MAC-Adresse als Quelladresse und als Zieladresse die MAC-Adresse der jeweiligen SPS. Mittels der „NetworkInterfaceList“ teilt der Netzwerkteilnehmer mit, welche Interfaces er besitzt. Pro Interface teilt er den Namen und MAC-Adresse mit, falls vorhanden, auch die IP-Adressen (IPv4 & IPv6).

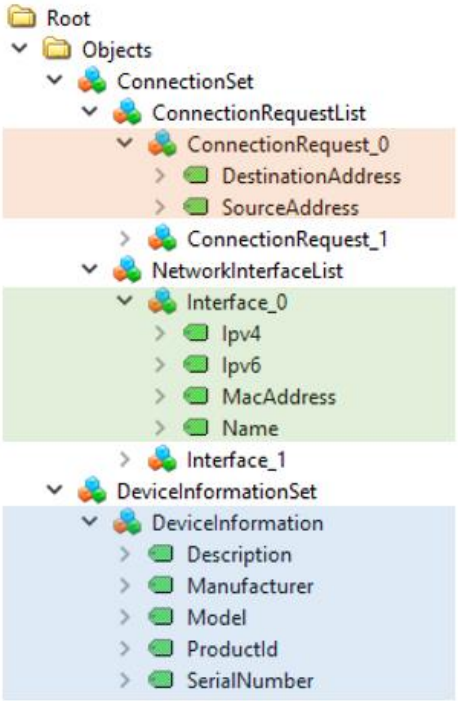

Abbildung 7: Modell des Informationshaushalts der Netzwerkteilnehmer

Das Informationsmodell wird mittels eines graphischen Editors erstellt. Aktuell bieten sich zwei nicht kommerziell vertriebene Editoren an: FreeOpcUa/opcua-modeler und der Siemens OPC UA Modeling Editor (SiOME). Der SiOME weißt dabei eine bessere Usability und weitreichendere Funktionalitäten auf, jedoch lässt sich in der aktuellen Version die verwendeten Informationsmodelle nur als gemeinsames XML exportieren. Die erstellten Informationsmodelle von beiden Editoren lassen sich trotz allem leicht in einen OPC UA-Server umsetzen. Hier wird der NodesetCompiler der Open-Source OPC UA Implementierung open65241 verwendet, um für jeden Netzwerkteilnehmer einen OPC UA-Server zu erstellen.

Durch die ergänzenden „DeviceInformationen“, die eine Beschreibung des Gerätes, den Hersteller, das Gerätemodell sowie Produkt- und Seriennummer enthalten, ist das Gerät eindeutig identifizierbar. Über die Abfrage der OPC UA-Informationsmodelle kennt das Netzwerkmanagement die Informationen, um ein Gerät eindeutig identifizieren zu können, die Netzwerk Interfaces und die teilnehmerspezifischen Verbindungsanfragen. Mit diesen Informationen und zusätzlichen Topologieinformationen können die Geräte, abhängig ihrer Anforderungen, miteinander vernetzt werden. Abbildung 8 zeigt die übertragenen Charakteristika beispielhaft. Die Kamera möchte eine Verbindung zu einer anderen Komponente herstellen. Wie bereits in Abschnitt 3.2 beschrieben, fällt hier die Rolle der Datenquelle mit der des Reservierers zusammen, da die Quelladresse der Verbindungsanfrage gleich der eigenen MAC-Adresse der Kamera ist.

#	Node Id	Display Name	Value	
1	NS2|Numeric|50211	Manufacturer	Weidmueller	DeviceInformation
2	NS2|Numeric|50213	Model	PiCam	
3	NS2|Numeric|50209	ProductId	54235.47	
4	NS2|Numeric|50212	SerialNumber	10054	
5	NS2|Numeric|50210	Description	Camera	
6	NS2|Numeric|50222	Name	eth0	Interface_0
7	NS2|Numeric|50224	MacAddress	b8:27:eb:3b:17:f9	
8	NS2|Numeric|50221	Ipv4	10.13.5.46	
9	NS2|Numeric|50223	Ipv6	fe80::ba27:ebff:fe3b:17f9	
10	NS2|Numeric|50216	SourceAddress	b8:27:eb:3b:17:f9	ConnectionRequest_0
11	NS2|Numeric|50215	DestinationAddress	c2:d9:d8:53:b2:6f	

Abbildung 8: Informationshaushalt der RPi Kamera (beispielhafte Werte)

4.2 Integrierungsmechanismus

Der in Python 3.6.4 implementierte Integrierungsmechanismus ist dafür zuständig, das Netzwerk auf die Topologieveränderung, bedingt durch die Umrüstung des RPi mit der Kamera, zu detektieren und darauf zu reagieren. Er sorgt dafür, dass die entsprechenden SDN-Flows auf den SDN-Switches durch den Controller installiert werden. Grundsätzlich kann er in die Komponenten „Aktivitätserkennung“ und „Integration“ getrennt werden. Die Aktivitätserkennung sorgt dafür, dass in der Topologie vorhandene und mit OPC UA erreichbare Komponenten als „aktiv“ gekennzeichnet werden. Der gegenteilige Fall hat eine Kennzeichnung mit „inaktiv“ zur Folge. Die Komponente Integration überwacht diesen Aktivitätsstatus. Bei einer Statusänderung von „inaktiv“ auf „aktiv“ geht der Service davon aus, die Komponente wurde physisch dem Netzwerk hinzugefügt. Somit reagiert er entsprechend und liest mittels OPC UA das auf der Komponente hinterlegt Informationsmodell aus. Entsprechend der hier übermittelten Eigenschaften und Anforderungen der Komponente und dem Netzwerkstatus, installiert der Service die benötigten Flows auf den Switches. Bei einer Statusänderung von „aktiv“ auf „inaktiv“ geht der Service davon aus, die Komponente wurde physisch vom Netzwerk entfernt. Die Flows werden wieder deinstalliert.

4.3 Umrüstvorgang

4.3.1 Vorbedingungen

In einer ersten Implementierung wird der Fokus auf die Integrierung mit Hilfe von SDN gelegt. Die initiale Registrierung einer Netzwerkkomponente, die noch nie im Netzwerk war, ist sehr von der Implementierung und entsprechenden Security Mechanismen abhängig und muss im Gegensatz zu Umrüstvorgängen, gesondert betrachtet werden. Somit wird angenommen, dass dem Netzwerk eine Liste mit registrierten Netzwerkteilnehmern vorliegt. Als Identifikatoren dienen die MAC-Adresse der Komponente sowie die OPC UA-Adresse. Über die MAC-Adresse wird in der Topologie erkannt, ob die Komponente vorhanden ist. Mittels der OPC UA-Adresse kann das Informationsmodell ausgelesen werden.

Es soll eine komplette Kontrolle des Netzwerks durch die Administration vorliegen, somit ist es dem Controller nicht erlaubt, Flows eigenmächtig zu installieren.

Auf allen Switches sind insgesamt fünf persistente Flows, aufgelistet in Tabelle 1, installiert. ARP-Datagramme sind uneingeschränkt zwischen den Netzwerkteilnehmern durchlässig und werden zur Topologieaufzeichnung auch an den Controller weitergeleitet. Für eine noch bessere Topologiedetektion werden LLDP-Datagramme ebenfalls an den Controller weitergeleitet (aktives ODL Loop-remover Modul). Datenverkehr über den OPC UA-Port 4840 (TCP) ist nur in Verbindung (&&=Verundung) mit der Quell- oder Ziel-IP des Integrierungsservice möglich. Der physische Anschluss des Service wäre durch die untere Einstellung nicht auf Port 1 von Switch 2 beschränkt. Findet sich kein Match, wird das Datagramm verworfen.

Tabelle 1: Persistent installierte Flows auf den SDN-Switches

Match:	**Anweisung:**
ARP-Protokoll	Ausgabe an Ports 1, 2, 3, *Controller*
LLDP-Protokoll	Ausgabe an Port *Controller*
Ziel ist IP-Adresse des Integrierungsservice && TCP-Port 4840 der Quelle	Ausgabe an Ports 1, 2, 3
Quelle ist IP-Adresse des Integrierungsservice && TCP-Port 4840 des Ziels	Ausgabe an Ports 1, 2, 3
Kein Match	Datagramm verwerfen

Zusätzlich zu den persistenten Flows, sind auf Switch 1 und 2 temporäre Flows installiert, die die Datenübertragung zwischen SPS und Kamera zulassen. Tabelle 2 zeigt die Konfiguration. Er werden zur Weiterleitungsentscheidung lediglich die MAC-Adressen der Teilnehmer verwendet. Dadurch ist eine Kommunikation zwischen SPS 1 und Kamera gegeben.

Tabelle 2: Installierte Flows für die Datenübertragung zwischen Kamera und SPS (Produktionslinie 1 Konfiguration)

	Match:	**Anweisung:**
Switch 1	Ziel ist MAC-Adresse der SPS 1 && Quelle ist MAC-Adresse der Kamera	Ausgabe an Port 1
	Ziel ist MAC-Adresse der Kamera && Quelle ist MAC-Adresse der SPS 1	Ausgabe an Port 2

Switch 2	Ziel ist MAC-Adresse der SPS 1 && Quelle ist MAC-Adresse der Kamera	Ausgabe an Port 2
	Ziel ist MAC-Adresse der Kamera && Quelle ist MAC-Adresse der SPS 1	Ausgabe an Port 3

4.3.2 Umrüstung

Wird nun die Kamera physisch von Port 3 des Switches 2 getrennt, erkennt dies der Service anhand der Topologieänderung und verifiziert dies dadurch, dass der OPC UA-Server der Kamera nicht mehr erreichbar ist. Der Status wird auf inaktiv gesetzt. Dies hat zur Folge, dass alle Flows, die der Kamerakommunikation mir der SPS dienen, gelöscht werden. Erfolgt nun die Umrüstung und die Kamera wird an Port 3 des Switches 4 angeschlossen, erkennt der Service den wieder integrierten Teilnehmer anhand der Topologieänderung. Der Service versucht nun per OPC UA auf die Kamera zuzugreifen, um das Informationsmodell herunterzuladen. Ist dies erfolgreich, wird der Status der Kamera von inaktiv auf aktiv gesetzt. Daraus resultiert, dass die benötigten Flows mit Hilfe der Einträge aus dem Informationsmodell auf den Switches installiert werden. Die Flows sind aufgrund der gleichen physischen Vernetzung von Produktionslinie 1 und 2 überwiegend konform zu den Angaben aus Tabelle 2. Lediglich die MAC-Adresse von SPS 1 ist nun diejenige von SPS 2 und die Flows werden auf den Switches 3 und 4 installiert.

4.4 Diskussion

Die Erstellung der SDN-Flows erfolgt in dieser ersten Implementierung noch semi-automatisch. Die jeweiligen Quell- und Ziel-MAC-Adressen werden aus dem Informationsmodell übernommen. Die Ports werden aus dem Netzwerkstatus, also der Topologie, extrahiert. Der Integrierungsmechanismus besitzt für diese erste Implementierung Kenntnis über die Topologie der Produktionslinien. Die jeweilige Anforderung und damit die Ziel-MAC-Adresse wird basierend auf der physikalischen Verbindung der Kamera mit Switch 2 oder 4 gewählt. Die Anzahl der Switches zwischen Kamera und SPS sowie die physikalische Verbindung ist dem Mechanismus ebenfalls bekannt.

Es zeigte sich bei der Implementierung, dass es zu spontanen Verbindungsabbrüchen der Zodiac FX Switches zum Controller kam. Zudem gestaltete es sich als schwierig, vier Switches gleichzeitig mit einem Controller zu verbinden. Meistens war die Southbound-Kommunikation nur mit weniger als vier Switches gleichzeitig möglich. Es mussten sowohl Controller als auch Switches mehrmals neu gestartet werden bis alle vier Switches mit dem Controller verbunden waren. Zudem bildeten sich gelegentlich nicht korrekte Schleifen in der Topologieaufzeichnung der OpenDaylight SDN-Plattform. Hierbei muss in Zukunft die Funktionalität der Plattform näher untersucht werden. Eine vollumfängliche Evaluierung gestaltete sich wegen der bereits beschriebenen Punkte als schwierig. Die Übertragung des Kamerabildes über die SDN-Switches war jedoch problemlos möglich. Die Übertragung war zudem erst möglich, sobald die Flows installiert wurden und vorher korrekterweise blockiert.

Des Weiteren kann gesagt werden, dass sich Python 3 als Programmierumgebung gut für die Implementierung eignet, die Integrierung der Schnittstellen erfolgte problemlos über die Python Software-Bibliotheken. In diesem Zusammenhang war auch die grundsätzliche Einbindung von OPC UA ohne großen Aufwand möglich und passend für den Use-Case. Das Northbound-Interface der SDN-Plattform ODL verwendet YANG als Modellierungssprache und REST als Schnittstelle, was als Basis verwendet werden kann, um aus der Netzwerkkommunikation eine Industrie 4.0-Komponente zu erstellen. Hierfür müssen die bereits modellierten Informationen auf industrielle Zweckmäßigkeit analysiert und gegebenenfalls erweitert werden. Eine weitere Möglichkeit wäre, ODL um eine OPC UA-Serverimplementierung zu erweitern, um eine Industrie 4.0-Verwaltungsschale zu erstellen.

5 Zusammenfassung und Ausblick

In diesem Beitrag wurde ein Integrierungsservice vorgestellt, der per OPC UA auf Charakteristika von Teilnehmern in einem Netzwerk zugreifen kann und darüber automatisch Konfigurationen für ein SDN-basiertes Netzwerk erstellt. Eine erste Implementierung des grundlegenden Mechanismus wurde anhand eines industriellen Use-case implementiert und beschrieben. Die Anreicherung eines Informationsmodells durch Eigenschaften von Netzwerkteilnehmern und deren Anforderungen an das Netzwerk kann eine Methode werden, um in Zukunft industrielle Netzwerke mittels Autokonfigurationsmechanismen effektiver administrieren zu können. Umrüstvorgänge im Sinne von Plug&Produce sind dadurch möglich.

In weiteren Arbeiten soll die initiale Registrierung von Netzwerkteilnehmern mittels SDN näher beleuchtet werden. Hierbei stehen Security-Mechanismen im Vordergrund und es muss abgeschätzt werden, wann ein potentieller Teilnehmer über das Netzwerk kommunizieren darf und wie ein solcher Registrierungsvorgang umgesetzt werden kann. Des Weiteren kann der Integrierungsmechanismus derart erweitert werden, dass er Verbindungsanfragen mit einem größeren Detailgrad verarbeiten kann.

Die konkrete Implementierung muss durch flexiblere Mechanismen zur Flow-Erstellung ausgestattet werden. Hierbei wird sich zuerst auf Algorithmen zur Pfaderstellung konzentriert. Des Weiteren wird die Informationsbasis über den Netzwerkstatus erweitert. Die Controller-Verbindung zu den Switches muss untersucht und robuster gestaltet werden. In weiteren Schritten werden die erstellten Mechanismen auf OPC UA-fähigen Industriekomponenten implementiert und im Feld getestet werden.

Forschungsaktivitäten können auf die semantische Beschreibung der Netzwerkteilnehmerinformationen und ihre Kombination gerichtet werden. Besonderer Augenmerk kann hierbei auch auf das Netzwerk selber gerichtet werden und es kann untersucht werden, wie eine Verwaltungsschale aus der Netzwerkkommunikation eine Industrie 4.0 Komponente erstellen kann.

Danksagung: Dieser Beitrag wurde im Rahmen des durch das BMBF geförderten Projekts FlexSi-Pro (Fördernummer 16KIS0650K) erstellt.

6 Literaturverzeichnis

[BE17] H. Bedenbender et al., „Industrie 4.0 Plug-and-Produce for Adaptable Factories: Example Use Case Definition, Models, and Implementation," Working Paper, Plattform Industrie 4.0, BMWI, Jun. 2017.

[KA14] G. Kálmán, D. Orfanus, und R. Hussain, „Overview and Future of Switching Solutions for Industrial Ethernet," in International Journal on Advances in Networks and Services, vol. 7, no. 3 & 4, 2014.

[HE16] D. Henneke, L. Wisniewski, und J. Jasperneite, „Analysis of Realizing a Future Industrial Network by Means of Software-Defined Networking (SDN)," IEEE World Conference on Factory Communication Systems (WFCS), Aveiro (Portugal), Mai 2016.

[HY18] S. Heymann et al., „Cloud-based Plug and Work architecture of IIC Testbed Smart Factory Web," IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), Turin (Italien), Sep. 2018.

[DI17] C. Diedrich et al., „Semantic interoperability for asset communication within smart factories," IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Limassol (Zypern), Sep. 2017.

[DI18] C. Diedrich et al., „Semantik der Interaktionen von I4.0-Komponenten," AUTOMATION – Leitkongress der Mess- und Automatisierungstechnik, Baden-Baden (Deutschland), Jul. 2018.

[PA18] S. K. Panda, Tizian Schröder, Lukasz Wisniewski, and Christian Diedrich, „Plug&Produce Integration of Components into OPC UA based data-space", International Conference on Emerging Technologies and Factory Automation (ETFA), Turin (Italien), Sep. 2018.

[PR17] S. Profanter, K. Dorofeev, A. Zoitl, und A. Knoll, „OPC UA for plug & produce: Automatic device discovery using LDS-ME," IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Limassol (Zypern), Sep. 2017.

[RA16] L. Rauchhaupt et al., "Network-based Communication for Industrie 4.0 – Proposal for an Administration Shell," Discussion Paper, Plattform Industrie 4.0, BMWI, Nov. 2016.

[HE18] D. Henneke, A. Brozmann, L. Wisniewski, und J. Jasperneite, „Leveraging OPC-UA Discovery by Software-defined Networking and Network Function Virtualization," IEEE International Workshop on Factory Communication Systems (WFCS), Imperia (Italien), Juni 2018.

[KR15] Kreutz et al., „Software-Defined Networking: A Comprehensive Survey," Proceedings of the IEEE | Vol. 103, No. 1, Jan. 2015.

[KA16] G. Kálmán, Prospects of Software-Defined Networking in Industrial Operations," International Journal on Advances in Security, vol. 9 no. 3 & 4, 2016.

[SC18] S. Schriegel, T. Kobzan, und J. Jasperneite, „Investigation on a Distributed SDN Control Plane Architecture for Heterogeneous Time Sensitive Networks," 14th IEEE International Workshop on Factory Communication Systems (WFCS), Imperia (Italien), Jun. 2018.

[CO17] J. Cox Jr. et al., „Advancing Software-Defined Networks: A Survey," IEEE Access, Oct. 2017.

[OF15] A. Nygren et al., „Openflow switch specification version 1.5.1," Open Networking Foundation, Tech. Rep., 2015.

[RA40] DIN SPEC 91345, „Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)," Apr. 2016.