Jürgen Jasperneite
Ulrich Jumar (Hrsg.)

# Kommunikation in der Automation

## Beiträge des Jahreskolloquiums KommA 2022, Lemgo

OPEN ACCESS

**Editor:**
Benedikt Lücke
*Institut für industrielle Informationstechnik - inIT | Technische Hochschule Ostwestfalen-Lippe*

# Impressum

13. Jahreskolloquium
**Kommunikation in der Automation**
(KommA 2022)

03. November 2022 • Lemgo

OPEN-Book

ISBN: 978-3-9818463-3-1
DOI: https://doi.org/10.25644/a4ws-9a49

Herausgeber:
Jürgen Jasperneite, Lemgo
Ulrich Jumar, Magdeburg

**Institut für industrielle Informationstechnik**
Technische Hochschule Ostwestfalen-Lippe

Campusallee 6
D-32657 Lemgo
Telefon: +49 5261 702-2400
Internet: www.init-owl.de | www.jk-komma.de

# Inhaltsverzeichnis


Seite 4 Vorwort

Seite 5 Komitees

Seite 6 Investigation of Ethernet TSN Interoperability in Industrial Multi-Protocol Networks

Seite 18 Planning an updated isochronous real-time schedule for a reconfiguration without interrupting the real-time communication

Seite 28 Traffic priority mapping for a joint 5G-TSN QoS model

Seite 39 Modelbasierte Fehlerursachenanalyse in zeitsensitiven Ethernet-Netzwerken

Seite 55 Practical investigation of an industrial converged network based on OPC UA PubSub and TSN

Seite 63 Open-Source Ethernet TSN-Testwerkzeuge für Linux

Seite 74 Speed advisor for buses to cross signaled intersection via V2I communication

Seite 86 Concept for Controller-Based Coexistence Management of Diverse Wireless Communication Systems

Seite 98 5G mmWave für industrielle Kommunikation

Seite 110 Latenzoptimierte, kontaktlose Datenübertragung für Industrial Ethernet

Seite 121 Entwurf eines einheitlichen Energieinformationsmodells für die Übertragung von Energieinformationen auf Basis von industriell genutzten Kommunikationsstandards

Seite 133 10BASE-T1L Plug-and-Play Architecture for I4.0 Field Devices over IO-Link

Seite 143 Nahfeldmesstechnik zur Designoptimierung und Bewertung von IoT-Komponenten

Seite 155 Security-Tests für Industriekomponenten nach IEC 62443 – Fuzzing-Testing von Kommunikationsschnittstellen

Seite 166 Describing wireless communication systems through the Asset Administration Shell

Seite 180 Model-Based Test Case Generation for Compliance Checking of Reactive Asset Administration Shells

Seite 192 An Asset Administration Shell Version Control to Enforce Integrity Protection

# Vorwort

## 13. Jahreskolloquium Kommunikation in der Automation – KommA 2022

Am 03. November 2022 fand am Institut für industrielle Informationstechnik - inIT der Technischen Hochschule Ostwestfalen-Lippe das 13. Jahreskolloquium der Reihe KommA – Kommunikation in der Automation statt. Das abwechselnd von den Instituten inIT, Lemgo und ifak, Magdeburg veranstaltete Kolloquium ist ein bewährtes Forum für Wissenschaft und Industrie zu allen technisch-wissenschaftlichen Fragen rund um die industrielle Kommunikation. Die Veranstaltung wird durch die ITG und die Gesellschaft für Informatik unterstützt.

Im Fokus des Jahreskolloquiums stehen Kommunikationssysteme – vom Feldbus, über Echtzeit-Ethernet bis zur drahtlosen Kommunikation und der Nutzung von IoT-Technologien. Bei der Systemanalyse und dem Entwurf von Kommunikationssystemen widmen sich die Tagungsbeiträge der formalen Modellierung, der Verifikation und Validierung sowie Interoperabilität, Konformität und Test. Als bedeutsame Aspekte vernetzter eingebetteter Echtzeitsysteme behandelt das KommA-Kolloquium u.a. die Dienstgüte, semantische Interoperabilität, Safety und Security, die Systemintegration und das Engineering. Neben aktuellen Kommunikationstechnologien bilden die vielfältigen Anwendungsbereiche der industriellen Kommunikation einen Schwerpunkt.

Unbenommen vom reichhaltigen Veranstaltungsangebot auf dem Gebiet der automatisierungstechnischen Kommunikation hat das Jahreskolloquium KommA seinen speziellen Platz. Neben den großen internationalen Tagungen wird bewusst der Austausch im kleinen Kreis gepflegt. Bei aller Technologie- und Anwendungsorientierung des Kolloquiums steht die Wissenschaftlichkeit im Fokus. Alle Beitragskurzfassungen werden deshalb von mindestens zwei Vertretern des Programmkomitees begutachtet. Der Verzicht auf parallele Sitzungen befördert den Gedankenaustausch.

Damit trotz der kurzen Tagungsdauer die wichtigsten Themen, Entwicklungen und Trends adressiert werden können, gehören neben den Vorträgen zusätzlich Poster zum Programm. Wie im Fall der Vorträge finden sich auch die Vollmanuskripte der als Poster präsentierten Beiträge im elektronischen Tagungsband.

**Prof. Dr. Jürgen Jasperneite**
*Institut für industrielle Informationstechnik - inIT*
*Technische Hochschule Ostwestfalen-Lippe,*
*Fraunhofer IOSB-INA*

**Prof. Dr. Ulrich Jumar**
*ifak e.V. - Institut für Automation und Kommunikation e.V.*
*Otto-von-Guericke-Universität Magdeburg*

# Komitees

## Tagungsleitung:

**Prof. Dr. Jürgen Jasperneite**

*Institut für industrielle Informationstechnik - inIT | Technische Hochschule Ostwestfalen-Lippe, Fraunhofer IOSB-INA*

**Prof. Dr. Ulrich Jumar**

*ifak e.V. - Institut für Automation und Kommunikation e.V. | Otto-von-Guericke-Universität Magdeburg*

## Programmkomitee:

**Stefan Bollmeyer**

*ABB Asea Brown Boveri Ltd*

**Holger Büttner**

*Beckhoff Automation GmbH & Co. KG*

**Prof. Dr. Christian Diedrich**

*Otto-von-Guericke-University*

**Prof. Dr. Mathias Fischer**

*Universität Hamburg*

**Prof. Dr. Mesut Günes**

*Otto-von-Guericke-University*

**Michael Höing**

*Weidmüller GmbH & Co. KG*

**Gunnar Lessmann**

*PHOENIX CONTACT Electronics GmbH*

**Dr. Malte Metzdorf**

*WAGO GmbH & Co. KG*

**Markus Rentschler**

*Murrelektronik GmbH*

**Detlef Tenhagen**

*HARTING Deutschland GmbH & Co. KG*

**Prof. Dr. Thilo Sauter**

*Technische Universität Wien*

**Dr. Sebastian Schriegel**

*Fraunhofer IOSB-INA*

**Prof. Dr. René Simon**

*Hochschule Harz*

**Prof. Dr. Henning Trsek**

*Institut für industrielle Informationstechnik -inIT*

*Technische Hochschule Ostwestfalen-Lippe*

**Dr. Christoph Weiler**

*Siemens AG*

**Prof. Dr. Jörg F. Wollert**

*FH Aachen*

**Prof. Dr. Martin Wollschlaeger**

*Techn. Universität Dresden*

## Organisationskomitee:

**Jasmin Zilz**

*Institut für industrielle Informationstechnik - inIT | Technische Hochschule Ostwestfalen-Lippe*

**Benedikt Lücke**

*Institut für industrielle Informationstechnik - inIT | Technische Hochschule Ostwestfalen-Lippe*

# Investigation of Ethernet TSN Interoperability in Industrial Multi-Protocol Networks

Janis Albrecht[1], Tarek Schlabeck[2], Jürgen Jasperneite[3]

**Keywords:** network convergence, Time Sensitive Networking (TSN), interoperability, PROFINET, OPC UA Field eXchange, CC-Link IE TSN

**Abstract**: Ethernet TSN is adapted by the automation industry. Both standardization and prototyping progresses as TSN-mechanisms are integrated with various application protocols and network technologies. This work investigates the consequences of sharing Ethernet TSN-mechanisms in multi-protocol networks and the impact on interoperability. For example, PROFINET and CC-Link are not interoperable in an Ethernet TSN network today. A detailed examination of TSN-mechanisms configuration parameters reveals areas of possible conflict. An exemplary side-by-side comparison of PROFINET, CC-Link IE TSN and OPC UA Field eXchange specifications shows concrete interoperability issues on selected Ethernet TSN-mechanisms that can be produced and measured in an experimental test environment. Based on the findings of the analyses of this work a solution discussion is started.

## 1 Introduction

Time Sensitive Networking (TSN) is a set of standards based on IEEE 802.1Q Ethernet network technology to address deterministic Quality of Service (QoS) requirements by introducing mechanisms for network traffic management and policing. Mechanisms like time synchronization, traffic scheduling or frame preemption are tools to shape Ethernet based traffic [IEEE22]. The IEC/IEEE 60802 TSN Profile for Industrial Automation defines a set of Ethernet TSN features as standardized infrastructure to facilitate convergence in industrial automation networks [In21a]. However, the current ongoing research and development phases within the industry as well as standardization work by different organizations also progresses at the same time as TSN profile development and may lead to diverging definitions and conflicting parallelism. An example of a framework focussed on converging IT and OT recently expanding on field level communication is OPC UA. Ethernet TSN is a key component for OPC UA Field eXchange (UAFX) to achieve vendor-independent

[1] Fraunhofer IOSB-INA, Campusallee 1, 32657 Lemgo, Germany, janis.albrecht@iosb-ina.fraunhofer.de
[2] Phoenix Contact Electronics GmbH, Dringenauer Straße 30, 31812 Bad Pyrmont, Germany, tschlabeck@phoenixcontact.com
[3] Fraunhofer IOSB-INA, Campusallee 1, 32657 Lemgo, Germany, juergen.jasperneite@iosb-ina.fraunhofer.de

interoperability [OP21]. Other fieldbus technologies adopting Ethernet TSN are PROFINET or CC-Link that has TSN-capable devices available on the market [In21b] [CLPA22]. The motivating questions of this work are posed by operating multi-protocol network environments: Are there configurations, topologies or combinations that could cause interoperability conflicts when Ethernet TSN-mechanisms are shared between protocols, and further, do potential conflicts contradict the principle of convergence by violating QoS requirements of applications? Conflict potential arises at protocol specification level. Differences in traffic types, configuration method and network model from protocol to protocol could potentially cause a collision of configuration, mechanism utilization, traffic prioritization and other areas of operation.
The goal of this paper is to identify and categorize areas of non-interoperability, determine concrete interoperability issues and discuss possible solutions for industrial Ethernet systems and networking technologies in the scope of industrial automation. To reach conclusions, analyses and comparison of standards and specifications are conducted. A set of popular networking technologies that utilize Ethernet TSN are selected for closer investigation in an operation scenario. In order to understand areas of non-interoperability and to develop solutions for conflicts it is necessary to analyse TSN-mechanisms on how they are used and configured. In the first step the relevant mechanisms are examined by configurable parameters to determine the exclusivity of a TSN-mechanism (section 3). This offers a generalized map for concrete conflict analyses of particular technologies in a side by side specification comparison. Exemplary networking technologies that utilize TSN are then compared afterwards (section 4). Possible solutions are discussed (section 5) and an experimental multi-protocol TSN-test-environment is constructed with available TSN hardware according to the identified areas of non-interoperability for solution development and validation (section 6). Fig. 1 illustrates Ethernet TSN resource usage by application protocols and potential conflicts.

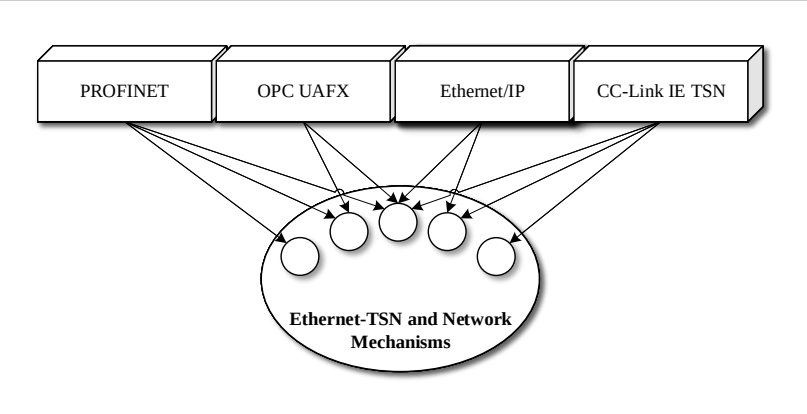



Fig. 1: Usage of TSN resources and potential conflict in a multi-protocol convergent Ethernet TSN network

# 2 State of the Art

In this section the terms interoperability, convergence, compatibility and co-existence are clarified and arranged. The resources of bridged networks are detailed and Ethernet TSN mechanisms and resources are examined for a generalized map of conflict areas. Protocols and networking technologies adapting Ethernet TSN are presented.

## 2.1 Interoperability, Convergence, Compatibility, Co-existence

A simple definition of interoperability in the context of communication systems can be: "*Interoperability between two or more information processing systems is given when the systems are able to exchange information in an expedient manner*"(translation by author) [Si19]. Co-existence is considered the lower level of interoperability with co-operation as upper level and is considered a more appropriate term regarding Ethernet TSN because it means operation without mutual interference [In18b] [Ha22]. Compatibility describes a compose-ability of components into a system in a 1:1 relationship [Gu15]. Convergence in context of communication systems describes the dissolving of the classical automation hierarchy into a more continuous communication model [Sc22]. Interoperability can be divided into four hierarchical layers. The lowest interoperability-layer is of structural nature. Connectors, cables and usage of uniform protocols (e.g. TCP/IP, HTTP) ensure information exchange capabilities. The layer above requires a common syntax for the information. This is achieved by uniform data formats. JSON established itself as the de-facto standard for this purpose in the IoT domain. The semantic layer requires a common understanding of meaning in order to enable interoperability on a higher level. The top layer organises processes and interactions in a efficient and effective manner. In so far, interoperability on the organisational layer (and in consequence each layer below) can be understood as a core-function to achieve the goals of Industry 4.0 (I4.0) [Si19]. Tab. 1 summarizes the layers of interoperability. The interoperability between standardized fieldbusses and realtime Ethernet systems using the same infrastructure is not given at this time but is an essential requirement for achieving the goals of I4.0 [OP21].

Tab. 1: Layers of Interoperability [Si19]

| Layer | Description |
|---|---|
| Organisational | Organisation of interacting systems |
| Semantic | A common interpretation of meaning |
| Syntactic | A common understanding of data-structures |
| Structural | Connectors, cables, protocols for data exchange |

The IEC and the IEEE identify three interdependent areas of interoperability in the TSN context: network configuration, stream configuration and establishment and application- / middleware configuration. In a layered structure these areas are on top of Ethernet-TSN. [In18b].

## 2.2 Bridged Network Resources

In this section Ethernet TSN resources are examined. In context of industrial automation application the features of TSN are used for internal and external real-time communication in a network of machines or between elements of one or multiple machines. According to [In18b] the elements of such an application can be controllers, sensors, actors, drives, Human-Machine-Interfaces (HMI), routers, bridges or other network equipment such as servers or diagnostic tools. The scope of this work focuses on bridges, although most resources on Ethernet TSN level are applicable to elements with Ethernet interfaces as well, say bridged controllers.

## 2.3 Ethernet TSN-Mechanisms and Resources

Ethernet TSN-mechanisms and their configurations can be interpreted as network resources. These configurations can either be static or dynamic, as well as global or granular. The following selected Ethernet TSN resources can be found in automation application elements:

- **Time Synchronization**: Time synchronization is a key mechanism of TSN. This work focuses on IEEE 802.1AS (gPTP). Critical Parameters and configuration options are the time domain and the method of configuration which is either dynamic or static. The exclusivity of the parameters is either per gPTP-instance or gPTP-port [In20]
- **Frame Preemption (FP)**: FP relies on the eMAC, pMAC and MAC-Merge Sublayer. The FP configuration is port- and queue-exclusive [In15] [In16b]
- **Time Aware Scheduling (TAS)**: TAS relies on Gate Control Lists that control the egress-queue-states of either closed or opened in a cyclic manner. TAS-parameters are port and queue exclusive [In16a]

Resources from the other layers may also carry conflict potential (e.g. concurrent network management protocols or a mix of TSN configuration models) but are not investigated in this work due to scope limitations. There are also more Ethernet TSN resources that carry conflict potential and that are not evaluated in this work, for example Per Stream Filtering and Policing (PSFP), scheduling algorithms or TSN-domains. Other network resources that are shared between protocols and constitute usage limits are bandwidth, buffers, memory and Filtering-Database-entries (FDB). Although an important field to investigate, this work is not focused on these resources.

## 2.4 Protocols over TSN

As TSN prototyping progresses within the automation industry, a number of vendors and organisations have shown interest to integrate Ethernet TSN with their application

protocols and are in the process of specifying the necessary adaptions. PROFINET is an industrial Ethernet standard designed for all branches of industrial automation from process and factory automation down to motion control. It is developed by PROFIBUS and PROFINET International (PI) and has a high growth rate within the industrial sector. The PROFINET specification version 2.4 is completed and includes TSN as an optional conformance class. Implementation demonstrations were shown at industrial fairs but there are no series-ready devices yet [PI22]. Control and Communications Link Industrial Ethernet (CC-Link IE) integrates TSN-mechanisms and defines certification classes for CC-Link IE TSN devices based on time synchronization and traffic scheduling in order to facilitate IT and OT convergence. Specifications regarding Ethernet TSN are available and a portfolio of CC-Link IE TSN devices is commercially available [CC20] [CLPA22]. OPC UA Field eXchange (UAFX) is an extension to the OPC UA framework to the field level. The goal is a worldwide vendor-independent industrial interoperability standard for industrial use-cases. The OPC Foundation released first versions of UAFX standardization with the 10000-8x series containing relevant TSN configurations [OP21]. Ethernet/IP is an industrial Ethernet variant based on ODVA´s Common Industrial Protocol (CIP). The ODVA has recently announced a committee to evaluate the adoption of TSN for Ethernet/IP technologies but will not specify the utilization of TSN before the release of IEC/IEEE 60802 (section 2.5) [Ha22] [ODVA22].

### 2.5 Converged Multi-Protocol Networks with IEC/IEEE 60802

The goal of the IEC/IEEE is to select a set of TSN mechanisms as a profile for industrial automation to unify the Ethernet TSN utilization and to facilitate interoperability. By conforming to the specified profiles and facets proposed by this standard interoperability issues can be avoided and minimized. The specifications are currently ongoing but draft versions exist. A release date has not been disclosed [In21a]. Parallel to the development of the profile there are already non-interoperable Ethernet-TSN solutions specified and market-available. The TSN Industrial Automation Conformance Collaboration (TIACC) consists of the AVNU Alliance, CC-Link Partner Association, ODVA, the OPC Foundation and PROFIBUS and PROFINET International. The goal is to provide a IEC/IEEE 60802 conformance test plan to ensure co-existence in multi-protocol networks [TIACC22].

## 3 Analyses of Basic Ethernet TSN Mechanisms and Resources for Conflict Potential

In this section the identified resources (section 2.3) are evaluated for their conflict potential by examining the exclusivity of configurable parameters. Tab. 2 summarizes the evaluations. For detailed parameter description see the corresponding standard. Time Synchronization and Traffic Types in particular carry severe conflict potential and are detailed in subsections 3.1 and 3.2.

Tab. 2: TSN resource conflict overview table

| TSN Resources | |
|---|---|
| **Resource** | **Exclusivity** |
| IEEE 802.1AS-2020 Time Synchronization | |
| priority1 | PTP-instance |
| priority2 | PTP-instance |
| domainNumber | PTP-instance |
| instanceEnable | PTP-instance |
| externalPortConfigurationEnabled | PTP-instance |
| ptpPortEnabled | PTP-port |
| delayMechanism | PTP-instance |
| meanLinkDelayThresh | PTP-port |
| delayAsymmetry | PTP-port |
| allowedLostResponses | PTP-port |
| allowedFaults | PTP-port |
| gPtpCapableReceiptTimeout | PTP-port |
| IEEE 802.1Qbu Frame Preemption | |
| preemptionActive | port |
| holdAdvance | port |
| releaseAdvance | port |
| framePreemptionStatusTable | port and queue |
| IEEE 802.1Qbv Time Aware Shaping | |
| queueMaxSDUTable | port and queue |
| GateEnabled | port |
| GateStates | port and queue |
| ControlListLength | port |
| ControlList | port |
| CycleTime | port |
| CycleTimeExtension | port |
| BaseTime | port |

### 3.1 Time Synchronization Conflict Potential

An optional state configuration option for gPTP is the external port state configuration. As an alternative to the Best Master Clock Algorithm (BCMA) it is possible to set up the gPTP-port-states of a time domain with a mechanism outside of the gPTP-standard. This allows for static, more precise configuration of the synchronization network than the algorithmic setup with the BMCA and gives control over each gPTP-ports state. The external configuration is activated by the corresponding parameter, see Tab. 2. Activating the external port configuration leads to altered function of different aspects of gPTP such as deactivation of particular state machines and the BMCA. In consequence, using the external

port configuration and the BMCA in the same time-domain is impossible and a severe resource conflict in context of Ethernet TSN applications [In19b]. The domainNumber parameter is sensitive: A mismatch of domain numbers leads to segmentation of the domains and non-interoperability. Another source of misconfiguration are the interval values for gPTP sync- and announce-messages. If there is a difference between the respective intervals the combination of interval and timeout factor leads to timeouts of the messages and dysfunctional time synchronization.

### 3.2 Traffic Type Conflict Potential

Traffic types are a subject of definition and prioritization. Depending on the context and the particular network traffic characteristics of a protocol or networking technology in addition to the requirements of the application, traffic types may vary from protocol to protocol. This leads to a concurrent utilization of the underlying resources, e.g. the expected handling of the network traffic by the infrastructure, and is a source of conflict [In18b]. From the perspective of the network (e.g. bridges) it is necessary to identify the traffic type of a network message in order to apply handling policies or TSN-mechanisms. For Ethernet TSN this can be done by the frame properties. VLAN-ID (VID) and VLAN-priority (TCI.PCP) can be used to distinguish a frame's traffic type, assign a traffic type to a hardware queue (traffic class) and segment streams of a particular traffic type within the network [In18a]. Other options for mapping a traffic type to a traffic class are the ethertype or Differentiated Services Code Point (DSCP, IP-header). The resources that are used by a traffic type depend on the TSN-mechanisms that are associated with the traffic type and the hardware queue (traffic class) the traffic type is mapped to. In consequence, independent and concurrent definitions of traffic types for different applications lead to reciprocal conflicts directly interfering with meeting application requirements. Conflict arises on three different dimensions:

- **Definition Conflict**: Two (or more) traffic types of different applications use the same traffic class but are defined differently (including identification method or value)
- **Traffic Class Conflict**: Two (or more) traffic types of different applications are share the same definition but are mapped to different traffic classes
- **Configuration Conflict**: Two (or more) traffic types of different applications share the same definition but use different TSN-mechanisms

All cases may lead to a failed configuration of the underlying network infrastructure or degradation of QoS. Because of the multidimensional conflict there are multiple solutions necessary to increase the interoperability and co-existence: unified traffic class mapping, unified traffic type definitions and unified QoS and TSN configurations per type.

## 4 Interoperability Investigation for Exemplary Protocol Standards

Three exemplary technologies are evaluated for existing interoperability issues in Ethernet TSN networks: PROFINET, UAFX and CC-Link IE TSN. To find non-interoperable specifications based on the previous identification and analyses of resources between these technologies the following questions are important: Which TSN-features does each technology utilize and how are they utilized? A comparison of the specifications and documentations is conducted for this purpose. This paper is focused on bridges in a single TSN-domain and an exemplary detail investigation of selected TSN-mechanisms: traffic types, time synchronization, TAS and Frame Preemption. Tab. 3 shows the utilization of the selected TSN-mechanisms.

Tab. 3: Ethernet TSN utilization table for PROFINET, CC-Link IE TSN and UAFX [In21b] [CC20] [OP22]

| **TSN-Mechanisms** | **PROFINET** | **CC-Link IE TSN** | **UAFX** |
|---|---|---|---|
| Time Synchronization | ✓ | ✓ | ✓ |
| TAS | ✓ | ✓ | ✓ |
| Frame Preemption | ✓ | x | ✓ |
| Traffic Type Definition | ✓ | x | ✓ |

A particular conflict exists between CC-Link IE TSN and PROFINET general utilization of **Frame Preemption and TAS**: PROFINET utilizes TAS only at link speeds lower than 1Gb/s while CC-Link IE TSN only utilizes TAS in general. A severe interoperability issue can also be identified with **time synchronization** parametrization between UAFX, CC-Link IE TSN and PROFINET. The latter uses different logSyncInterval timings of 31,25ms while the former both apply gPTP standard values of 125ms. This discrepancy in the interval leads to a dysfunctional time synchronization between devices [In21b]. A **traffic type** comparison and resulting conflicts between PROFINET and UAFX is shown in Tab. 4.

Tab. 4: Comparison of Traffic Type to Traffic Class mapping for UAFX and PROFINET [In21b] [OP22]

| **Traffic Class** | **PCP / PROFINET Type** | **PCP / OPC UA Type** |
|---|---|---|
| 7 | 6 Isochronous | 7 - |
| 6 | 5 Cyclic Synchronous | 6 - |
| 5 | 7 Network Control | 5 Network Control |
| 4 | 4 Cyclic Asynchronous | 4 OPC UA PubSub |
| 3 | 3 Alarms / Events | 3 - |
| 2 | 2 Configuration | 2 Configuration |
| 1 | 1 Best Efford+ | 0 Best Efford+ |
| 0 | 0 Best Efford- | 1 Best Efford- |

# 5 Proposal of Interoperability Solutions

Conflicting usage of TSN-resources impacts the interoperability of multiple protocols in a single TSN-domain. Many conflicts are solved by consistent and unified understanding of **traffic types** and the mapping to traffic classes. The IEC/IEEE 60802 suggests both a set of traffic type definitions and their mappings to traffic classes as well as a method for mapping application defined traffic types to the network configuration. The latter is proposed by a traffic type translation table. The same principle can be used for **time-domain** conflicts [In21a]. However, existing TSN-capable devices do not implement such tables at the time of this work. As the interoperability issues mainly occur on concurrent Ethernet interface usage a solution could be a configuration-merge algorithm to produce a QoS-configuration for TSN mechanisms with multiple sets of QoS-requirements as input. The algorithm can be a stand alone implementation or included on network management level (e.g. Centralized Network Configuration). Further, the translation table for traffic types does not solve the configuration conflict described in section 3.2. The Industrial Internet Consortium proposed a unification and TSN utilization per traffic type [In19a]. This is not necessary if enough Ethernet interfaces are available in the network. Classic methods of separation and segmenting the network can be a solution in this case. To solve interoperability issues that involve the difference in utilization of **Frame Preemption and TAS** a simple approach can be to allow the configuration of each mechanism outside of specification limitations.

# 6 Test Environment for Multi-protocol Network Solutions

For measuring the behaviour of Ethernet TSN based traffic in a multi-protocol network environment a test setup is required. This section describes the laboratory conditions, the chosen network infrastructure and devices, the composition of communication technologies. The goal is to share common Ethernet TSN resources as summarized in Tab. 2 between different technologies at the same time and observe the impact on interoperability. This allows to evaluate the severity of resource conflicts and to develop solutions for particular conflicts.

## 6.1 Measurement Tools and Topology

In order to produce precise measurements of network traffic on an oscilloscope a proprietary FPGA-based solution developed at Fraunhofer IOSB-INA is deployed. The TSN-Monitor allows free configuration of traffic filters on the basis of frame attributes and visualization by an oscilloscope. A proposed topology to investigate interoperability conflicts in a multi-protocol environment is shown in Fig. 2. Different applications share Ethernet interfaces over two Ethernet TSN bridges. For this purpose the FL2316 TSN switch by Phoenix Contact was selected as the underlying network infrastructure.

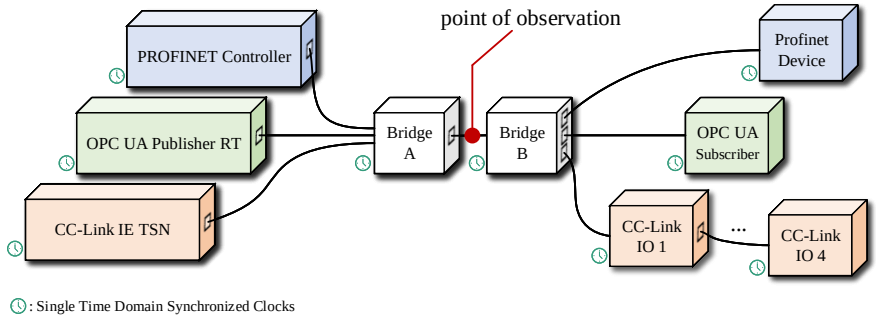



Fig. 2: Composition and topology of a laboratory test environment

### 6.2 Exemplary Validation of Outside-of-Specification Solution

Fig. 3 shows the utilization of Frame Preemption for CC-Link IE TSN messages outside of the CC-Link specifications that would increase interoperability between CC-Link IE TSN and PROFINET.

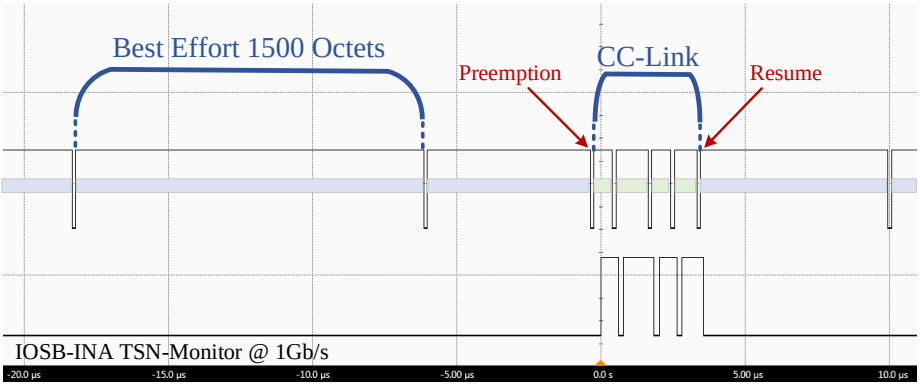



Fig. 3: CC-Link IE TSN messages configured as express traffic via ethertype-to-queue mapping @ 1Gb/s link speed outside of specification preempting best effort traffic

## 7 Conclusion and Future Work

The adaption of Ethernet TSN within the automation industry progresses without question on specification level and prototyping. TSN mechanisms and resources are utilized in different ways by application protocols and networking technologies. This may lead to interoperability issues in multi-protocol network environments. This work identified Ethernet TSN mechanisms, interpreted them as network resources and presented areas of interoperability conflicts. Further analyses of PROFINET, OPC UA Field eXchange

and CC-Link IE TSN specifications based on the previous steps were conducted. It could be shown that concrete interoperability issues exist in different areas. A topology for an experimental test environment is proposed for developing interoperability issue solutions. An exemplary validation of an outside-of-specification solution was successfully conducted. As the topic of Ethernet TSN interoperability spans different perspectives and a multitude of resources, this work offers an entry point for future investigations in an experimental multi-protocol setup to accompany Ethernet TSN development of the automation industry.

## References


[CC20] CC-Link Partner Association CLPA: CC-Link IE TSN Compatible Product Development Method Guide. 2020.

[CLPA22] CC-Link Partner Association CLPA: CC-Link IE TSN, 2022, URL: https://eu.cc-link.org/de/cclink/cclinkie/cclinkie_tsn, visited on: 08/10/2022.

[Gu15] Gunzert, M.: Compatibility and interoperability in field device integration - A view on EDDL, FDT and FDI. Institute of Electrical and Electronics Engineers (IEEE), 2015.

[Ha22] Hantel, M., Woods, J.: The Integration of Time-Sensitive Networking into EtherNet/IP™ Technologies. 2022.

[IEEE22] Institute of Electrical and Electronics Engineers IEEE: Time-Sensitive Networking (TSN) Task Group, 2022, URL: https://1.ieee802.org/tsn/, visited on: 03/17/2022.

[In15] Institute of Electrical and Electronics Engineers (IEEE): IEEE Std 802.1Qbu-2016 (Amendment to IEEE Std 802.1Q-2014): IEEE Standard for Local and metropolitan area networks - Bridges and Bridged Networks - Amendment 26: Frame Preemption. 2015.

[In16a] Institute of Electrical and Electronics Engineers (IEEE): IEEE 802.1Qbv Standard for local and metropolitan area networks - bridges and bridged networks - Amendment 25: Enhancements for Scheduled Traffic. 2016.

[In16b] Institute of Electrical and Electronics Engineers (IEEE): IEEE Std 802.3br-2016 Draft Standard for Ethernet Amendment: Specification and Management Parameters for Interspersing Express Traffic. 2016.

[In18a] Institute of Electrical and Electronics Engineers (IEEE): IEEE 802.1Qcc Standard for Local and metropolitan area networks - Bridges and Bridged Networks. 2018.

[In18b] International Electrotechnical Commission and Institute of Electrical and Electronics Engineers IEC/IEEE: 60802 Industrial Use Cases V1.3. 2018.

[In19a] Industrial Internet Consortium (IIC): Characterization and Mapping of Converged Traffic Types. 2019.

[In19b] Institute of Electrical and Electronics Engineers (IEEE): IEEE 1588-2019 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. 2019.

[In20] Institute of Electrical and Electronics Engineers (IEEE): IEEE 802.1AS-2020 Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications. 2020.

[In21a] International Electrotechnical Commission and Institute of Electrical and Electronics Engineers IEC/IEEE: 60802 TSN Profile for Industrial Automation. 2021.

[In21b] International Electrotechnical Commission IEC: Profinet Specification. 2021.

[ODVA22] ODVA: EtherNet/IP™, 2022, URL: https://www.odva.org/technology-standards/key-technologies/ethernet-ip/, visited on: 08/10/2022.

[OP21] OPC Foundation: Extending OPC UA to the field: OPC UA for Field eXchange (FX). 2021.

[OP22] OPC Foundation: OPC Unified Architecture Field eXchange (UAFX) Part 82: UAFX Networking. 2022.

[PI22] PROFIBUS and PROFINET International (PI): PROFINET - the leading Industrial Ethernet Standard, 2022, URL: https://www.profibus.com/technology/profinet/overview, visited on: 08/10/2022.

[Sc22] Schriegel, S.: Kompatibilitätsverfahren für Profinet-Hardware mit Ethernet Time Sensitive Networks. Springer Vieweg, 2022.

[Si19] Sinsel, A.: Das Internet der Dinge in der Produktion: Smart Manufacturing für Anwender und Lösungsanbieter. Springer Vieweg, 2019.

[TIACC22] TSN Industrial Automation Conformance Collaboration (TIACC): About TSN Industrial Automation Conformance Collaboration, 2022, URL: https://www.tiacc.net/about, visited on: 08/10/2022.

# Planning an updated isochronous real-time schedule for a reconfiguration without interrupting the real-time communication

Christian von Arnim [1], Armin Lechler [2], Oliver Riedel [3]

**Abstract:** One part of the configuration of isochronous real-time networks is the scheduling of time. The schedule fulfills real-time communication requirements by specifying at which times data is transported on which paths in the real-time network. This paper describes an approach to generate a schedule when requirements have been changed, that allows a transition from the currently existing schedule to a new schedule fulfilling all existing requirements at any point in time. This is achieved by using a Satisfiability Modulo Theories (SMT) solver to restrict the solution space to those solutions that can be achieved without violating the existing real-time network requirements. Subsequently, the steps leading from the current schedule to the newly computed schedule are determined.

**Keywords:** scheduling; real-time network; isochronous; time-sensitive-networking

## 1 Introduction

The digitalization of the factory in the evolution of Industry 4.0 provides the possibility to produce customized products. Depending on the level of customization, a product may require a reconfigurable production [Ma21]. Such a reconfiguration may include hardware, software (e.g., controller applications) and the real-time communication. These reconfigurations must be applied during operation without downtime of the production [GGS21]. In case that changes to the real-time communication are done, it must be ensured that the existing communication is not interrupted during the reconfiguration. Real-time networks that are used by multiple participants can be realized by time-division multiple access (TDMA). TDMA ensures fixed latencies by reserving exclusive time slots for applications. Fig. 1 shows a TDMA schedule for three time slots, which may be used by three independent applications. The first time slot A starts right at the beginning of the communication cycle and has a size of 3. Time slot B begins at offset 4 and also has a size of 3. Finally, time slot C ranges from offset 8 to 11. This pattern is repeated in the following communication cycles.

When this schedule of the TDMA communication needs to be updated due to new requirements, it must be ensured that all operational real-time applications still get their data

[1] Institute for Control Engineering of Machine Tools and Manufacturing Units, University of Stuttgart, Seidenstr. 36, 70174, Stuttgart, christian.von-arnim@isw.uni-stuttgart.de
[2] armin.lechler@isw.uni-stuttgart.de
[3] oliver.riedel@isw.uni-stuttgart.de

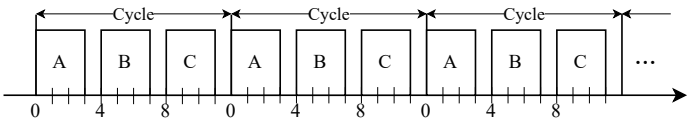


Fig. 1: TDMA communication with three reserved time slots.

in time. For this reason, an arbitrary modification is not possible [vLR21]. The following paper shows how such a reconfiguration can be planned without interrupting the operational communication. Section 2 introduces several implementations of real-time networks. In Section 3 a model for a TDMA real-time network is described. Section 4 introduces an operation for modifying an existing real-time schedule, which is integrated into the model in Section 5. Section 6 describes an algorithm to determine the individual reconfiguration steps. Finally, the paper is summarized in Section 7.

## 2 State of the Art

There exist multiple field bus systems, which support real-time communication. Open Platform Communication Unified Architecture (OPC UA) is one technology to exchange data between machines. Aside from the Client/Server communication, OPC UA does also specify a publisher/subscriber communication (OPC UA PubSub), that can be used together with Time-Sensitive Networking (TSN) to exchange data in real-time as described in the OPC UA Field Exchange standards (OPC UA FX). The real-time performance is achieved by using an exclusive TSN traffic class for the OPC UA PubSub traffic to separate the real-time traffic from other traffic in the network [OP22b], so that OPC UA PubSub traffic is only separated from non-OPC UA PubSub traffic, but not from OPC UA PubSub traffic of other devices. The connection between multiple automation components is established by the OPC UA ConnectionManager, which is capable to start and stop certain connections [OP22a]. The specification focuses on a configuration ahead of the communication in engineering tools. It is possible to establish additional connections at a later time, but this may interfere with the existing OPC UA PubSub communication, as the traffic class does not separate the individual OPC UA PubSub communication between different devices.

Profinet over TSN is another real-time communication field bus which is using TSN. In Profinet, the isochronous TDMA communication is possible using Profinet IRT, which reserves all required time slots at the beginning of the cycle, according to DIN EN 61158-5-3 [PR]. Profinet supports a dynamic reconfiguration of the applications, but this will interrupt the IRT communication[DJF15]. Therefore, a continuous communication is not possible.

Jatzkowski; Kleinjohann [JK14] describes the reconfiguration of a real-time network of cyber physical systems without modifying the existing communication. In case there is no adapted schedule, which fulfills the requirements of an additional communication, the integration is aborted.

## 3 Calculating a TDMA Schedule

Calculating a schedule for TDMA real-time networks has been shown in [St10]. This paper uses a similar simplified model to describe time slots in the real-time network. Thereby, the focus lies on the network schedule, and conditions such as limited available memory are not considered in order to keep the model simple. The model is solved using the Z3 SMT solver. In case a more advanced model is required, additional conditions can be added to the model and similarly solved by an SMT solver.

In the following, the network model is described. The network is modeled as an undirected graph $G$ with endpoints $ep$ and bridges $br$, as the nodes and connections $c$ (see (1) and Fig. 2). Each connection in the full-duplex network consists of two unidirectional links $(l_{x1}, l_{x2})$. A stream $S$ is unidirectional, connects two endpoints and uses a set of links. Such a stream has a cycle time $t_{cycle}$, a frame size $f$ and a maximum latency $t_{lat}$ (see (2)). On each link, the stream requires a time slot $s$ which is described by an offset in the cycle $t_{off}$ and a duration $d$ (see (3)). The duration is determined by the frame size and the connection speed $c_s$ (see (4)). As on a single link only one frame can be transmitted at the same time, two time slots on the same link must not overlap (5). Finally, a bridge has a certain amount of forwarding delay $d_{fd}$, to forward a frame. This duration can be considered constant when using the cut-through strategy which forwards a package as soon as the destination address is received without waiting to receive the full package (see (6)). In the following, a network with cut-through and a constant forwarding delay is assumed. Such a network provides the lowest possible latencies, as no data is queued on any bridge.

$$G = (\{ep, br\}, c) \tag{1}$$

$$S = ([l_x, l_y, ...], t_{cycle}, f, t_{lat}) \tag{2}$$

$$s = (t_{off}, d) \tag{3}$$

$$d = \frac{f}{c_s} \tag{4}$$

$$\forall (s_1, s_2) \in l, s_1 \neq s_2 : \neg overlapp(s_1, s_2) \tag{5}$$

$$\forall s_1, s_2 \in S, s_2 = successor(s_1) : t_{off}(s_2) - t_{off}(s_1) \geq d_{fd} \tag{6}$$

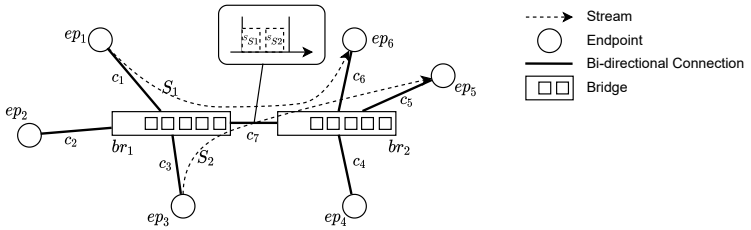



Fig. 2: Model of the TDMA network

In case the streams have different cycle times, a hyper period can be defined as the least common multiplier of all cycle times [St10]. In the following, a uniform cycle time is assumed for all streams.

To achieve a schedule for the network, the model is solved using the Z3 SMT solver. This provides offsets for all time slots on all links in the network. These offsets need to be transformed to a network configuration and deployed to all network devices. This process depends on the network technology and is outside the scope of this paper (e.g., Metaal et al. [Me20] has shown this process for a TSN network).

## 4 Rescheduling During Operation

Adjustments to the network schedule may be necessary to respond to changing requirements of a network. This section describes an operation, which allows the adaptation of a network schedule at runtime without interrupting the real-time communication. Real-time application requires, that exchanged data using the real-time network has a maximum transmission time, and the duration between two exchanged data packages must be constant to ensure a guaranteed reaction time. However, as real world applications always have some amount of jitter, real-time applications also need to tolerate a certain amount of deviations in the timing of the received data. von Arnim et al. [vLR21] uses these tolerances to adjust the schedule of an existing real-time network across several cycles. This concept is shown in Fig. 3. In the initial cycle $c_0$ the three time slots A, B and C are located at the offsets 0, 4 and 8. In cycle $c_1$, time slot C is scheduled slightly earlier (e.g., 10 ns). This process is repeated in cycle $c_2$, where time slot C is scheduled even earlier than before, until at cycle $c_n$ time slot C is located at offset 7 right after time slot B.

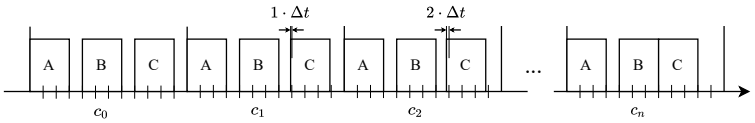


Fig. 3: The position of time slot C is adjusted a little bit several times in a row.

This concept allows the modification of an existing schedule during operation and provides a greater flexibility to fulfill changed requirements. Hereby, any existing time slot can be moved to a different location in the cycle, if there are no other time slots between the current and the target location.

## 5 Calculating an Updated Network Schedule

This section describes how an updated schedule can be calculated. This updated schedule fulfills all requirements and can be reached using the operation from Section 4. At first, some

possible and impossible schedules are regarded. Based on this, additional constraints for the SMT solver are defined. The SMT solver provides a solution which can be accomplished by shifting existing time slots. Finally, the individual steps are calculated to get a sequence of shifts for the existing time slots.

Fig. 4 (i) visualizes a cycle with a length of 12 containing three time slots A, B and C scheduled with a length of 3 at offsets 0, 4, and 8, respectively. This existing schedule should be extended by an additional time slot D with a length of 3. In this existing schedule, there is not enough space for the additional time slot D, as the existing schedule is fragmented and there is no consecutive space of 3 available. Using the operation introduced in Section 4, time slot B may be moved to offset 3 and time slot C may be moved to offset 6. This leads to a consecutive space of size 3, beginning at offset 9 to insert time slot D as visualized in Fig. 4 (ii). Similarly, the configuration Fig. 4 (iii) can be reached. At first, time slot C is moved to offset 9, then, time slot B is moved to offset 6 and time slot A is moved to offset 3. Afterwards, time slot C is moved across the cycle border to offset 0 and B is moved to offset 9. Finally, time slot D can be inserted at offset 6. Unlike the schedules (ii) and (iii) in Fig. 4, the configuration (iv) can not be achieved by the reconfiguration operations. The time slot A and C are, compared to (iii), swapped, which is not possible by movements, as time slot A and C would have been overlapped during this process. So, possible reconfigured schedules are characterized by the fact that the order of the existing time slots remains intact. In the original schedule in Fig. 4 (i), the order of the time slots is ABC. The same order occurs in Fig. 4 (ii). In Fig. 4 (iii), the order of the time slots is CAB. But as all time slots are transmitted cyclically, time slot C is also sent right after time slot B at the beginning of the next cycle. So, when choosing time slot A as a reference, the order of the time slots is ABC. In Fig. 4 (iv) the order of the time slots is ACB - this would require a swap of time slots when transforming from the initial schedule to this schedule.

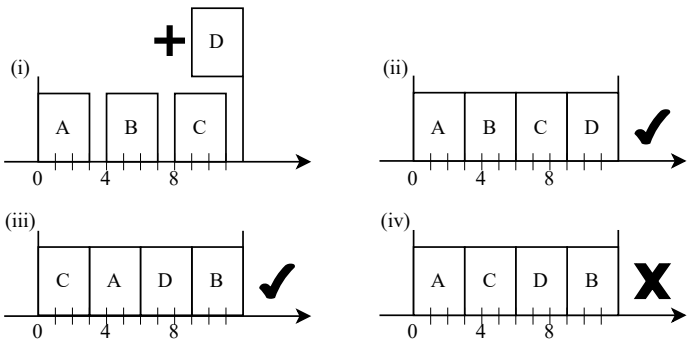



Fig. 4: An initial schedule (i) where an additional time slot D should be inserted. (ii) and (iii) are possible reconfigured schedules, but (iv) is impossible to reach through a reconfiguration.

Next, the conditions are defined for the SMT solver, which must be met by all existing time slots. An existing time slot $s_0$ is chosen on each link. The duration $\Delta t(s, s_0)$ is defined in (7). It describes the duration from the time slot $s_0$ to the beginning of the time slot $s$ (see

Fig. 5 (i)), which may be in the following cycle (see Fig. 5 (iii)). This duration is defined as the difference between the offsets of time slot $s_0$ and time slot $s$. If this offset is negative, the cycle time is added to ensure that the duration is within $[0, t_{cycle}]$.

The time slots preserve the order in the schedule when for each pair of time slots on a single link in the original schedule $s_{orig1}, s_{orig2}$ and the corresponding pair of time slots in the calculated updated schedule $s_1, s_2$ either $\Delta t(s_1, s_0)$ is smaller than $\Delta t(s_2, s_0)$ in the original and the new schedule, or $\Delta t(s_1, s_0)$ is bigger than $\Delta t(s_2, s_0)$ in the original and the new schedule.

$$\Delta t(s, s_0) = (t_{off}(s) - t_{off}(s_0) + t_{cycle}) \bmod t_{cycle} \tag{7}$$

$$s_0, s_{orig0} \in l \forall s_1, s_2, s_{orig1}, s_{orig2} \in l :$$

$$\Delta t(s_{orig1}, s_{orig0}) < \Delta t(s_{orig2}, s_{orig0}) \Leftrightarrow \Delta t(s_1, s_0) < \Delta t(s_2, s_0)$$

$$\Delta t(s_{orig2}, s_{orig0}) < \Delta t(s_{orig1}, s_{orig0}) \Leftrightarrow \Delta t(s_2, s_0) < \Delta t(s_1, s_0) \tag{8}$$

This is illustrated in Fig. 5 using the example from Fig. 4. The time slot A is selected as the reference. In the original schedule (i), the time difference $\Delta t(A, B)$ between time slot A and B is smaller than the time difference $\Delta t(A, C)$. Therefore, in all achievable schedules, $\Delta t(A, B)$ must be less than $\Delta t(A, C)$. This condition holds for (ii) and (iii), but not for the unreachable schedule (iv).

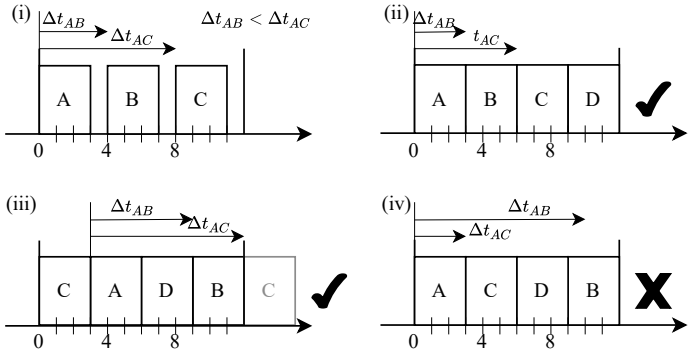



Fig. 5: The Condition $\Delta t_{AB} < \Delta t_{AC}$ is fulfilled if the order of the existing time slots A, B and C is the same.

The conditions (8) are added to the model from Section 3. Thus, the solutions, computed by the SMT solver are limited to schedules which are reachable by the current schedule using the operation described in Section 4.

## 6 Determine the Reconfiguration Steps

The solution from the SMT solver provides a valid reachable schedule, which contains an offset for each stream. This section describes an algorithm, which calculates the steps to

achieve this calculated schedule from the original one. Each step describes a shift of a single stream, which affects all time slots of this stream. Thereby, it is required that, at each point in time during the reconfiguration of the schedule, there is a valid schedule for the original set of streams. So, time slots must not overlap at any time. Hence, shifting of a stream to a new offset is only possible, if on all links there is no time slot between the original and the target offset.

The streams are divided in two sets. The first set contains all existing streams, the second set all newly added streams. At first the existing streams are shifted to their new offset in the cycle, then, the new streams are added. In case all time slots of all existing streams do only require to be shifted within the cycle and no time slot needs to cross the border between two cycles (e.g., Fig. 4 (ii)), each stream needs to be shifted only once to reach its target offset. The required steps are shown in Fig. 6. At first, stream B is shifted to offset 3, afterwards, stream C is shifted to offset 6, and finally, stream D is inserted.

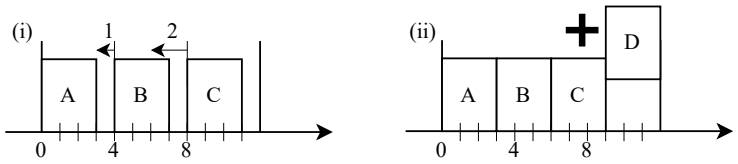



Fig. 6: The steps to achieve the schedule of Fig. 4 (ii)

In case some time slot of some streams needs to be shifted across the border of a cycle (e.g., stream C in Fig. 4 (iii)), shifting a time slot directly to its target offset might not be always possible. List. 1 shows an algorithm that calculates the steps towards the target schedule. The basic idea is that all streams are shifted in the same direction. In case the stream can not reach its target offset directly due to another stream, this other stream is also shifted, even if it is already at its desired offset.

Fig. 7 shows the application of this concept for the example from Fig. 4 (iii). In this example and the following description of the algorithm, all streams are only shifted towards the beginning of the cycle. Shifting all streams to the other direction is possible in a similar way. In the initial schedule (see Fig. 7 (i)), none of the three time slots can be shiftet to its target offset. Stream C is chosen arbitrarily and shifted until stream B prevents a further movement (see Fig. 7 (ii)). To enable further shifting of stream C, stream B is shifted before stream C. This is possible until stream B is blocked by stream A, as shown in Fig. 7 (iii). So, first stream A and afterwards stream B is shifted to enable further shifting of stream C. This sequence is repeated once more (see Fig. 7 (iv)) until stream C reaches its target offset in the cycle as shown in Fig. 7 (v). At this stage, only stream A needs to be shifted once more before the new stream D can be added to the schedule in Fig. 7 (vi).

The formal algorithm is shown in List. 1. It uses two functions `RequiredShift` and `GetMaxPossibleShift`, which are illustrated in Fig. 8. The function `RequiredShift` calculates for a stream (B) and the target offset (targetOffset), the required shift (shiftStream).

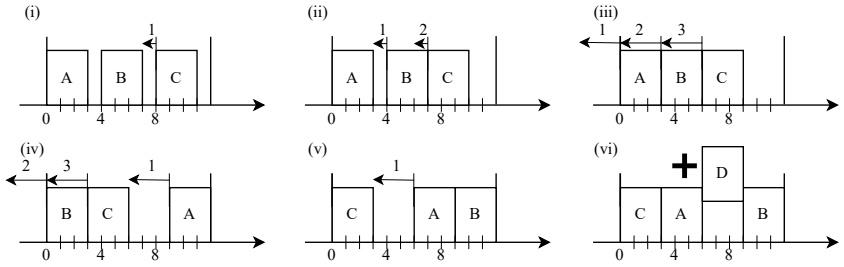


Fig. 7: The steps to achieve the schedule of Fig. 4 (iii)

`GetMaxPossibleShift` provides the largest possible shift (maxShift) and the restricting stream for a given stream (B). While not all existing streams are at their target offset, each stream, which is either not at its target offset or is blocking another stream, is trying to be shifted. To achieve this, at first, the required shift and the maximum possible shift is calculated. If the required shift is larger than the possible shift, the shift is reduced and the blocking stream is added to the list of blocking streams. If the stream can be shifted sufficiently, it is removed from the blocking streams. Each shift is appended to the list of steps. After all streams have reached their target offset, the new streams are added. When the algorithm has been completed, the list of steps contains a sequence of steps that leads from the initial schedule to the calculated schedule of the SMT solver.

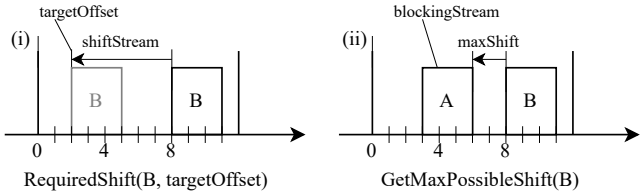


Fig. 8: The parameters and return values of the functions `RequiredShift` and `GetMaxPossibleShift`.

```
blocking = []
steps = []
while not all existing streams at desired offset
  for stream in blocking ⋃ NotAtTargetOffset
    shiftStream = RequiredShift(stream, targetOffset)
    // shiftStream < 0, as the time slots are shifted towards the
        begining of the cycle
    maxShift, blockingStream = GetMaxPossibleShift(stream)
    if shiftStream < maxShift
      shiftStream = maxShift
      blocking.Add(blockingStream)
    else
      blocking.Remove(stream)
```

```
    stream.Offset += shiftStream
    stream.Offset %= CycleTime // Keep Offset within [0;CycleTime)
    steps.Append(MoveStream(stream, shiftStream))

for stream in AdditionalStreams
  steps.Append(InsertStream(stream))
```

List. 1: Algorithm for getting the steps towards the target configuration.

To visualize the individual steps of the reconfiguration of the schedule, a visualization has been created. In the visualization, the individual steps on links are shown. This enables an easy check, if all the steps are valid according to the restrictions. A screenshot of the visualization is shown in Fig. 9. The individual steps can be selected on the top, the corresponding schedule is visualized below.

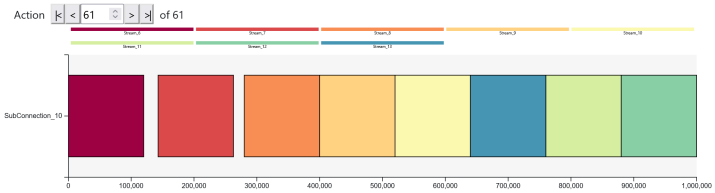



Fig. 9: Screenshot of the step visualization

## 7 Summary and Further Work

Real-time networks offer a bounded guaranteed latency for the communication. Adjustments during operation are limited, since guaranteed latencies must also be maintained during reconfiguration. The model of a TDMA network can be extended to calculate a schedule with the help of an SMT solver. This calculated schedule can be achieved by an existing one while maintaining the guaranteed latencies. Afterwards, the introduced algorithm can provide a sequence of steps to transform an existing network schedule to the new network schedule.

The deployment of this sequence of steps to a real-time network is subject to further work. This requires the endpoints and bridges of the real-time network to execute the individual steps synchronously. This may be done by an extension of TSN networks. In addition, the algorithm to determine the individual steps can be optimized, e.g., to determine which steps can be extended in parallel to decrease the total time of the reconfiguration.

## Acknowledgment

The presented work was funded by the Ministry of Science, Research and the Arts of the Federal State of Baden-Württemberg within the 'Innovation Campus Future Mobility', to which we gratefully acknowledge the support.

## References

[DJF15] Durkop, L.; Jasperneite, J.; Fay, A.: An analysis of real-time ethernets with regard to their automatic configuration. In: WFCS 2015. IEEE, Piscataway, N.J., pp. 1–8, 2015.

[GGS21] Gundall, M.; Glas, C.; Schotten, H. D.: Feasibility Study on Virtual Process Controllers as Basis for Future Industrial Automation Systems. In: 2021 22nd IEEE International Conference on Industrial Technology (ICIT). IEEE, pp. 1080–1087, 2021.

[JK14] Jatzkowski, J.; Kleinjohann, B.: Towards Self-reconfiguration of Real-time Communication within Cyber-physical Systems. Procedia Technology 15/, pp. 54–61, 2014.

[Ma21] Mayrhofer, M.; Mayr-Dorn, C.; Guiza, O.; Egyed, A.: Dynamically Wiring CPPS Software Architectures. In: 2021 22nd IEEE International Conference on Industrial Technology (ICIT). IEEE, pp. 1060–1067, 2021.

[Me20] Metaal, M. A.; Guillaume, R.; Steinmetz, R.; Rizk, A.: Integrated industrial Ethernet networks: Time-sensitive networking over SDN infrastructure for mixed applications. In: 2020 IFIP Networking Conference (Networking). Pp. 803–808, 2020.

[OP22a] OPC Foundation: OPC Unified Architecture Field eXchange (UAFX) 1.0.0 RC3: Part 81: UAFX Connecting Devices and Information Model./, 2022.

[OP22b] OPC Foundation: OPC Unified Architecture Field eXchange (UAFX) 1.0.0 RC3: Part 82: UAFX Networking./, 2022.

[PR] PROFIBUS Nutzerorganisation e. V.: PROFINET IRT Engineering Verison 1.35: Guideline for PROFINET, URL: https://de.profibus.com/downloads/profinet-irt-engineering-guideline.

[St10] Steiner, W.: An Evaluation of SMT-Based Schedule Synthesis for Time-Triggered Multi-hop Networks. In: The 31st IEEE Real-Time Systems Symposium. IEEE Computer Society, Los Alamitos, California, pp. 375–384, 2010.

[vLR21] von Arnim, C.; Lechler, A.; Riedel, O.: Operations for non-disruptive modification of real-time network schedules. In: 2021 22nd IEEE International Conference on Industrial Technology (ICIT). IEEE, pp. 1131–1137, 2021.

# Traffic priority mapping for a joint 5G-TSN QoS model

Niklas Ambrosy[1] and Lisa Underberg[2]

**Abstract:** In order to integrate 5G mobile radio into Time-Sensitive Networking (TSN), the 3rd Generation Partnership Project (3GPP) specified the model of a virtual 5G-TSN bridge. This contains TSN translators which map principles such as time synchronization and Quality of Service (QoS) mechanisms from TSN to 5G. However, practical implementations are not available yet. This paper clarifies the differences of TSN and 5G in the prioritization of data traffic and provides possible solutions to map the priorities to each other. This serves as a basis for the development of TSN translators and finally of a joint QoS model.

**Keywords:** 5G, Time-Sensitive Networking, Quality of Service, Prioritization

## 1 Introduction

Flexible production processes for Industry 4.0 require effortless reconfigurability and mobility and therefore a combination of wired and wireless real-time capable communication technologies. In particular, this combination is essential to time-critical machine communication. Possible use cases include wireless human-machine interfaces (HMIs) with emergency stop or Automated Guided Vehicles (AGVs) in order to guarantee personal safety in mobile applications. The combination of Time-Sensitive Networking (TSN) and 5G mobile radio is considered suitable candidates to meet the communication requirements of these use cases.

TSN is the umbrella term for several IEEE 802.1 sub-standards that enable real-time capabilities and determinism for Ethernet. TSN includes mechanism for time synchronization, bounded latency, high reliability, and dedicated resource management [IE22]. The IEC/IEEE 60802 TSN Industrial Automation Profile intends to explicitly standardize the use of TSN in industrial automation, but is currently still in the draft stage [IE21].

For the first time in mobile radio communication, 5G as its fifth generation promises performance that natively addresses industrial requirements. For example, the Ultra-Reliable Low Latency Communication (URLLC) feature supports time-critical communication.

In order to achieve a seamless integration of 5G into TSN, 3GPP TS 23.501 specifies the 5G system as a virtual TSN bridge [3G21a]. Fig. 1 shows the corresponding model in

[1] Volkswagen AG, Berliner Ring 2, 38440 Wolfsburg, niklas.ambrosy@volkswagen.de
[2] Institut für Automation und Kommunikation e. V., Werner-Heisenberg-Str. 1, 39106 Magdeburg, lisa.underberg@ifak.eu

which the 5G system is a black box and appears as a TSN bridge within the TSN network. At the system boundaries, i.e., ingress and egress ports, the Device-Side TSN Translator (DS-TT), Network-Side TSN Translator (NW-TT), and TSN Application Function (TSN AF) translation functions are necessary. These entities provide the integration of QoS frameworks, forwarding and topology information, with industrial network management, and with the synchronization framework [RCK20]. The aspect of time synchronization is already part of the standardization since 3GPP Release 16 (cf. older versions of [3G21a]).

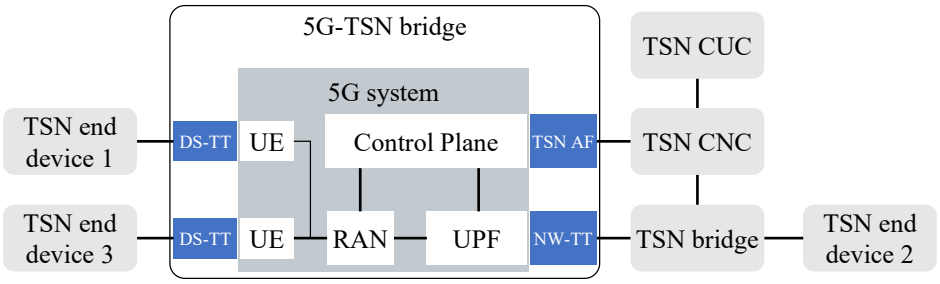



Fig. 1: 5G system as a virtual TSN bridge [3G21a]

Research is currently focusing much more on time synchronization [SS21, Pa21], which forms the basis for several traffic shaping mechanisms. A QoS mapping algorithm for 5G-TSN already exists, but there is no reference to the TSN Industrial Automation profile and the focus is on PER rather than priority [Sa22a]. However, it is equally relevant to achieve a common understanding of priorities of different traffic types in both systems, especially when time-critical traffic of mobile safety applications requires prioritization.

Therefore, this paper focuses on traffic prioritization, which is fundamentally different in both technologies and not explicitly standardized [3G21a]. The paper explains prioritization in TSN and 5G, maps the priorities of 5G QoS flows to TSN streams based on the respective parameters, and finally discusses this mapping in terms of a joint QoS model.

The remainder of this paper is organized as follows: Section 2 explains traffic prioritization in TSN and Section 3 for 5G accordingly. Section 4 maps the respective priorities to lay the foundation for a joint 5G-TSN QoS model, distinguishing between the use of standardized and non-standardized 5G Quality Indicator (5QI) values. Section 5 concludes the paper with a summary and an outlook.

## 2 Traffic prioritization in TSN

This section describes TSN mechanisms and parameters for the traffic prioritization as well as industrial protocols enabling real-time communication with TSN.

In accordance with IEEE 802.1Q - Strict Priority, the traffic types in industrial communication are assigned to a total of eight traffic classes [In19], as listed in Tab. 1. A traffic class is identified using the Priority Code Point (PCP) as part of the Virtual Local Area Network (VLAN) tag. Priorities range from 0 to 7, where 7 represents the highest and 0 the lowest priority.

| Traffic Class & Priority | Traffic Type | Periodicity | Data Delivery Guarantee | Data size: Fixed/ Variable | Criticality |
|---|---|---|---|---|---|
| 7 | Network control | Periodic 50-1000 ms | Bandwidth/ Data rate | V: 50-500 Bytes | High |
| 6 | Isochronous | Periodic 0.1-2 ms | Deadline | F: 30-100 Bytes | High |
| 5 | Cyclic synchronous or asynchronous | Periodic 0.5-20 ms | Latency | F: 50-1000 Bytes | High |
| 4 | Events (control) | Sporadic 10-50 ms | Latency | V: 100-200 Bytes | High |
| 3 | Alarms (operator commands) | Sporadic 2000 ms | Latency | V: 100-1500 Bytes | Medium |
| 2 | Configuration & diagnostics | Sporadic N/A | Bandwidth/ Data rate | V: 500-1500 Bytes | Medium |
| 1 | Audio/Video | Periodic A: 40 ms V: 10 ms | Bandwidth/ Data rate, Latency | V: 1000-1500 Bytes | Low |
| 0 | Best effort | Sporadic N/A | None | V: 30-1500 Bytes | Low |

Tab. 1: Traffic types, classes and priorities [In19]

Prioritization based on traffic classes, along with network-wide time synchronization according to IEEE 802.1AS, forms the basis for further TSN traffic shaping mechanisms. These include the time-aware shaper for exclusive gating (IEEE 802.1Qbv), credit-based shaping (IEEE 802.1Qav), frame preemption (IEEE 802.1Qbu), frame replication (IEEE 802.1CB), ingress policing (IEEE 802.1Qci), cut-through switching and reservation/scheduling [IE22]. Although the time-aware shaper and frame preemption are able to override simple prioritization depending on their configuration, both mechanisms should be seen as additional optimization options beyond prioritization.

In automation, Industrial Ethernet-based protocols, such as PROFINET, play a significant role. For instance, in PROFINET, TSN can replace the two proprietary real-time extensions on Layer 2, PROFINET IRT (isochronous real-time, cf. traffic class 6) and PROFINET RT (real-time, cf. class 5) [SJ21].

OPC UA is gradually finding its way into automation and can be used for controller-to-controller (C2C) and controller-to-device (C2D) communication. In pure C2C communication, classes 0-4 are covered. If C2D, i.e. field level communication, is also considered via OPC UA TSN, all traffic classes can be affected [Br19].

# 3 Traffic prioritization in 5G

This section briefly explains the 5G QoS framework, including the relevant parameters for traffic prioritization.

3GPP TS 23.501 specifies the 5G QoS model. The 5G system establishes a Protocol Data Unit (PDU) session between the User Plane Function (UPF) and the User Equipment (UE). A PDU session can contain one or more QoS flows including the respective QoS Flow ID (QFI). Each QoS flow in turn has a specific QoS profile, which is identified by the 5G QoS Identifier (5QI). The QoS profile comprises several QoS parameters [5G21b], such as Resource Type, Priority Level, Packet Delay Budget (PDB), Packet Error Rate (PER), Maximum Data Burst Volume (MDBV) and Averaging Window. The parameters are explained below and listed with concrete values in Tab. 2 [3G21a].

- *Resource Type:* This parameter indicates how the Packet Delay Budget, Packet Error Rate, and Maximum Data Burst Volume should be handled. The resource can be of type Guaranteed Bit Rate (GBR), Non-GBR, or Delay-Critical GBR. The required bit rates are permanently allocated.
- *Priority Level:* indicates a flow's priority in relation to other flows for scheduling resources. Unlike TSN, the lowest priority level value corresponds to the highest priority.
- *Packet Delay Budget (PDB):* sets an upper time limit for the delay between the UE and the UPF, before the packet is counted as lost. Packet fragmentation can affect the PDB and limit the packet size.
- *Packet Error Rate (PER):* defines the level of reliability by providing an upper bound on the number of incorrectly received and lost packets divided by the total number of received packets. The larger the packet and the lower the PDB, the higher the PER.
- *Maximum Data Burst Volume (MDBV):* indicates the amount of data that can be sent without exceeding the PDB.
- *Averaging Window:* This parameter refers only to GBR resources and indicates the calculation time of Guaranteed Flow Bit Rate (GFBR) and Maximum Flow Bit Rate (MFBR) for a given traffic flow. While GFBR can be expected for a QoS flow over the averaging window, MFBR defines the maximum value of an actual bitrate.
- *Allocation and Retention Priority (ARP):* indicates a QoS flow's relative priority with a value between one (highest priority) and 15 in combination with 5QI. Depending

on ARP and the specific implementation, the 5G system decides how a QoS flow should be served or preempted when resources are limited [5G21b].

| 5QI Value | Resource Type | Default Priority Level | Packet Delay Budget | Packet Error Rate | Default Maximum Data Burst Volume | Default Averaging Window |
|---|---|---|---|---|---|---|
| 1 | GBR | 20 | 100 ms | $10^{-2}$ | N/A | 2000 ms |
| 2 | | 40 | 150 ms | $10^{-3}$ | N/A | 2000 ms |
| 3 | | 30 | 50 ms | $10^{-3}$ | N/A | 2000 ms |
| 4 | | 50 | 300 ms | $10^{-6}$ | N/A | 2000 ms |
| 65 | | 7 | 75 ms | $10^{-2}$ | N/A | 2000 ms |
| 66 | | 20 | 100 ms | $10^{-2}$ | N/A | 2000 ms |
| 67 | | 15 | 100 ms | $10^{-3}$ | N/A | 2000 ms |
| 75 | | reserved for future use | | | | |
| 71 | | 56 | 150 ms | $10^{-6}$ | N/A | 2000 ms |
| 72 | | 56 | 300 ms | $10^{-4}$ | N/A | 2000 ms |
| 73 | | 56 | 300 ms | $10^{-8}$ | N/A | 2000 ms |
| 74 | | 56 | 500 ms | $10^{-8}$ | N/A | 2000 ms |
| 76 | | 56 | 500 ms | $10^{-4}$ | N/A | 2000 ms |
| 5 | Non-GBR | 10 | 100 ms | $10^{-6}$ | N/A | N/A |
| 6 | | 60 | 300 ms | $10^{-6}$ | N/A | N/A |
| 7 | | 70 | 100 ms | $10^{-3}$ | N/A | N/A |
| 8 | | 80 | 300 ms | $10^{-6}$ | N/A | N/A |
| 9 | | 90 | 300 ms | $10^{-6}$ | N/A | N/A |
| 69 | | 5 | 60 ms | $10^{-6}$ | N/A | N/A |
| 70 | | 55 | 200 ms | $10^{-6}$ | N/A | N/A |
| 79 | | 65 | 50 ms | $10^{-2}$ | N/A | N/A |
| 80 | | 68 | 10 ms | $10^{-6}$ | N/A | N/A |
| 10 | | 90 | 1100 ms | $10^{-6}$ | N/A | N/A |
| 82 | Delay-critical GBR | 19 | 10 ms | $10^{-4}$ | 255 Bytes | 2000 ms |
| 83 | | 22 | 10 ms | $10^{-4}$ | 1354 Bytes | 2000 ms |
| 84 | | 24 | 30 ms | $10^{-5}$ | 1354 Bytes | 2000 ms |
| 85 | | 21 | 5 ms | $10^{-5}$ | 255 Bytes | 2000 ms |
| 86 | | 18 | 5 ms | $10^{-4}$ | 1354 Bytes | 2000 ms |
| 87 | | 25 | 5 ms | $10^{-3}$ | 500 Bytes | 2000 ms |
| 88 | | 25 | 10 ms | $10^{-3}$ | 1125 Bytes | 2000 ms |
| 89 | | 25 | 15 ms | $10^{-4}$ | 17000 Bytes | 2000 ms |
| 90 | | 25 | 20 ms | $10^{-4}$ | 63000 Bytes | 2000 ms |

Tab. 2: QoS profiles with standardized 5QIs from 3GPP TS 23.501 5.7.4-1 [3G21a]

# 4 Priority Mapping

This section presents an approach to a generic translation process for the TSN translator functions within the 5G-TSN bridge and systematically derives explicit mapping solutions for standardized and non-standardized 5G QoS parameters. Both mapping solutions are critically discussed.

A 5G system can receive QoS information for the TSN traffic from the centralized network configuration (CNC) via TSN AF. The QoS mapping table preconfigured by the TSN AF is used to identify a suitable 5G QoS profile. Based on the information received, the 5G system selects an appropriate QoS profile for each TSN stream in order to establish a corresponding 5G QoS flow for delivering TSN traffic between the ingress and egress ports of the 5G bridge. In addition, it is possible to use packet filters on the UE and UPF side, which can be used to map different TSN streams to corresponding 5G QoS flows [5G21b].

Different TSN traffic classes need to be prioritized accordingly within the 5G system. IEEE 802.1Q – Strict Priority takes precedence as IEEE 802.1Qbv can open multiple gates simultaneously and then prioritize again according to the PCP. The 3GPP specifications mention QoS mapping tables without making any specific statement about their content [3G21a, 3G21b]. Even 5G-ACIA (Alliance for Connected Industries and Automation), which focuses on industrial automation, does not derive any 5G QoS mapping table for TSN in detail [5G21a].

A meaningful mapping requires that the 5G system understands the TSN parameters, which can be translated as follows:

- TSN frame size ≙ GFBR
- TSN frame size ≙ MDBV
- periodicity ≙ averaging window [Ma21].

Three of the eight TSN traffic classes are real-time streams (4-6). These correspond to the 5G delay-critical GBR category [Ma21]. Automation is given as an exemplary service for 5QI values 82 and 83, but both have a packet delay budget of 10 ms, which likely exceeds the time requirements of isochronous applications. Ethernet frames can reach a length of up to 1522 Bytes including the VLAN tag, which exceeds the MDBV of 1354 Bytes maximum. However, the TSN Industrial Automation Profile limits the real-time streams to a maximum of 1000 Bytes.

In both TSN and 5G, network control requires guaranteed bandwidth with the highest priority. On the other hand, network control, configuration & diagnostics, and best effort are not time-critical. Best effort can even be assigned to a 5QI from the non-GBR category.

Since TSN can be used in combination with different possible deployment scenarios for non-public 5G networks [5G19], two solutions are discussed in the following subsections:

public network integrated non-public networks (PNI-NPNs) are expected to rely on using exclusively or primarily standardized 5QI values in coordination with the mobile network operator (MNO), while the operator of a standalone non-public network (SNPN) can freely define additional 5G QoS parameters. The standardized 5QIs are mandatory in both public and non-public networks. Thus, they have the role of a common basis, e.g., to ensure interoperability between different networks and operator models. If required, the operator can define additional 5QIs.

### 4.1 Using standardized 5QI

Assuming that only standardized 5QI values are used, Tab. 3 maps the respective priorities to each other. Due to the 5G time-division duplex (TDD) pattern, uplink traffic usually provides less capacity and is more complex to schedule than downlink traffic. Consequently, isochronous uplink streams should be prioritized higher than the corresponding downlink streams, to ensure bidirectional reliability. The alternative values in parentheses from Tab. 3 can be used for this purpose.

Another reason for setting 5QI fallback values is to allow the service to adapt to current performance and conditions, as wireless networks are inherently less reliable than wired ones. Moreover, a fallback solution makes sense since QoS-adaptive applications support the flexibility in modern factories [5G21b].

| Universal Priority | TSN Traffic Class & Priority | Mapped 5QI Value | 5G Priority Level |
|---|---|---|---|
| 1 | 7 | 65 (67) | 7 (15) |
| 2 | 6 | 86 (85) | 18 (21) |
| 3 | 5 | 82 (83) | 19 (22) |
| 4 | 4 | 84 (88) | 24 (25) |
| 5 | 3 | 3 (2) | 30 (40) |
| 6 | 2 | 4 (71) | 50 (56) |
| 7 | 1 | 73 (72) | 56 (56) |
| 8 | 0 | 80 (7, 8, 9) | 68 (70, 80, 90) |

Tab. 3: Proposed 5G-TSN QoS mapping based on standardized 5QI

As mentioned earlier, the priority values in TSN are opposite to 5G and the universal priority understanding. Thus, the values in Tab. 3 can be derived as follows:

1. Real-time TSN streams with strict latency requirements (i.e. traffic classes 4, 5 and 6) need to be mapped to a 5QI with Delay-critical GBR (5QI ≥ 82).
2. Data rate-intense TSN traffic classes correspond to GBR 5QI.
3. The TSN frame length needs to match the MDBV [Ma21].

4. The TSN periodicity needs to match the averaging window [Ma21]. Since only default values are specified (2000 ms), these can be adjusted accordingly.
5. At the same time, the order of the 5G Priority Level must be observed in accordance with the TSN priority.

Hence, the mapping for standardized 5QI values is performed starting from the 5G domain. The user must inevitably accept performance compromises here:

- There are inconsistencies in several PDB values (e.g. 5 ms for isochronous traffic although 2 ms are required and 300 ms for audio/video against 40 ms).
- The PER of network control traffic is relatively high ($10^{-2}$).
- It is an open question whether a lower limit exists for the averaging window. The default value of 2000 ms is too high for industrial automation.
- The 3GPP table only specifies example services. Therefore, it is unclear whether in a PNI-NPN the end user is allowed to utilize all 5QIs for TSN traffic or whether this is already firmly reserved for other traffic types by the MNO. This question may have a significant impact on the suitability of the mapping and thus represents a risk.

Another unresolved compromise would be to use only 5QI values 82 or 83, which are dedicated to industrial automation, and implement relative prioritization based on ARP. Consequently, all TSN traffic within the 5G system would have the same 5QI value, but with relative prioritization via ARP values 1 (for network control) to 8 (best effort).

### 4.2 Using non-standardized 5QI

In contrast to the restrictions in a 5G network offering only the standardized 5QIs, the operator of an SNPN can flexibly set 5QIs for its specific purposes. In factories, the most likely scenario is an overarching 5G network for the entire plant site and several separate TSN networks per production cell, group of production cells, or, at its largest, an entire production hall. However, this requires a more differentiated prioritization in the 5G network than just eight traffic classes or priority levels or ARPs, since it does not exclusively serve TSN data traffic. Furthermore, depending on the scalability or dimensioning of the network, there must be an upper limit at which the capacity of the 5G network is no longer able to handle the over-demanding QoS parameters. This reveals another research gap, as there has been no work on this to date in real-world networks.

Tab. 4 lists the proposed customized 5G QoS parameters based on the traffic parameters in IEC/IEEE 60802 (as shown in Tab. 1). The derivation corresponds methodically to that in Section 4.1, but starts from the TSN perspective. All values are self-defined based on the characteristics of the TSN traffic types and are not covered by standardized 5QIs. For instance, 5G promises a latency of 1 ms in the long term, so a PDB of 2 ms for isochronous traffic seems realistic. The PER is based on the assumption of a long-term reliability for URLLC traffic of 99.99999 %. The following values differ from the standardized ones:

Isochronous traffic cannot be covered by the existing standardized 5QIs. The low values for the averaging window parameter are questionable for classes 7, 5 and 4, and the MDBV value for classes 3-0. These are specified in 3GPP by default as 2000 ms and "not applicable", respectively.

| TSN Priority | 5QI Value | Resource Type | 5G Priority Level | PDB | PER | MDBV [Bytes] | Averaging Window |
|---|---|---|---|---|---|---|---|
| 7 | 11 | GBR | 1 | 1 s | $< 10^{-5}$ | 50-500 | 50-1000 ms |
| 6 | 12 | DC-GBR | 2 | 2 ms | $< 10^{-5}$ | 30-100 | < 2 ms |
| 5 | 13 | DC-GBR | 3 | 20 ms | $< 10^{-5}$ | 50-1000 | 2-20 ms |
| 4 | 14 | DC-GBR | 4 | 50 ms | $< 10^{-5}$ | 100-200 | 10-50 ms |
| 3 | 15 | GBR | 5 | 2000 ms | $< 10^{-4}$ | 100-1500 | 2000 ms |
| 2 | 16 | GBR | 6 | 1000 ms | $< 10^{-4}$ | 500-1500 | N/A |
| 1 | 17 | GBR | 7 | 10 ms | $< 10^{-4}$ | 1000-1500 | N/A |
| 0 | 18 | (Non-) GBR | 8 | none | $< 10^{-3}$ | 30-1500 | N/A |

Tab. 4: Proposed 5G-TSN QoS mapping based on non-standardized 5QI

In contrast to the previous section, the 5G QoS parameters are selected with respect to the TSN parameters. Since one of the goals of TSN is to guarantee sufficient bandwidth to all traffic types, it seems reasonable to define best effort as GBR as well. The prerequisite for the mapping presented here is appropriate dimensioning and design of the 5G campus network in order to be able to provide the required performance.

Validation of the QoS mapping concept was performed as a simulation in OMNeT++ [MP21, Sa22b]. Their QoS mapping table is based on standardized 5QIs of the DC-GBR category. The simulation takes two different PCP values as an example without addressing each QoS parameter of the TSN traffic classes. Thus, our paper represents a necessary preliminary work and focuses on the systematic derivation of holistically matching QoS parameters, so that all priorities at the 5G-TSN system boundaries are preserved.

# 5 Conclusion

In this paper, we investigated traffic prioritization in TSN and 5G. We analyzed the QoS parameters of the TSN traffic classes in accordance with the TSN Industrial Automation

Profile and explained the 5G QoS framework. Based on that, we systematically derived the mapping of priorities in TSN and 5G with standardized and non-standardized 5QIs.

Our results show that the standardized 5QIs are only partially suitable for the stringent TSN QoS requirements and motivate the definition of new 5QIs. Note that the presented mapping tables represent only two of several other suitable solutions. Nevertheless, we propose to include Tab. 4 in 3GPP TS 23.501 to provide end users with a consistent solution.

While this paper focused on the aspect of traffic prioritization, the QoS mechanisms in TSN and 5G are much more multi-layered. A solution for a joint QoS model is not trivial and requires further research in a holistic approach, including time synchronization, prioritization, scheduling and other QoS mechanisms. In addition, it should be investigated how the network determines to drop flows in case not all high-priority applications can be served.

Future work includes further simulations, e.g. as an extension to [Ma20, Sa22b], as well as experiments with PROFINET and OPC UA over 5G and TSN in order to validate the practicality in real factory implementations.

## 6 References


[3G21a] 3GPP: TS 23.501: System architecture for the 5G System (V17.3.0), 2021.

[3G21b] 3GPP: TS 23.503: Policy and charging control framework for the 5G System (V17.3.0), 2021.

[5G19] 5G-ACIA: 5G Non-Public Networks for Industrial Scenarios, 2019.

[5G21a] 5G-ACIA: Integration of 5G with Time-Sensitive Networking for Industrial Communications, 2021.

[5G21b] 5G-ACIA: 5G QoS for Industrial Automation, 2021.

[Br19] Bruckner, D. et al.: An Introduction to OPC UA TSN for Industrial Communication Systems. Proceedings of the IEEE 6/107, pp. 1121–1131, 2019.

[IE21] IEC/IEEE 60802: TSN Profile for Industrial Automation, 2021.

[IE22] IEEE: Time-Sensitive Networking (TSN) Task Group. 1.ieee802.org/tsn, accessed 29 Mar 2022.

[In19] Industrial Internet Consortium: Time Sensitive Networks for Flexible Manufacturing Testbed - Characterization and Mapping of Converged Traffic Types, 2019.

[Ma20] Martenvormfelde, L. et al.: A Simulation Model for Integrating 5G into Time Sensitive Networking as a Transparent Bridge: 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 1103–1106, 2020.

[Ma21] Martenvormfelde, L. et al.: Co-configuration of 5G and TSN enabling end-to-end quality of service in industrial communications: Kommunikation in der Automation (KommA 2021) 12. Jahreskolloquium, 18.11.2021 in Verbindung mit dem Industrial Radio Day, 17.11.2021 Tagungsband, Magdeburg, 2021.

[MP21] Magnusson, A.; Pantzar, D.: Integrating 5G Components into a TSN Discrete Event Simulation Framework. Master thesis, Västerås, Sweden, 2021.

[Pa21] Patel, D. et al.: Time error analysis of 5G time synchronization solutions for time aware industrial networks: 2021 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS). IEEE, pp. 1–6, 2021.

[RCK20] Rost, P. M.; Chandramouli, D.; Kolding, T.: 5G plug-and-produce - How the 3GPP 5G System facilitates Industrial Ethernet, 2020.

[Sa22a] Satka, Z. et al.: QoS-MAN: A Novel QoS Mapping Algorithm for TSN-5G Flows.

[Sa22b] Satka, Z. et al.: Developing a Translation Technique for Converged TSN-5G Communication: 2022 IEEE 18th International Conference on Factory Communication Systems (WFCS). IEEE, pp. 1–8, 2022.

[SJ21] Schriegel, S.; Jasperneite, J.: A Migration Strategy for Profinet Toward Ethernet TSN-Based Field-Level Communication: An Approach to Accelerate the Adoption of Converged IT/OT Communication. IEEE Industrial Electronics Magazine 4/15, pp. 43–53, 2021.

[SS21] Striffler, T.; Schotten, H. D.: The 5G Transparent Clock: Synchronization Errors in Integrated 5G-TSN Industrial Networks: 2021 IEEE 19th International Conference on Industrial Informatics (INDIN). IEEE, pp. 1–6, 2021.

**Disclaimer:**

The results, opinions and conclusions expressed in this publication are not necessarily those of Volkswagen AG.

# Modelbasierte Fehlerursachenanalyse in zeitsensitiven Ethernet-Netzwerken

Tobias Ferfers[1], Alexander Biendarra[1], Sebastian Schriegel[1] und Juergen Jasperneite[1]

**Abstract:** Fehlerursachenanalysen in vernetzten Systemen sind häufig komplex und aufwendig. Zeitsensitive Ethernet-Systeme wie Time Sensitive Networks nutzen Mechanismen wie Zeitsynchronisation, Frame-Preemption und Time Aware Shaper, die in einem komplexen Zusammenspiel für Echtzeitfähigkeit sorgen. In diesem Bereich sind Fehlerursachenanalysen besonders komplex, da spezielles Expertenwissen und spezielle Messtechnik notwendig sind. Modelbasierte Fehlerursachenanalysen können hier in Zukunft als automatisches Diagnoseassistenzsystem Unterstützung leisten. Das Ziel dieser Arbeit ist die Ermittlung von Fehlermodellen, die einen Zusammenhang zwischen messbaren Symptomen und möglichen Fehlerursachen in zeitsensitiven Systemen herstellen. Die entwickelten Fehlermodelle wurden in einem prototypischen Ethernet TSN-Netzwerk mit Messtechnik validiert.

**Keywords:** Fehlerursachenanalyse, Root Cause Analysis, Real-Time-Communication, Ethernet TSN

## 1 Einleitung und Motivation

Die Erschließung neuer Anwendungsfälle, wie z.B. datengetriebene Services erzeugt neue Anforderungen an die industrielle Netzwerktechnik [W17]. [FSJ21] zeigt eine Architektur eines Industrial Internet of Things (IIoT), welches eine Vernetzung von der Sensorik bis in die Cloud vorsieht und die zunehmende Verschmelzung von IT und OT (Operational Technology, Feldebene) zeigt. Ebendies führt dazu, dass nicht nur die Netzwerkstrukturen zunehmend komplexer werden, sondern auch Einzelkomponenten und die Konfiguration. Neue Vernetzungstechnologien wie z.B. 5G, Ethernet Time Sensitive Networking (TSN) mit verschiedenen Bitraten, Single Pair Ethernet und WiFi werden mit bestehenden Technologien kombiniert und zu komplexen Gesamtsystemen zusammengeführt. Diese gesteigerte Komplexität und Systemheterogenität erschweren das Design, den Aufbau und die Fehlersuche.

Falls während einer Inbetriebnahme oder dem Betrieb eines Kommunikationssystems ein Fehler auftritt, dann wird auf Diagnoselösungen zur Fehlersuche zurückgegriffen. Eine Fehlerursachenanalyse ist häufig aufgrund von komplexen zeitlichen Zusammenhängen und Abhängigkeiten zwischen Protokollen, Hardware, Software und Gerätekonfiguration aufwendig. Zeitsensitive Kommunikationslösungen, wie zum Beispiel Ethernet TSN

[1]Fraunhofer IOSB-INA, Campusallee 1, Lemgo, Germany {tobias.ferfers, alexander.biendarra, sebastian.schriegel, juergen.jasperneite} @iosb-ina.fraunhofer.de

[TSN22], mit denen eine Koexistenz von Echtzeit-Kommunikation und Nicht-Echtzeitkommunikation realisiert wird, stellen eine noch größere Herausforderung dar. Ein heute übliches Hilfsmittel in der Fehlerdiagnose ist unter anderem das Protokollanalysetool Wireshark [Wi22]. Wireshark ermöglicht Live-Mitschnitte und Offlineanalyse von Netzwerkprotokollen. Die Analyse dieser Mitschnitte erlaubt nur eine begrenzte Bewertung der zeitlichen Zusammenhänge des Netzwerkverkehrs, da u. U. von tausenden Frames Übertragungszeiten händisch geprüft werden müssen. Für die bessere Analyse zeitlicher Zusammenhänge können z.B. Oszilloskop-Lösungen oder spezielle Ethernet-Messtechnik eingesetzt werden. Diese Messtechnik ist in realen Fabrikumgebungen häufig nicht praktikabel, da Schnittstellen oder Geräte in einer Anlage oft nur schwer zugänglich sind. Zudem muss die Auswertung der Messergebnisse von Experten häufig manuell erfolgen.

Bei der modellbasierten Fehlerdiagnose wird parallel zum physikalischen System (hier das Netzwerk) ein digitales Modell genutzt und mit den Messwerten verglichen. Ein Beispiel der modellbasierten Diagnose in der industriellen Kommunikation ist der automatische Topologievergleich in PROFINET Class B und Class C-Netzwerken. Der vorgegebene Sollwert (die Netzwerktopologie) wird kontinuierlich überprüft und bei Abweichung erfolgt ein Diagnosealarm [PN21]. Falls beispielsweise eine fehlende Komponente in der Topologie erkannt wird, ist die Ursache, warum diese Komponente nicht mehr in der Topologie vorhanden ist, in der Regel nicht zu ermitteln.

In diesem Paper wird eine modellbasierte Fehlerursachenanalyse für Ethernet TSN-Netzwerke und Funktionen vorgeschlagen und untersucht. Neben einer Beschreibung des Gesamtkonzeptes (Kapitle 3) werden Fehlerursachen beschrieben und in einer auf der Web Ontology Language (OWL) basierten Notation modelliert (Kapitel 4) und mit Hilfe eines realen Ethernet TSN-Netzwerkes validiert (Kapitel 5).

## 2 Stand der Technik

In den 80er Jahren begannen Arbeiten an Begriffsdefinitionen im Bereich von Fehlern, Fehlerdiagnose, aber auch Fehlermanagement. Daraus entstanden vielfältige Definition, die in unterschiedlichen internationalen Normen festgehalten wurden. [IB97] gibt eine Übersicht über die wichtigsten Begriffsdefinition auf Basis eben dieser internationalen Normen, die auch in dieser Arbeit verwendet werden. Der Begriff *Fehler* bezieht sich auf eine unerlaubte Abweichung von mindestens einer charakteristischen Eigenschaft oder Parameter eines Systems vom akzeptablen, normalen oder üblichen Zustand. *Fehlerdiagnose* besteht aus der Fehlererkennung, Fehlerisolation und der Fehleridentifizierung. *Fehlererkennung* beschreibt die Möglichkeit (eines Systems) überhaupt zu erkennen, dass ein Fehler vorliegt und wann dieser aufgetreten ist. *Fehlerisolation* beschreibt Art, Ort und Zeitpunkt der Erkennung. *Fehleridentifikation* beschreibt die Auswirkung / Größe des Fehlers und das zeitvariante Verhalten des Fehlers.

Nach einer erfolgreichen Fehlerdiagnose ist der Ort und die schwere des Fehlers bekannt, die Ursache und Maßnahmen zur Behebung des Fehlers sind noch zu bestimmen. Zur Behebung des Fehlers ist Expertenwissen notwendig. Die Fehlerursachenanalyse widmet sich genau diesem Problem und ordnet auftretenden Störungen oder Fehlern eine mögliche Ursache zu. [APL81] zeigt beispielsweise einen Ansatz der Fehlerursachenanalyse für Kraftwerkkomponenten und analysiert dabei mögliche Ursachen in einer Vielzahl von Studien für Turbinengenerator, Pulverisator und Kessel. Heutzutage gibt es IT-Protokolle wie zum Beispiel syslog [Ge09], die dem Austausch von Log-Nachrichten in Netzwerken dienen und grundsätzlich dazu verwendet werden, Log-Nachrichten in einem zentralen System abzulegen. [An18] nutzt syslog für eine Fehlerursachenanalyse und Reparatursystem in IP-basierten Netzwerke. Die syslog Nachrichten werden an eine weitere zentrale Auswerteeinheit weitergeleitet und analysiert. Für zwei Fehlerursachen-Kategorien (Performance und Verbindungsursachen) wurden Fehlerbäume erstellt in deren mögliche Ursachen festgehalten werden. In einen Feldversuch wurden vier mögliche Fehler analysiert: Zwei Verbindungsprobleme (doppelt vergebene IP und link up/ down) sowie zwei Performanceprobleme (Loopback und MAC flapping). In einem Testversuch mit einem erfahrenen Netzwerkadministrator wurde die manuelle und unterstützte Fehlersuche mit zufällig ausgewählten Problemen innerhalb und außerhalb der Bürozeiten durchgeführt. Bei diesem empirischen Test wurde gezeigt, dass das automatische Lösungssystem über 90% schneller die Probleme finden und beheben konnte. Im Bereich der industriellen Kommunikation gibt es bereits Protokolle, die eine umfangreiche Diagnose ermöglichen, wie beispielsweise PROFINET [Pn20]. Bei PROFINET erfolgt die Diagnose auf Komponentenebene. Jedes (Sub-)Modul einer Komponente tauscht Diagnoseinformationen untereinander aus und speichert diese in einer zentralen Datenbank (Diagnose Application Service Element). Mit Hilfe von Alarmen werden detaillierte Diagnoseinformationen mit anderen Komponenten ausgetauscht.

Auch im Bereich der industriellen Kommunikation gibt es Lösungen, die zusätzliche Messtechnik einsetzen, um ein Netzwerk genauer diagnostizieren zu können. Der PROFINET-INspektor® NT der Firma Indusol ist ein Diagnosegerät zur Validierung, Abnahme und Prüfung von neu installierten PROFINET-Netzwerken. Das Gerät überwacht dabei beispielsweise Ausfälle, Anzahl Neuanläufe, Aktualisierungsraten, Jitter der Kommunikation, Telegrammlücken oder die Netzlast / Datendurchsatz [In22]. Der EC-Monitor der Firma acontis ist eine Softwarebibliothek, die die Aufzeichnung, Analyse und Evaluierung des Datenverkehrs ermöglicht [Ac22]. Auch in diesem Ansatz muss ein zusätzlicher Test Access Point in dem Netzwerk untergebracht werden. Die Bibliothek ermöglicht das mitschneiden und analysieren des Datenverkehrs und kann so in unterschiedlichen Use-Cases eingesetzt werden. Ein System zur Diagnose muss vom Benutzer erstellt werden.

Fehlerursachenanalyse findet man häufig im Bereich von (industriellen) Prozessen oder im Qualitätsmanagement z.B. beim automatischen Löten von Platinen [BYD13]. Das Problem dabei ist, dass die Symptome und vor allem die Ursachen manuell erstellt und festgehalten werden z.B. durch Fehlerbäume, Fischgrätenmodelle oder Ursache-

Wirkungs-Diagramm. Zum Ermitteln von möglichen, Ursachen sind diese Tools geeignet, die Überführung in automatische, computergestützte Fehlerursachenanalyse Systeme gestaltet sich da meist aufwendiger. Ein weiterer wachsender Bereich ist die Fehlerursachenanalyse mit Hilfe von Künstlicher Intelligenz, häufig Machine Learning. [TSTM21] beschreibt eine Fehlerursachenanalyse für Software defined Networks. Prinzipiell wird dabei Netzwerkverkehr aufgenommen und die für die Künstliche Intelligenz benötigten Features extrahiert. Die Features sind zum Beispiel Latenz, Paketverluste, gesendete Pakete, empfangene Pakete oder Anzahl an Bytes werden in der Fehlerursachenanalyse. Erkennt das Fehlerursachenanalyse-Modul eine Veränderung oder Abweichung in den Features benachrichtigt es den Netzwerkadministrator über einen Alarm. Die vier betrachteten Fehlerfälle Buffer overload, Link-Fehler und Switch-Fehler verwenden vordefinierte Schwellwerte für einen normalen Betrieb z.B. Buffer overload definierte Bandbreite und Senderate. Nachteil bei diesem System ist, dass nur vordefinierte Anomalien erkannt werden können. [JC22] nutzt einen ähnlichen KI basierten Ansatz für die Fehlerursachenanalyse in industriellen Netzwerken. [BDO16] verfolgt einen anderen Ansatz und modelliert das System, aber auch die einzelnen Geräte sowie verschiedene Arten von Anomalien. Durch das Kategorisieren von Anomalien in kontinuierliche, logische und zeitliche Anomalien und der Abbildung dieser in Modellen wird ermöglicht eine Vielzahl gleicher Anomaliearten zu erkennen. Modelliert wird das System, die Instanzen, die Anomalien, Algorithmen und das Wissen mit der Web Ontology Language (OWL) [He22]. OWL stammt aus dem semantischen Web. Im Semantischen Web wird die Bedeutung der Informationen maschinenlesbar modelliert, damit es einfacher ist für Maschinen automatisch Informationen zu verarbeiten und zu integrieren. Ontologien definieren Begriffe die genutzt werden um Wissen zu beschreiben und zu repräsentieren. Das Ziel dieser Modellierung war ein System, mit dem Menschen und Maschinen mit natürlicher Sprache interagieren können z.B. „Wie ist der Sensorwert xy?“, „Gibt es Anomalien im System?“ und das System mit Hilfe des Modells diese Fragen beantworten kann und beim Betrieb der Anlage unterstützt.

# 3 Lösungsansatz einer modellbasierte Fehlerursachenanalyse für zeitsensitive Kommunikation

In diesem Kapitel wird ein Konzept für eine modellbasierte Fehlerursachenanalyse in zeitsensitiven Systemen vorgestellt. Ziel einer modellbasierten Fehlerursachenanalyse ist ein Assistenzsystem, dass Serviceingenieure bei der Fehlerursachenanalyse von zeitsensitiven Systemen unterstützt. Symptome von Fehlern können sehr unterschiedlich sein und z.B. von in die Systeme integrierter Software oder Hardware festgestellt (Jitter, Sync-Verlust, Telegrammausfälle, Topologieabweichung) und teilweise auch quantitativ gemessen werden. Bei Systemen, die die spezifischen Symptome nicht selbst feststellen können oder bei denen für die Analyse von Fehlern weitere Informationen notwendig sind, ist zusätzliche Messtechnik zu installieren. Eine direkte Erkennung der Ursache aus den gemessenen Daten und den Symptomen ist häufig nicht möglich. Insbesondere Anwender haben es schwer die Ursache und damit die Behebungsmaßnahme zu finden. Das Konzept

der modellbasierten Fehlerursachenanalyse berücksichtigt deshalb Fehlermodelle, um die Ursache für den Fehler automatisch zu finden. Abbildung 1 zeigt das zugrundeliegende Prinzip dieser Arbeit. Ausgehend von dem Physikalischen Netzwerk, in dem die Kommunikationsgeräte vernetzt sind, werden von den Geräten oder Messinstrumenten Daten zur Verfügung gestellt, die den Ist-Zustand abbilden. Ergänzend dazu stehen Netzwerkmodelle bereit, die den Soll-Zustand vorgeben wie z.B. Protokolle (Nachrichtenaufbau, Ablauf), projektierte Topologien, oder Schedules. Mit Hilfe des Ist-Zustandes des Physikalischen Netzwerks und den Netzwerkmodellen können nun Abweichungen / Symptome festgestellt werden, die auf einen Fehler im Netzwerk hindeuten. Von Experten erstellte Fehlermodelle ordnen Symptomen bestimmte Fehlerursachen zu. Das System gibt auf der Basis der Symptome und den Fehlermodellen eine mögliche Ursache und einen Fehlerbehebungshinweis.

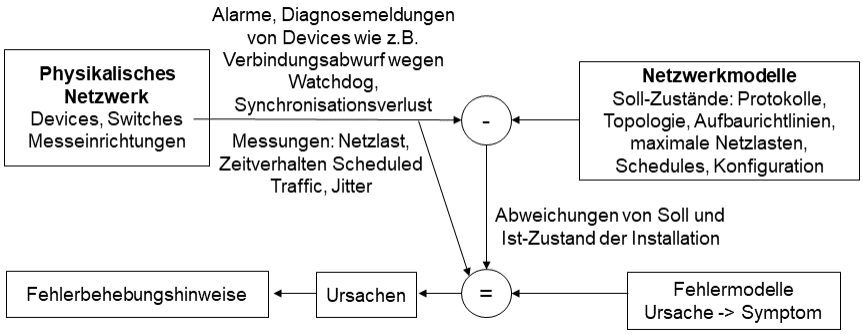



Abbildung 1: Modellbasierte Fehlerursachenanalyse

# 4 Modelle zur Fehlerursachenanalyse

In diesem Anschnitt werden die Fehlermodelle erläutert, dabei wird zunächst auf die messbaren Symptome und anschließend mögliche Ursachen eingegangen. Modelliert werden die Fehlermodelle mit der Web Ontology Language (OWL). OWL stellt dazu im Wesentlichen drei Dinge bereit: Klassen (Dinge), Beziehungen zwischen den Dingen und Eigenschaften (Attribute) dieser Dinge. Ähnlich wie in der Softwaretechnik geben Klassen einen Bauplan vor und Instanzen beschreiben dabei konkrete Dinge auf Basis einer Klasse [He22]. OWL besitzt mehrere Untersprachen die sich jeweils in Umfang und Zweck unterscheiden. Ergänzt wird OWL Description Language durch die Semantic Web Rule Language (SWRL), die OWL und RuleML kombiniert und die Bildung von Regeln für Instanzen von Klassen und unter anderem den Vergleich von Dateneigenschaften beispielsweise „Wenn das Alter einer Person unter 18 ist, ist sie ein Teenager“ ermöglicht [todosSWRL]. Eine weitere Komponente in OWL ist der reasoner, der auf Basis der Ontolgie in OWL und Regeln aus SWRL, Schlüsse zieht und die Konsistenz der Ontologie überprüft. In dieser Arbeit wird ein einfacher an OWL DL und SWRL angelehnter Syntax verwendet um eine Ontologie zu beschreiben. [Al22] bietet ein Beispiel für eine Ontologie

zur Diagnose von Herzkrankheiten, das auf die Netzwerktechnik übertragen werden könnte.

### 4.1 Messbare Symptome in Ethernet TSN-Netzwerken

Im nachfolgenden Kapitel werden beispielhaft messbare Symptome genauer betrachtet und eingeordnet. Dabei wird auf das Symptom, die Messgröße, Quelle und deren Erkennungsort eingegangen.

**Symptom 1:** *Deadline für den Scheduled Traffic wird verletzt* (*DeadlineScheduledTrafficViolated):* Bei der Deadline für den Scheduler Traffic steht eine vorgegebene Konfiguration bereit, die feste, wiederholende Zeitslots für die Queues im Ethernet Controller definiert. So werden Zeitslots definiert für beispielsweise zyklische Echtzeit-Kommunikation und „Best Effort“ Traffic. Dieser Schedule wird vom Nutzer einmal für die gesamte Domäne festgelegt und dann zyklisch von dem Ethernet Controller abgearbeitet. Wird dieser Schedule verletzt kann dies am besten mit Messtechnik ermittelt werden, die erfasst in welchen Zeiträumen diese Frames gesendet oder empfangen wurden. Da die Geräte in einem zeitsensitiven System synchronisiert sind weiß jedes Gerät wann welcher Slot aktiv sein müsste. Anschließend kann ein SOLL-IST-Vergleich erfolgen, um festzustellen ob der Schedule verletzt wird.

**Symptom 2:** *Die Reihenfolge des Scheduled Traffic ist nicht wie erwartet* (*SequenceScheduledTrafficNotasExcpeted*): Im Scheduled Traffic gibt es eine vorgegebene Reihenfolge der Frames. Wird diese verletzt kann dies am besten mit zusätzlicher Messtechnik ermittelt werden. Dabei erfolgt die Aufnahme der Reihenfolge der angekommen Frames, die dann mit der vorgegangenen Reihenfolge verglichen wird. Denkbar wäre auch eine Erkennung der Kommunikationspartner untereinander. Generell kennt jeder Teilnehmer die vorgegebene Reihenfolge und könnte selbst feststellen in welcher Reihenfolge die Frames ankommen.

**Symptom 3:** *Telegrammverluste* (*PacketLoss):* Telegrammverluste sind je nach (Industrie-)Protokoll schon im gewählten Kommunikationsprotokoll erkennbar, wenn dieses beispielsweise den Erhalt von Frames quittiert oder Frames erwartet, kann erkannt werden ob und wann Frames ausbleiben oder verloren gehen. Dies kann auch von den Geräten selbst erfolgen, da sie Teil des jeweiligen Kommunikationsnetzwerkes sind und selbst am besten wissen ob Frames ausbleiben.

**Symptom 4:** *Jitter zyklischer Kommunikation*. (*JitterOutOfBounds):* Hierbei kann die Erfassung des Symptoms durch Messtechnik erfolgen. Die Messtechnik misst den Abstand der kritischen Frames und speichert diese für die Auswertung ab. Prinzipiell könnte auch der Kommunikationspartner einen Jitter messen. Die Gegenstelle erwartet zyklische Datenpakete und kann dessen Abstand messen und so Abweichungen erkennen.

**Symptom 5:** *Netzlast in Kommunikationsklassen* (*netload):* Dieses Symptom beschreibt unerwartet auftretende Netzlast im Netzwerk. Das Symptom kann am besten durch

Messtechnik erfolgen die den Netzwerkverkehr mitschneidet und klassifiziert. Der gemessene und klassifizierte Netzwerkverkehr muss dabei mit vorgegebenen normalen Zuständen z.B. Zyklus oder Datenrate, Pakete pro Zeiteinheit oder benötigte Bandbreite verglichen werden.

**Symptom 6:** *Abwurf von Verbindungen*. (*connectionLoss)* Hierbei ist es in Industrieprotokollen in der Regel so, dass dies durch das Protokoll selbst erkannt und Diagnosemöglichkeiten bereits bestehen. Das Netzwerkgerät selbst kann am besten feststellen ob die gewünschte Kommunikation zustande gekommen ist oder nicht.

### 4.2 Fehlerursachen in TSN-Netzwerken

Die im vorherigen Kapitel aufgezeigten Symptome liegen Ursachen zu Grunde, die in der Regel durch Expertenwissen erfasst werden. Dabei kann es sein, dass eine Ursache mehrere Symptome hat. Denkbar ist auch, dass ein Symptom mehrere Ursachen zugeordnet werden kann. Im Folgenden sind beispielhaft mögliche Ursachen für Fehler in zeitsensitiven Netzwerken aufgeführt.

**Ursache 1**: *Preemption auf einem Link nicht aktiviert. (PreemptionNotActive):* IEEE IEE 802.1Qbu (Frame-Preemption) ist eine Erweiterung des Scheduled Traffic und ermöglicht das Unterbrechen und spätere fortführen der Übertragung von nicht kritischen Frames [Ie22]. Grund hierfür kann eine falsche Gerätekonfiguration durch den Nutzer oder ein Softwarefehler (Implementierung, Treiber, Netzwerkstack) vorliegen. Erkennbar ist es an den Symptomen, dass Deadlines für den Scheduled traffic nicht eingehalten werden und kritische Telegrame einen erhöhten Jitter aufweisen können.

**Ursache 2**: *TAS auf einem Link nicht korrekt konfiguriert (TASNotConfiguredCorrectly):* Time Aware Shaper (IEEE 802.1Qbv) ist eine Möglichkeit Netzwerkverkehr in Zeitslots zu unterteilen. Diese Zeitlosts werden dann zyklisch abgearbeitet. Bei einer falschen Konfiguration kann es sein, dass Frames nicht zu den erwarteten Zeitpunkten gesendet oder empfangen können. Beispielsweise kann ein Zeitslot zu klein eingestellt werden und so würde Traffic dem darauffolgenden Zeitslot des zyklischen Tasks verschoben.

**Ursache 3**: *Topologie anders als eingelesen (TopologyDeviation)*, Bei dieser Ursache sind erwartete Geräte im Netzwerk nicht vorhanden oder aber es sind Geräte im Netzwerk die nicht dort sein sollten. Die Ursache dafür liegt vor allem an einer falschen Konfiguration der Geräte, des Netzwerks oder der Projektierung im Engineering Tool. Weiterhin ist es möglich, dass die Geräte die Konfiguration falsch einlesen oder umsetzen.

**Ursache 4**: *Domänengrenzen nicht aktiviert oder falsch konfiguriert (DomainBoundaryFaulty)*: Dies liegt in der Regel an einer falschen Konfiguration des Netzwerks oder der Geräte und kann sich in mehreren Symptomen bemerkbar machen z.B. Synchronisationsungenauigkeit (auf den falschen Master synchronisiert), einer erhöhten Netzlast oder falsche Netzwerkklassen.

**Ursache 5**: *Falsche Link-Geschwindigkeit.* (*IncorrectLinkSpeed*): Diese Ursache liegt hier häufig in einer Fehlerhaften Konfiguration der Netzwerkschnittstelle. In industriellen Bereichen wird häufig 100 Mbit/s oder 1 Gbit/s verwendet.

**Ursache 6**: *Synchronisationsgenauigkeit,Offset außerhalb der Toleranz (OffsetOutOfBounds):* Dies kann viele Ursachen haben wie z.B. an falschen Geräte- oder Netzwerkkonfiguration. Aber auch eine fehlerhafte Zeitsynchronisationsimplementierung oder Netzüberlastung ist möglich. Umwelteinflüsse (Temperatur) und auch Angriffe auf das Netzwerk können ebenfalls die Ursachen sein.

### 4.3 Fehlermodelle

In Abbildung 2 ist eine Ontologie für eine Fehlerursachenanalyse vorgeschlagen. Mit Hilfe dieser Ontologie lassen sich nun Formulierungen bilden, um Fehlermodelle zu definieren. Für eine vollständige Modellierung wäre es notwendig, dass Dateneigenschaften zu den Klassen hinzugefügt werden und mit Hilfe von SWRL-Regeln für Instanzen festgelegt werden. Damit wäre dann eine Formulierung wie zum Beispiel Wenn ein *TSNDevice* die Symptome *DeadlineScheduledTrafficViolated* und *JitterOutOfBounds* hat, ist die Wahrscheinlichkeit hoch, dass auf einem Link *PreemptionNotActive*. Die vollständige Modellierung übersteigt allerdings den Rahmen dieser Arbeit. Die nachfolgenden Fehlermodelle dienen der Verdeutlichung und folgen keinem konkreten OWL oder SWRL Syntax.

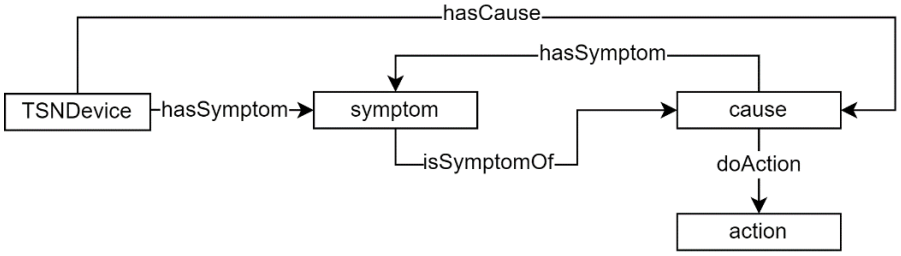



Abbildung 2: Klassen und Beziehungen der Ontologie

**Definition Fehlermodell 1:**

TSNDevice1 hasSymptom *DeadlineScheduledTrafficViolated*
TSNDevice1 hasSymptom *JitterOutOfBounds*
*DeadlineScheduledTrafficViolated* isSymptomOf *PreemptionNotActive*
*JitterOutOfBounds* isSymptomOf *PreemptionNotActive*
➔ TSNDevice1 hasCause *PreemptionNotActive*
*PreemptionNotActive* doAction CheckePreemptiononDevices

**Definition Fehlermodell 2:**

TSNDevice1 hasSymptom *SequenceScheduledTrafficNotAsExcpeted*

TSNDevice1 hasSymptom *DeadlineScheduledTrafficViolated*
*SequenceScheduledTrafficNotasExcpeted* isSymptomOf *TASNotActivated*
*DeadlineScheduledTrafficViolated isSymptomOf TASNotActivated*
➔ TSNDevice1 hasCause *TASNotConfiguredCorrectly*
*TASNotActivated* doAction CheckTASConfiguration

**Definition Fehlermodell 3:**

TSNDevice1 hasSymptom *DeadlineScheduledTrafficViolated*
*DeadlineScheduledTrafficViolated* isSymptomOf *IncorrectLinkSpeed*
➔ TSNDevice1 hasCause *IncorrectLinkSpeed*
TSNDevice doAction CheckDeviceLinkSpeed

# 5 Validierung der Fehlerursachenmodelle mit Prototypen

Um die in den vorangegangenen Abschnitten aufgezeigten Ansätze für Fehlerursachenmodelle zu validieren, werden zwei fehlerbehaftete Situationen innerhalb der nachfolgenden Testtopologie bestehend aus einem TSN-Controller, drei TSN-Switches und vier TSN-Devices nachgestellt.

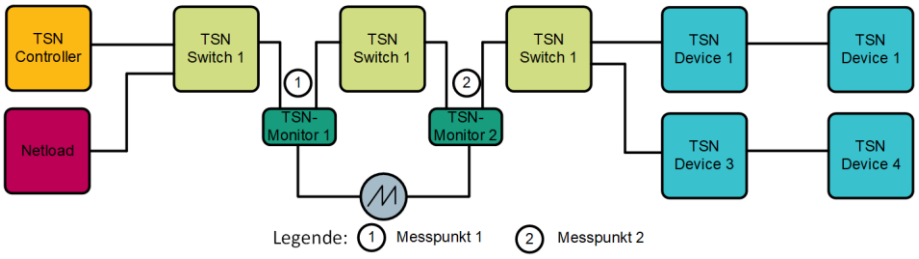



Abbildung 3: Testtopologie bestehend aus TSN Komponenten und Messtechnik

Ergänzt wird der Aufbau durch zusätzliche Messtechnik in Form von zwei TSN-Monitoren, einem Oszilloskop zur Visualisierung des Netzwerkverkehrs und einem Netzlastgenerator. Die innerhalb der Topologie eingesetzten TSN-Switches unterstützen sowohl Datenraten von 100 Mbit/s als auch 1 Gbit/s sowie die TSN-Standards IEEE 802.1AS (Zeitsynchronisation), IEEE 802.1Qbu (Frame Preemption) und IEEE 802.1Qbv (Time Aware Shaping). Der TSN-Controller und die TSN-Devices unterstützen ebenfalls Datenraten von 100 Mbit/s und 1Gbit/s und den TSN-Standard IEEE 802.1AS. Für die Kommunikation im fehlerfreien Zustand wird eine Datenrate von 1 Gbit/s und eine Zykluszeit von 1 ms projektiert. Nachfolgend ist als Referenz zum Vergleich mit den in den nächsten Abschnitten dargestellten Fehlkonfigurationen die korrekte Kommunikation an den Messpunkten 1 und 2 (siehe Abbildung 4), dargestellt.

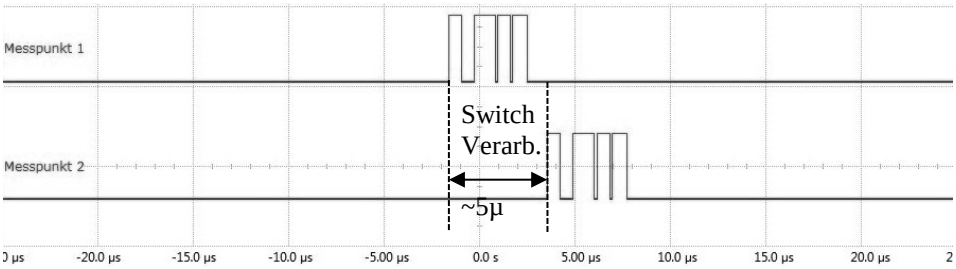



Abbildung 4: Fehlerfreie Outbound Kommunikation an den Messpunkten 1 und 2

### 5.1 Fehlerhafte Frame Preemption Konfiguration auf einem Link

Um die Auswirkungen eines fehlerhaft konfigurierten Links aufzuzeigen an dem Preemption nicht aktiviert wurde, wird Preemption zwischen den TSN-Switches zwei und drei deaktiviert. Dies führt dazu, dass der Nutzdatenverkehr zwischen den zwei Switches nicht mehr davon profitiert, Vorrang vor dem Best-Effort Datenverkehr zu erhalten. Das beschriebene Verhalten ist in Abbildung 5 abgebildet. Zu sehen ist, dass bei einer fehlerhaften Konfiguration die Deadline für den Scheduled Traffic nicht eingehalten werden kann. Anstatt dem direkten aufeinanderfolgen der Frames, wie im Normalfall in Abbildung 4, kann das Symptom *DeadlineScheduledTrafficViolated* identifiziert werden, da die Frames erst deutlich später (~7 µs) am Messpunkt 2 ankommen als erwartet.

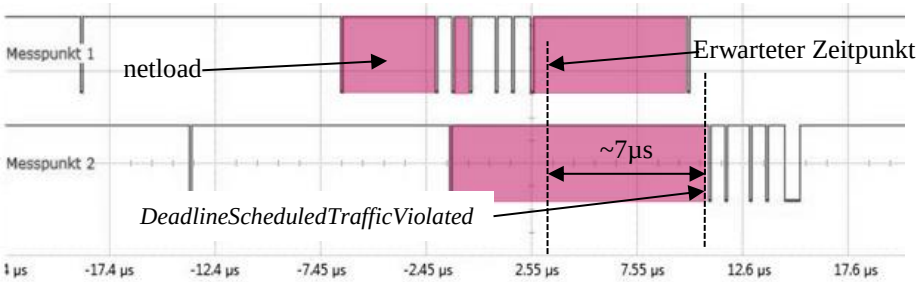



Abbildung 5: Fehlkonfiguration Frame Preemption an Messpunkt 2

### 5.2 Fehlerhafte Time Aware Shaper Konfiguration

Das zweite Fehlerbild zeigt die Auswirkungen einer fehlerhaften TAS-Konfiguration. Die TAS-Konfiguration sieht vier Phasen vor: Die erste Phase für Netzwerk-Management Datenverkehr wie beispielsweise die Zeitsynchronisation (gelb,NM), die zweite Phase für die echtzeitkritischen Nutzdaten (orange, Krit.), die dritte Phase dient wird für Best-Effort Datenverkehr genutzt (rot,BE) und letztlich eine Phase in der kein Datenverkehr übertragen werden soll (blau,NT). Zwischen den TSN-Switches 1 und 2 wurde die TAS-

Konfiguration korrekt eingestellt. Zwischen den TSN-Switches 2 und 3 kam es zu einer Fehlkonfiguration in der Form das die Phase für die echtzeitkritischen Nutzdaten kürzer

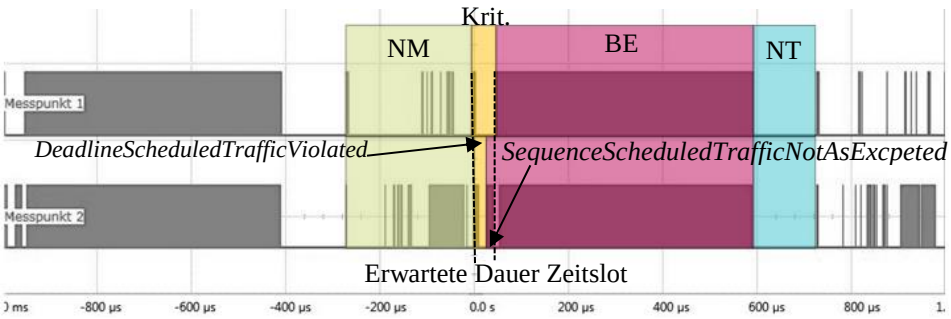



Abbildung 6: Fehlkonfiguration TAS am Messpunkt 2

berechnet wurde und die Phase für den Best-Effort Datenverkehr anstatt dessen verlängert wurde. Zum einen ist in Abbildung 6 erkennbar, dass der Zeitlost für die kritische Echtzeitkommunikaiton kleiner als erwartet ausfällt (*DeadlineScheduledTrafficViolated)*, das hat zur Folge, dass sich in dem eigentlich kritischen Zeitslot bereits Best-Effort-Traffic befindet (*SequenceScheduledTrafficNotAsExcpeted*).

# 6 Bewertung und Zusammenfassung

In diesem Paper wurde der Ansatz einer modelbasierten Fehlerursachenanalyse für zeitsensitiven Ethernet-Netzwerken vorgestellt. Dazu wurden Fehlermodelle beschrieben, die die Ursache-Wirkungsbeziehung zwischen Fehlerursache mit messbarem Symptom herstellt. Für die Fehlerursachenanalyse wurde eine Ontologie erstellt, die für ein computergestütztes System beispielweise mit Dateneigenschaften, SWRL-Regeln und exakteren Beschreibungen der Symptome und der Ontologie, zukünftig weiter ausgebaut werden soll. Ausgewählte Fehlermodelle wurden an realen Ethernet TSN-Systemen exemplarisch durch Messungen überprüft.

In der weiteren Arbeit sollen die Geräte und Messtechnik erweitert werden um automatisch Symptome erkennen zu können, dabei ist auch zu definieren wie genau die Symptome aussehen und ob dabei nicht bereits auf Informationen der eingesetzten Protokolle zurückgegriffen werden kann. Des Weiteren sollen die Fehlermodelle ausgebaut und auf größere und komplexere Netzwerke übertragen werden. Es soll ein System entwickelt werden, dass auf Basis der Wissensdatenbank mit den Fehlermodellen in der Lage ist, die Wahrscheinlichkeit für verschiedene Fehlerursachen anzugeben und in einer Netzwerktopologie einzugrenzen. Zudem soll der Ansatz neben drahtgebundener Ethernet TSN-Kommunikation auf drahtlose Kommunikation z.B. mit WLAN oder 5G als auch nicht zeitgesteuerter Kommunikation erweitert werden. Denkbar ist in ferner

Zukunft auch, dass eine vollständig modellierte Ontologie mit einem KI-basierten Ansatz verglichen werden kann, um so Aufwände und Genauigkeiten der Ansätze zu vergleichen.

## Literaturverzeichnis


[W17] Wollschläger, Martin; Sauter, Thilo; Jasperneite, Jürgen: The Future of Industrial Communication. In: IEEE Industrial Electronics magazine IEEE, März 2017

[An18] T. Anusasamornkul: A Network Root Cause Analysis and Repair System, 2018 6th International Symposium on Computational and Business Intelligence (ISCBI), 2018

[FSJ21] Ferfers, T. et al., Investigation in IoT and 5G architectures for deployment of Artificial Intelligence into urban mobility and production, 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)

[Ge09] Gerhards, R.: The Syslog Protocol. RFC 5424,38, 2009

[TSN22] Time-Sensitive Networking (TSN) Task Group, https://1.ieee802.org/tsn/, 25.03.2022

[IB97] Isermann R.; Ballé P., Trends in the Application of Model-Based Fault Detection and Diagnosis of Technical Processes, Control Engineering Practice, 5 (5), 1997

[APL81] Armor A. F et al., "Root Cause Analysis of Fossil Power Plant Equipment Failures: The EPRI Program," in IEEE Transactions on Power Apparatus and Systems, June 1981

[Pn20] PNO, Diagnosis for PROFINET – Guideline for PROFINET, PROFIBUS Nutzerorganisation e.V., 2020

[In22] Indu-Sol GmbH, PROFINET-Inspektor® NT, https://www.indu-sol.com/produkte/profinet/diagnose/profinet-inspektorr-nt/, verfügbar 15.08.2022

[Ac22] acontis technologies GmbH, EC-Monitor: Software library for monitoring (sniffing) of EtherCAT® networks, https://www.acontis.com/en/ec-monitor.html, 15.08.2022

[BYD13] Bin Y. et al., Key failure modes of solder joints on HASL PCBs and root cause analysis, 2013 14th International Conference on Electronic Packaging Technology

[TSTM21] Tong V et al., Machine Learning based Root Cause Analysis for SDN Network, 2021 IEEE Global Communications Conference (GLOBECOM), 2021, pp. 1-6

[JC22] Jiang J. -R.; Chen Y. -T., Industrial Control System Anomaly Detection and Classification Based on Network Traffic, in IEEE Access, pp. 41874-41888, 2022

[BDO16] Bunte A. et al., Integrating semantics for diagnosis of manufacturing systems, 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 1-8

[He22] Heflin, J., OWL Web Ontology Language – Use Cases and Requirements, https://www.w3.org/TR/webont-req/, 15.08.2022

[Ie22] IEEE Inc, https://www.ieee802.org/1/pages/802.1bu.html , 16.08.2022

[Al22] Alagha, H. M., Diagnosis Heart Disieases Using Ontology and SWRL Rules, https://library.iugaza.edu.ps/thesis/121921.pdf, 17.08.2022

[PN21] PROFINET Specification IEC 61158 (V2.4) MU3, Profibus User Organization 2021.

# Practical investigation of an industrial converged network based on OPC UA PubSub and TSN

Oliver Konradi[1], Mario Schoppmeier[2], Lukasz Wisniewski[3], Jürgen Jasperneite[4]

**Abstract:** Converged networks offer the flexibility that modern Industry 4.0-ready applications require. State-of-the-Art technologies such as OPC UA and TSN have the potential to be the enabler the convergence of Information Technology and Operational Technology. While TSN includes real-time capabilities to standard Ethernet and allows the coexistence of applications with real time and non-real time requirements in a single network, flattens OPC UA the standard automation pyramid. However, even though these technologies and the idea of the combination of these technologies (OPC UA TSN) exists, they are not widely applied in industrial environment. Therefore, a practical heterogeneous setup is created, to serve as a testbed and to give first hands on experience to SME's. The practical realization of a network with OPC UA TSN is hindered by the limited available devices and the lack of available configuration and engineering tools.

## 1 Introduction

The topic of Industry 4.0 draws a lot of challenges and new requirements to the industries of today. Such new challenges need new concepts and solutions in various fields. Industrial networks must be reconsidered and extended with appropriate technologies to fulfill requirements of plug-and-play or self-x services. By considering such concepts, the boundaries of classic Information Technology (IT) and Operational Technology (OT) vanish. It is therefore an important factor to consider technologies to enable efficient and performant communication of diverse applications inside one network infrastructure. Promising technologies that enable such converged industrial networks are the Time Sensitive Network (TSN) and the Open Platform Communication Unified Architecture (OPC UA).

TSN extends the classic Ethernet with abilities to handle messages in real-time. It's standardized mechanisms allow coexistence of different applications that need to exchange both real-time and best effort data in a single network. Furthermore, OPC UA complements these abilities with an information model for applications so it can be integrated on every vertical level of the automation pyramid. In addition to that, it offers properties, such as

[1] Technische Hochschule OWL, Institut Industrial IT, Campusallee 6, 32657 Lemgo, Germany oliver.konradi@th-owl.de

[2] Fraunhofer IOSB-INA, Campusallee 1, 32657 Lemgo, Germany, mario.schoppmeier@iosb-ina.fraunhofer.de

[3] Technische Hochschule OWL, Institut Industrial IT, Campusallee 6, 32657 Lemgo, Germany, lukasz.wisniewski@th-owl.de

[4] Fraunhofer IOSB-INA, Campusallee 1, 32657 Lemgo, Germany, juergen.jasperneite@iosb-ina.fraunhofer.de

platform independence and function scalability. Herewith, it can be implemented on a variety of platforms[IJ13]. With the usage of the newer specification OPC UA PubSub, the horizontal communication on field level with OPC UA is a viable option. OPC UA and TSN in combination are among others considered as enablers for converged industrial networks [Br19; Fi18; Me18].

To investigate the limitations and advantages of the these technologies, a testbed is conceptualized and realized. This testbed will demonstrate and visualize core functionalities of OPC UA and TSN so it can be used in training and workshops for SME's. In this work, the current version of the testbed is presented. It consists of multiple end devices and TSN capable switches. In this work, the first evaluation and conclusions regarding the implementation of a converged network.

This document is structured as follows: Section 2 describes briefly the important functionalities and features of TSN and OPC UA. In Section 3, the testbed is presented and described including its implemented components and software libraries. In the subsequent section 4, the testbed of Section 3 is critically analyzed. Finally, the content is summarized and concluded in section 5.

## 2 Background

This section gives a basic introduction to Time Sensitive Networking (TSN) and OPC UA technologies in order to better follow the subsequent chapters.

### 2.1 Time Sensitive Networking

Industry 4.0 is characterised by dynamically networked production lines with frequent reconfiguration of machines and supporting systems as well as the use of cloud technology and data-based smart services. This often requires continuous, efficient and flexible communication from the field level to the edge or public cloud, in which different protocols are used simultaneously. Under the name Ethernet TSN (Time Sensitive Networks), the IEEE is developing a set of amendments for Ethernet making it a truly real-time protocol. The use of Ethernet TSN for industrial automation is currently defined in the profile standard IEC/IEEE 60802 TSN-IA. [SJ21; SKJ18; SKJ20]

We will focus on the features Fram Preemption, Time Aware Shaping including Time Synchronization according to IEEE 802.1Qbu, IEEE 802.1Qbv and IEEE 802.AS respectively as they provide the TSN mechanisms for a deterministic communication.

### 2.1.1 802.1AS - Timing and Synchronization

IEEE 802.1AS defines a clock synchronization protocol to achieve a common notion of time among network members in a defined network domain. The basic principle of this protocol is to cyclically send time information (synchronization frames) from the Grandmaster (GM) to timekeepers (also called slaves or Ordinary Clocks (OCs)). The OCs synchronize their clock to the received timing information. However, due to link delays, the timing information ages. For this reason, the link delay of the synchronization frames is determined and compensated. For an increased accuracy, the individual determination of the line delays and the delays of the frames in the bridges is used. This timing compensation takes place in each bridge. A bridge that can compensate for timing information this contributing to better overall synchronization accuracy, is called a Transparent Clock (TC). [SJ21; SKJ18; SKJ20]

### 2.1.2 802.1Qbv - Time Aware Shaper

In order to achieve low latency, Time Aware Shaping (TAS) was included in the IEEE 802.1Q standard [802.1]. The function makes it possible to determine by configuration which queues can dispatch frames at which times. This is called queue masking. Dedicated gates are connected upstream of the queues and suppress transmission at the output port (ëgress") as shown in the Figure 1. [SJ21; SKJ18; SKJ20]

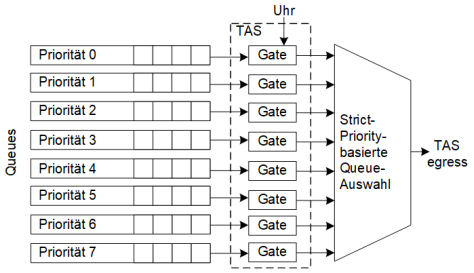



Fig. 1: IEEE 802.1Qbv

### 2.1.3 802.1Qbu - Frame Preemption

In order to be able to achieve low latency and secure determinism, in addition to TAS, a second mechanism called preemption was included in the IEEE 802.1Q standard. Preemption allows frames that are currently being sent on an Ethernet port to be interrupted in order to send a higher priority frame, as Figure 2 shows. A frame that can be interrupted is called preemptable. A frame that is allowed to interrupt other frames as "preemptive". [SJ21; SKJ18; SKJ20]

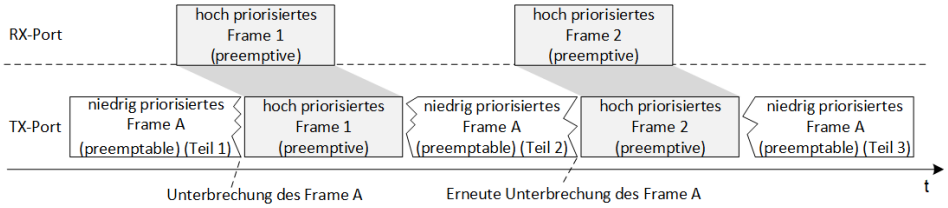



Fig. 2: IEEE 802.1Qbu

## 2.2 OPC UA PubSub

Open Platform Communication Unified Architecture (OPC UA) is a standard that defines communication between various devices and systems for industrial applications. It unifies the previous OPC specification and obsoletes them to enable platform and vendor independence. The communication is based on a Client/Server model, where a client sends requests to a server to which the server will send appropriate responses. In order to do that, a client establishes a (secure) channel to the OPC UA Server and requests the information of the servers *AdressSpace* [OP17].

OPC UA complements the Client/Server model with the PubSub model. A publisher offers various *DataSets* to subscribers. A subscriber chooses the *DataSets* that are of interest and subscribes to it to retrieve the messages. If a predefined event is triggered, the publisher sends *NetworkMessages* to its subscribers. Other than in the Client/Server model, the publishers and the associated subscribers are loosely coupled, and many-to-many communication is enabled. Compared to OPC UA Clients, OPC UA Server have typically a larger footprint and need more computation power, because in addition to respond to clients, requests have to be processed. However, by using the PubSub model, the computation power is shifted to a so called Message Oriented Middleware. The main responsibility of the Message Oriented Middleware is to forward messages to the correct address. OPC UA defines various middleware solutions [OP18]. This enables scalable PubSub applications on the end devices.

A simplified PubSub infrastructure is depicted in Figure 3.

Multiple publishers and subscribers are connected via a *Message Oriented Middleware*. The subscribers register themselves on a registry to the publisher [6]. The publisher send their *Dataset* as a payload of *NetworkMessages* via the *Message Oriented Middleware*. The middleware distributes the messages to the associated subscriber. OPC UA defines two types of Message Oriented Middleware, the broker-less and the broker-based middleware. Both solutions trade off footprint and performance. Since the broker-less solution does not require a software to process the data before it gets forwarded and it does not need a configuration

[5] Based on Figure 1 of [OP18]
[6] The registry entity is not depicted in the figure

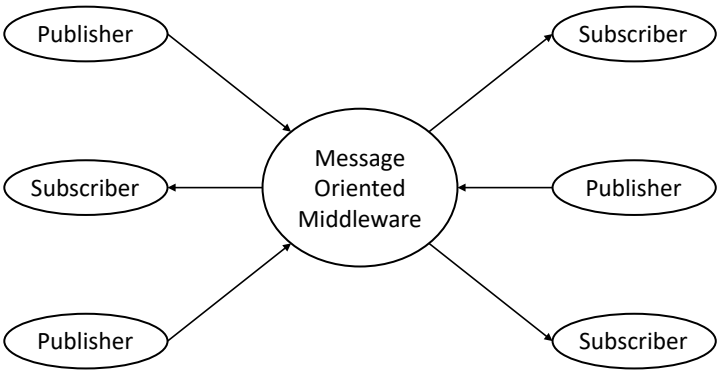



Fig. 3: Basic OPC UA PubSub infrastructure[5]

in a simple setup, it is expected to have a reduced transmission time and a lower complexity. Therefore, in this work the broker-less Message Oriented Middleware is used.

## 3 OPC UA TSN testbed

This section describes the practical setup and the implemented devices as well as the relevant software stacks. At the beginning, the overall testbed is being presented and described. The following subsections describe the network devices and end devices in more detail.

In order to give SMEs the opportunity to carry out training on real devices, to investigate the possibilities and limits of the OPC UA technologies in combination with TSN and to test prototypes, a testbed is being realized. A concept of such a testbed is depicted in Figure 4.

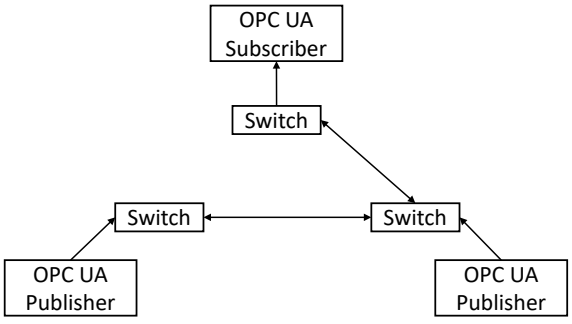



Fig. 4: Concept of the testbed

Three end devices in Figure 4 execute two OPC UA Publishers and one OPC UA Subscriber. They are connected over TSN capable switches forming a line topology. One publisher is

executed on a device, which implements time synchronization, and VLAN tagging. These implementations enable the TSN features Frame Preemption and Time Aware Shaping in the TSN domain. This device will publish messages to the subscriber with a higher priority than the other device. This demonstrates that applications that send messages with real-time requirements are able to coexist in a network with applications that have best-effort requirements.

The current version of the practical realization is shown below:


Fig. 5: Picture of current version of testbed

The current version of this network is depicted in Figure 5. These devices are intended to be replaced or extended by either prototypes or finished products to better demonstrate the capabilities of the OPC UA over TSN. By replacing and extending the components, the

advantages of the vendor independent specifications and implementations will be highlighted, which ultimately results in a heterogeneous network. Finally, adequate measurement tools are used to verify and visualize the implemented features.

### 3.1 TSN Switches

The used switches in this network are two Layerscape LS1028A from the vendor NXP and the TSN switch RELY-TSN-BRIDGE 22.1.0 from the vendor RELYUM. An overview of the implemented TSN feature set is given in Section 4.4.1. The Relyum switch is completely configurable via a graphical user interface accessable by a web interface. The manufacturer hands over the switch to the customer before it is set up, and the user can log in directly to the web interface with the standard password. The user manual provides all relevant information for setting it up and configuring the TSN-Domain. On the other hand, the NXP switch is configurable via a command line interface. It should also be mentioned that the LS1028 does not have an operating system in its factory state. This must be compiled manually with all relevant drivers and packages. A guide from the manufacturer is available for this. Both devices implement Time Synchronization, Time Aware Shaping and Frame Preemption. The associated parameter can be configured using the according user interface.

Table 1 shows the comparison of the two devices at operating system and software level:

| Stack | Layerscape LS1028A | RELY-TSN-BRIDGE 22.1.0 |
|---|---|---|
| Linux-Version | OpenIL 1.10 | RelyTsnBridge-352422008 |
| Kernel-Version | 5.4.3-rt1 | 5.4.0 |
| Software-Stack | ptp4l 3.0 / tsntool V0.4 | ptp4l 2.0 |

Tab. 1: Switch specifications

### 3.2 End Devices and OPC UA Implementation

The OPC UA Publishers and OPC UA Subscriber are each executed on Raspberry Pi's of the 4th generation with 4 GB RAM. The available OPC UA PubSub implementations are limited to only two versions (see Section 4.4.2). Due to this limitations, we focused on the open source C library open62541[7] as this is well known implementation and with the available example implementations it offers an easy entry into the topic of PubSub. Since the focus is rather on the network then on an application, the example implementation is used. There the publishers continuously send their system time every 100 ms over the TSN switches to the subscriber to fill the payload of the messages. The subscriber accepts these messages and writes the values into its node set. Simultaneously, an OPC UA Server is running on the same device as the OPC UA Subscriber and offers these values to OPC UA Clients. The message are being transmitted via UDP Multicast messages.

[7] http://www.open62541.org/

### 3.3 Modifications to the testbed

The current version of the network is an approaches the requirements of the final network. However, due to the extensive chip shortage on the broad tech market, the procurement of devices is in some cases delayed indefinitely. Therefore, the realization of the final network needs to be delayed. The final network requires a complete heterogeneous environment. Such a network highlights the need for missing engineering tools for the configuration of each network device.

## 4 Evaluation

This section evaluates the practical setup of section 3. It discusses the importance of combining TSN and OPC UA, the interoperability of the TSN switches and the relevance of TSN on end devices. Furthermore, it shows a brief overview of available devices and implementations related to TSN and OPC UA.

### 4.1 OPC UA over TSN

OPC UA as a modern architecture for industrial communication offers many benefits such as platform independence and scalability. Especially, the Client/Server model was implemented in several applications [Br19; Li20; Pf18]. PubSub reduces the overhead of communication establishment of the Client/Server model and enables a one-to-many communication. This attributes enables the possibility for OPC UA to be implemented on the lowest level of the automation pyramid. Especially on this level, the communication requirements tends to increase in terms of real-time requirements. A technology that prevails in this field is TSN. It enhances the standard Ethernet network with real time capabilities and allows various communication protocols with different requirements to coexist in a single network. By using the combined technologies OPC UA TSN in a network, a continuous connected network horizontally and vertically on every layer of the automation pyramid is possible. This results into a convergence of Information Technology (IT) and Operational Technology (OT) structures.

### 4.2 TSN Switch configuration and Interoperability

All switches in the setup implement the 802.1AS clock synchronization, 802.1bu frame preemption and 801.1Qbv time aware shaping according to the corresponding standard amendment. Although the TSN features are standardized, the configuration and the user interface is not. The RELY-TSN-Bridge is configurable via a web interface. On the other hand, the Layerscape LS1028A is configurable via the command line tool *tsntool*. It

becomes apparent that a large scale network needs additional tools and interfaces other than the manual input of a user to configure these devices. This is especially true for a large heterogeneous setups.

### 4.3 TSN on End Devices

A TSN domain is not only limited to end devices but is extendable to all network devices including end devices. In the current version of the testbed shown in 3, the end devices do not implement the necessary functions to be integrated into the TSN domain. In order to do so, the end devices should be synchronized with the grandmaster clock in the TSN domain and add a VLAN Tag to the Layer 2 frames so that the sent frames are prioritizable. Such devices will be included in a future version of this setup.

### 4.4 Overview of available devices and software libraries

This subsection provides an overview of available TSN devices and identifies the provided features. Subsequently, the OPC UA PubSub implementations are listed and discussed.

#### 4.4.1 Available TSN devices

In order to create a time sensitive network, suitable devices are needed. A summary of available devices can be found in the following table:

| Vendor | Model | Supported TSN Features | | |
|---|---|---|---|---|
| | | AS | Qbv | Qbu |
| Belden/Hirschmann | BOBCAT Series | x | x | - |
| Relyum | RELY-TSN-BRIDGE | x | x | x |
| NXP | Layerscape LS1028A | x | x | x |
| CISCO | IE 4000 Series | x | x | - |
| Phoenix Contact | FL Switch TSN 2316 | x | x | x |
| TTTech | TSN Starter-Kit incl. B&R 0ACST052.1 | x | x | x |
| InnoRoute | RaspberryPi RT-Hat | x | x | - |

Tab. 2: Available and TSN capable devices[Ko22]

Table 2 shows that most devices implement time synchronization according to IEEE 802.1 AS and Frame Preemption according to IEEE 802.1 Qbu. This table also shows that there are not many devices overall available[8]. Furthermore, available TSN capable end devices are more limited than the TSN capable network devices.

[8] The list in this table is to the best of our knowledge complete

However, as the demonstrations in Section 3 show, the interface and the options for the configurations are diverge. This is shows that the manual configuration is possible but not practical, which demonstrates the need for a tool for parametrization.

### 4.4.2 Available OPC UA PubSub implementations

OPC UA is available in different programming languages. Haskamp et al. summarized and performed a benchmark of available OPC UA implementations in [Ha17]. However, this document focuses on the available implementations with a client/server model. The new specification of OPC UA [OP18] defines the PubSub communication model. However, this specification is still not applied and updated in all existing software libraries yet. In fact, there are in total two available implementations that offer the PubSub model, the open source implementation open62541 and the software library from unified automation[9].

## 5 Conclusion

This work showed a network, which includes the technologies OPC UA PubSub and TSN. It was designed to be a testbed such as the possibilities and limitations of these technologies can be investigated. Furthermore, this network is designed as an assisting tool for training and workshops for SME's. The network is beeing evaluated and essential points such as the limitation of devices as well as tools for configuration are highlighted.

The current version of this network is already able to show that there is is limitations especially for TSN capable devices as well as OPC UA PubSub implementations. The implementation of such a network is complicated by indefinite delivery times in addition to the equipment offered, due to the chip shortage. No statement can be made about the economic viability, as the prices are volatile due to the current market situation. Some devices are available for up to a four-digit amount and are thus many times more expensive than commercially available devices that do not implement these technologies.

The TSN devices that are integrated in this network need additional configuration to execute and demonstrate TSN relevant functions. The configuration of these devices have a different interface to do so. The switch from the vendor NXP is configurable via the command line with the tool *tsntool* while the switch from the vendor Relyum is configurable via a web interface. Based on the current version of this setup, specifically the TSN functionalities Frame Preemption and Time Aware Shaping are to be configured.

This approach is at first done manually. However, a manual configuration is suitable for small scale networks. For complex heterogeneous modular environments in the industry, this approach is not practical. Configurations on larger scale should be done with appropriate

[9] https://www.unified-automation.com/products/pubsub-sdk.html

tools. Such tools do not exist at the time. The presented testbed will be extended and modified until it is heterogeneous and easily configurable also for large scale networks. This network will also help to define requirements for tools such as it can be easily integrated in a large and complex network.

## Acknowledgment

This research is part of the research project Converged Industrial Networks for Industry 4.0 (CINI4.0). This project is funded and supported by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) on the basis of a decision by the German Bundestag with the funding code *IGF Vorhaben 309 EN* and a sponsor from Belgium, Flanders Innovation & Entrepreneurship.

## References

[Br19] Bruckner, D.; Stănică, M.-P.; Blair, R.; Schriegel, S.; Kehrer, S.; Seewald, M.; Sauter, T.: An Introduction to OPC UA TSN for Industrial Communication Systems. Proceedings of the IEEE 107/6, pp. 1121–1131, 2019.

[Fi18] Finn, N.: Introduction to Time-Sensitive Networking. IEEE Communications Standards Magazine 2/2, pp. 22–28, 2018.

[Ha17] Haskamp, H.; Meyer, M.; Möllmann, R.; Orth, F.; Colombo, A. W.: Benchmarking of existing OPC UA implementations for Industrie 4.0-compliant digitalization solutions. In: 2017 IEEE 15th International Conference on Industrial Informatics (INDIN). IEEE, pp. 589–594, 2017.

[IJ13] Imtiaz, J.; Jasperneite, J.: Scalability of OPC-UA down to the chip level enables "Internet of Things". In: 2013 11th IEEE International Conference on Industrial Informatics (INDIN). Pp. 500–505, 2013.

[Ko22] Konradi, O.; Mankowski, A.; Wisniewski, L.; Trsek, H.: Towards an Industrial Converged Network with OPC UA PubSub and TSN. 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)/, 2022.

[Li20] Li, Y.; Jiang, J.; Lee, C.; Hong, S. H.: Practical Implementation of an OPC UA TSN Communication Architecture for a Manufacturing System. IEEE Access 8/, pp. 200100–200111, 2020.

[Me18] Messenger, J. L.: Time-Sensitive Networking: An Introduction. IEEE Communications Standards Magazine 2/2, pp. 29–33, 2018.

[OP17] OPC Foundation: OPC UA Specification: Part 1 - Overview and Concepts. Version 1.04, November 22, 2017.

[OP18] OPC Foundation: OPC UA Specification: Part 14 - PubSub. Version 1.04, February 06, 2018.

[Pf18] Pfrommer, J.; Ebner, A.; Ravikumar, S.; Karunakaran, B.: Open source OPC UA PubSub over TSN for realtime industrial communication. In: 2018 IEEE 23rd international conference on emerging technologies and factory automation (ETFA). Vol. 1, IEEE, pp. 1087–1090, 2018.

[SJ21] Schriegel, S.; Jasperneite, J.: A Migration Strategy for Profinet Toward Ethernet TSN-Based Field-Level Communication: An Approach to Accelerate the Adoption of Converged IT/OT Communication. In: IEEE Industrial Electronics Magazine./, 2021.

[SKJ18] Schriegel, S.; Kobzan, T.; Jasperneite, J.: Investigation on a Distributed SDN Control Plane Architecture for Heterogeneous Time Sensitive Networks. In: 14th IEEE International Workshop on Factory Communication Systems (WFCS)./, 2018.

[SKJ20] Schriegel, S.; Kobzan, T.; Jasperneite, J.: Guideline PROFINET over TSN V1.21, Profibus International./, 2020.

# Open-Source Ethernet TSN-Testwerkzeuge für Linux

*Gunnar Leßmann[1], Kurt Kanzenbach[2], Alexander Biendarra[3], Tobias Ferfers[3], Sergej Gamper[3]*

[1]Phoenix Contact GmbH, Bad Pyrmont
*glessmann@phoenixcontact.com*

[2]Linutronix GmbH, Uhldingen-Mühlhofen
*kurt.kanzenbach@linutronix.de*

[3]Fraunhofer IOSB-INA, Lemgo
*{alexander.biendarra , tobias.ferfers, sergej.gamper }@iosb-ina.fraunhofer.de*

**Abstract:** Die Implementierung von IEEE Ethernet-TSN-Standards unter Linux erforderte und erfordert bisher spezielle Netzwerktreiber und Optimierungen, um die für industrielle Anwendungen geforderten Leistungs- und Robustheitsmerkmale zu erreichen. Die Linux Open Source Community integriert Ethernet-TSN-Funktionalitäten und Echtzeitmechanismen inzwischen aber mehr und mehr in Standard-Linux. Hierbei ist sicherzustellen, dass die spezifische TSN Hardware- und Software Implementierung des Chipherstellers die notwendigen Anforderungen erfüllt. Dieses Papier beschreibt ein Testwerkzeug, das zur Validierung der Funktionalität und Leistungsfähigkeit von Hardware mit entsprechender Linux-basierter Software genutzt werden kann. Bei dem Werkzeug handelt es sich um eine Ethernet-TSN-Testbench, die in Zukunft Quell-offen verfügbar sein soll. Mit diesem Werkzeug soll die Einführung von Ethernet TSN in Komponenten und Unternehmen und somit in den Markt der Industriellen Kommunikationstechnik vereinfacht und beschleunigt werden. Zusätzlich wird die notwendige Interoperabilität verbessert.

## 1 Motivation

Ethernet-basierte industrielle Kommunikationssysteme wie z.B. Profinet erfordern unter Linux anforderungsgerechte Netzwerktreiber, um die für industrielle Anwendungen erforderliche Leistung und Robustheit zu erreichen [17] [18]. In älteren Linux-Versionen (< Kernel-Version 5) sind keine geeigneten Ethernet Standard-Interfaces und Methoden enthalten, die für zeitgesteuerte Kommunikation unter Echtzeitbedingungen geeignet waren. Spezifische, proprietäre Netzwerktreiber sind nicht nur komplex und aufwändig in der Wartung, sie profitieren auch nicht von Entwicklungen, die in der Hauptentwicklungslinie von Linux („Mainline“) erfolgen [15]. Des Weiteren ist eine Implementierung auf neuen Hardwareversionen oder Varianten ohne spezielle Herstellerkenntnisse der Ethernet-Hardware nahezu unmöglich. Die Implementierungen entsprechen der bisher dominierenden OT-Systemdenkweise, bei der geschlossene und starre Systeme auf Basis proprietärer Plattformen auf Echtzeiteigenschaften optimiert wurden. Die zukünftige Integration von klassischen OT-Systemen und -Services mit der IT-Welt und der Nutzung von z.B. IEEE Standards und Time Sensitive Networking (TSN) erfordert nicht nur bei den Ethernet-Standards [2, 3, 4, 5, 6, 7], sondern auch bei den Implementierungen eine offene Standard-Architektur und Teilkomponenten [1]. Ein Nutzenversprechen von Ethernet-TSN-Standards ist die Implementierung in Mainstream-Hardware und -Software [14]. Dies gilt insbesondere für die Linux-Plattform, die zunehmend den Anforderungen von hochdeterministischen industriellen Applikationen erfüllen kann. Bei Linux liegt die Verantwortung für die TSN-Netzwerktreiber bei den Herstellern der Ethernet-Hardware. Die Schnittstellen zur Applikation sind in den neuesten Linux Kernelversionen bereits größtenteils etabliert. Dies ermöglicht es prinzipiell die Hardware eines Herstellers austauschen zu können, ohne dass sich das auf die Applikation auswirkt. Dies wird in Abbildung 1 dargestellt.

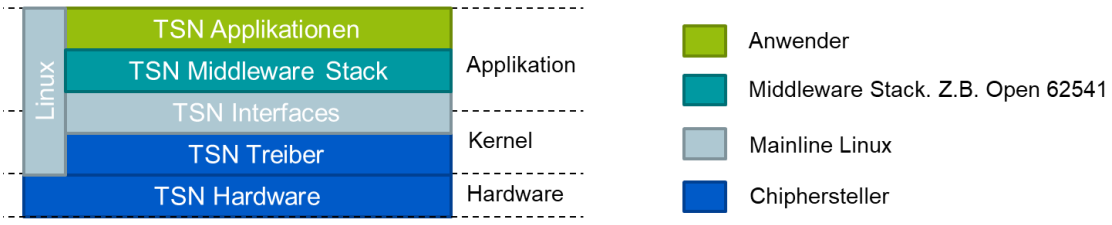



Abbildung 1: Architektur und Verantwortlichkeiten unter Linux

Damit Ethernet-TSN-Mechanismen auf jeder Hardware genutzt werden können, muss sichergestellt sein, dass die Funktionalität und Qualität der Ethernet-Hardware und derer Treiber für die jeweilige industrielle Applikation gewährleitet ist. Hierbei spielen folgende Aspekte eine Rolle:

- Robustheit auch dann, wenn andere Applikationen auf derselben Hardware laufen
- Robustheit der Echtzeitkommunikation gegen externe und interne Netzlasten
- Erreichbare Zykluszeiten, Latenzen und deren Jitter sowie Abhängigkeiten zu den o.a. Faktoren
- Unterstützung und korrekte Umsetzung der notwendigen Ethernet TSN-Standards, wie z.B. 802.1Q

Nur unter Einhaltung dieser Aspekte kann Ethernet-TSN das Versprechen erfüllen, dass Komponenten auf Basis von Standard-IT-Hardware und -Software (COTS – Commercial off-the-shelf) genutzt werden können. Automatisierungshersteller wie z.B. Phoenix Contact, welche Linux-basierte offene Steuerungsarchitekturen wie PLCnext einsetzten sind darauf angewiesen, dass diese Kriterien von den jeweiligen Zulieferern erfüllt werden. Um dabei einen Vendor Lock-In zu vermeiden, muss jede in Produkten eingesetzte Hardware und Software die oben beschriebenen Kriterien erfüllen. Da weder die Hardware noch die Treiber in einer solchen offenen Architektur von spezifischen Automatisierungstechnikhersteller selbst stammen, stellt sich die Frage nach einer Referenzimplementierung, die herstellerneutral zur Validierung der jeweiligen Linux TSN-Implementierung herangezogen werden kann.

## 2 Technischer Ansatz einer Ethernet TSN-Testbench für Linux

Aus den oben beschriebenen Gründen wurde eine Ethernet TSN-Testbench entwickelt, die eine Evaluierung der Ethernet Hard- und Software einer Linux-basierten Endstation (siehe Kapitel 4) ermöglicht. Zusätzlich kann diese Testbench auch für die Prüfung der korrekten TSN-Kommunikation in Netzwerken genutzt werden. Abbildung 2 zeigt das Konzept und die Architektur der Testwerkzeuge, die zusammen und mit ihrer Dokumentation als TSN-Testbench bezeichnet werden. Dieser Beitrag beschreibt den Aufbau, Funktionalität, Konfigurationsmöglichkeiten sowie Messergebnisse. Darüber hinaus wird ein Ausblick auf zukünftige Entwicklungsmöglichkeiten sowie die geplante Open-Source-Aktivität gegeben.

Die vorgestellten, frei verfügbaren und offene Ethernet-TSN-Testwerkzeuge für Realtime-Linux können zu einem wesentlichen Schlüsselfaktor für die Verbreitung von Systemen wie Profinet und OPC UA Pub/Sub über TSN beitragen. Sie stellen sicher, dass unabhängig vom Hersteller, der verwendeten Hardwareplattform und der jeweiligen Kommunikations-Middleware eine robuste und performante Implementierung der TSN-Schnittstellen unter Linux gewährleistet ist.

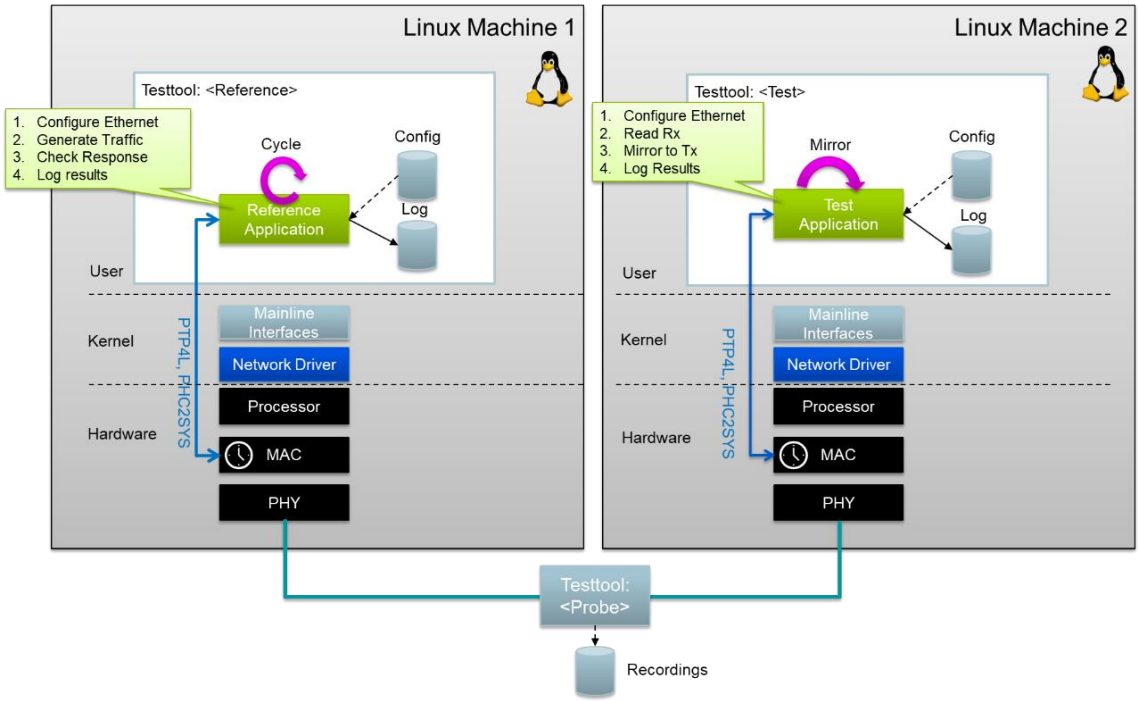



Abbildung 2: Architektur der Ethernet TSN-Testwerkzeuge

Es ist geplant diese Testwerkzeuge Open-Source bereit zu stellen, um eine kontinuierliche Weiterentwicklung von TSN auf vielen Plattformen sowie neue Funktionalitäten zu ermöglichen. Dies ist im Interesse der Automatisierungsanbieter, da möglichst viele Chipsätze mit TSN-fähigen Schnittstellen unter Linux genutzt werden sollen, ohne dass hierfür spezielle Treiberentwicklungen notwendig sind. Auch die Hersteller der Chipsätze profitieren von quelloffenen Testwerkzeugen zur Validierung.

# 3 Architektur und Funktionen der Ethernet TSN-Testbench

Die TSN-Testbench besteht aus mehreren Komponenten, welche zur Evaluierung und Validierung von TSN-fähigen Chipsätzen und deren Linux Kernel Treibern entwickelt wurde. Die Testbench besteht zurzeit aus einer Profinet TSN Simulation. Generische Erweiterungen für andere Ethernet basierte Protokolle wie zum Beispiel OPC/UA PubSub sind ebenfalls implementiert. Die Bestandteile und andere eingesetzte Werkzeuge sind:

- **Reference & Test-Applikation**: Versand und Empfang von Ethernet Traffic unterschiedlicher Traffic-Classes.
- **I/O Tasks**: Simulation von User Tasks
- **Ethtool, tc**: Linux Netzwerkinterface Konfiguration
- **Ptp4l, phc2sys**: Zeitsynchronisation gemäß IEEE 802.1AS
- **Hackbench**: CPU-Last Erzeugung
- **Iperf**: General Purpose Ethernet Traffic Generierung

Die *Referenz*-- und *Testapplikation* sind die Kernkomponenten der TSN-Testbench. Diese sind verantwortlich für die Erzeugung, Versand, Empfang und Prüfung der Profinet Ethernet Frames. Die Referenzapplikation generiert Ethernet Frames verschiedener Traffic Klassen und sendet diese an die Testapplikation. Dort werden die Pakete gespiegelt und im nächsten Zyklus an die Referenzapplikation zurückgesendet. Für jeden einzelnen Ethernet Frame wird die Round-Trip Zeit bestimmt. Anhand derer wird die Echtzeitfähigkeit der gesamten Kommunikation abgeleitet. Des Weiteren, ist jedes Paket mit einem Zykluszähler und einem bestimmten Inhalt versehen. Die Prüfung des Zählers und des Inhalts innerhalb der Applikationen gibt Aufschluss über den korrekten Versand und Empfang, sowie die Sicherstellung der Paketreihenfolge. Außerdem kann die Testbench eingesetzt werden, um spezifische Hardwareparameter wie beispielsweise den Interpacket-Gap oder die Genauigkeit der Frame Preemption zu bestimmen. Um dies messtechnisch darzustellen ist eine zusätzliche Messkomponente erforderlich. Im Rahmen dieses Papiers wurde dies durch eine FPGA-basierte Lösung realisiert (siehe Kapitel 5.2).

Zu der TSN-Testbench gehören ebenfalls die *I/O Tasks*, welche Aufgaben eines typischen PLC-Programms simulieren. Diese werden zeitversetzt zum Versand und Empfang der Netzwerkpakete ausgeführt. Die Konfiguration der Netzwerkschnittstellen und die notwendige Zeitsynchronisation erfolgen mit den Linux Werkzeugen *ethtool*, *tc* und *ptp4l*. Um reale Bedingungen auf den Testmaschinen darzustellen, wird parallel zur Profinet Simulation sowohl CPU- als auch Netzwerklast erzeugt. Alle Applikationen und Werkzeuge greifen auf Standard Linux Interfaces für die Netzwerkkommunikation und Konfiguration zurück.

Die Softwarearchitektur der TSN-Testbench ist in Abbildung 3 dargestellt.

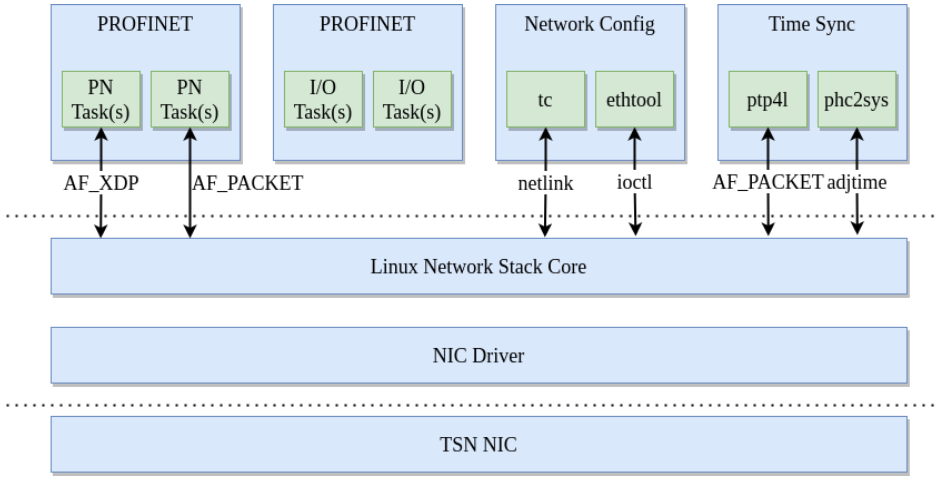



Abbildung 3: TSN-Testbench Softwarearchitektur

Damit die TSN-Testbench auf verschiedenen Hardwareplattformen, Prozessoren und Netzwerkkarten eingesetzt werden kann, ist sie in weiten Bereichen konfigurierbar. Dazu gehören unter anderem die Zykluszeiten, sowie die Anzahl und Größe der Ethernet Frames aller beteiligten Traffic Klassen. Ebenso ist das Schedulingverhalten aller involvierten Tasks konfigurierbar. Zuletzt sind Debug- und Loggingmöglichkeiten vorgesehen.

# 4 Grundlagen Ethernet TSN-End Station Architektur

Ein Ethernet TSN-Netzwerk besteht grundsätzlich aus TSN-Endstations in der Form von Steuerungen und IO-Endgeräten, den TSN-Switches als Infrastrukturkomponente, der Konfiguration des TSN-Netzwerkes und den Applikationen die das TSN-Netzwerk nutzen [10]. Im Profinet-Standard V2.4 ist ein Model einer solchen TSN-Endstation definiert. Abbildung 4 [1] zeigt dieses Modell, dass etwa an das ISO/OSI-Referenzmodell angelehnt ist: Unten sieht man die Bitübertragungsschicht in Form des Ethernet PHY und darüber die Sicherungssicht in Form einer Ethernet MAC. Hier im Model ist auch die Express-Mac (eMAC) dargestellt, da Endstations Preemption unterstützen können. Oberhalb der MAC ist der synchronisierte Netzwerkzugang mit acht (zeitgesteuerten) Queues dargestellt, auf den die in Profinet definiten Kommunikationsklassen (HIGH, LOW, RT, NW, Non-stream) abgebildet werden. Es wird gezeigt, dass dieses Queueing auf die Working Clock mittels IEEE 802.1AS synchronisiert werden kann und über IEEE 802.1Q Managed Objects konfiguriert werden kann. Oberhalb der drei Queues, welche die Echtzeitkommunikationsklassen enthalten, arbeitet die PPM (Provider Protocol Machine), welche die zyklischen Echzeitframes erzeugt und bearbeitet. Oberhalb der PPM ist die Applikation dargestellt.

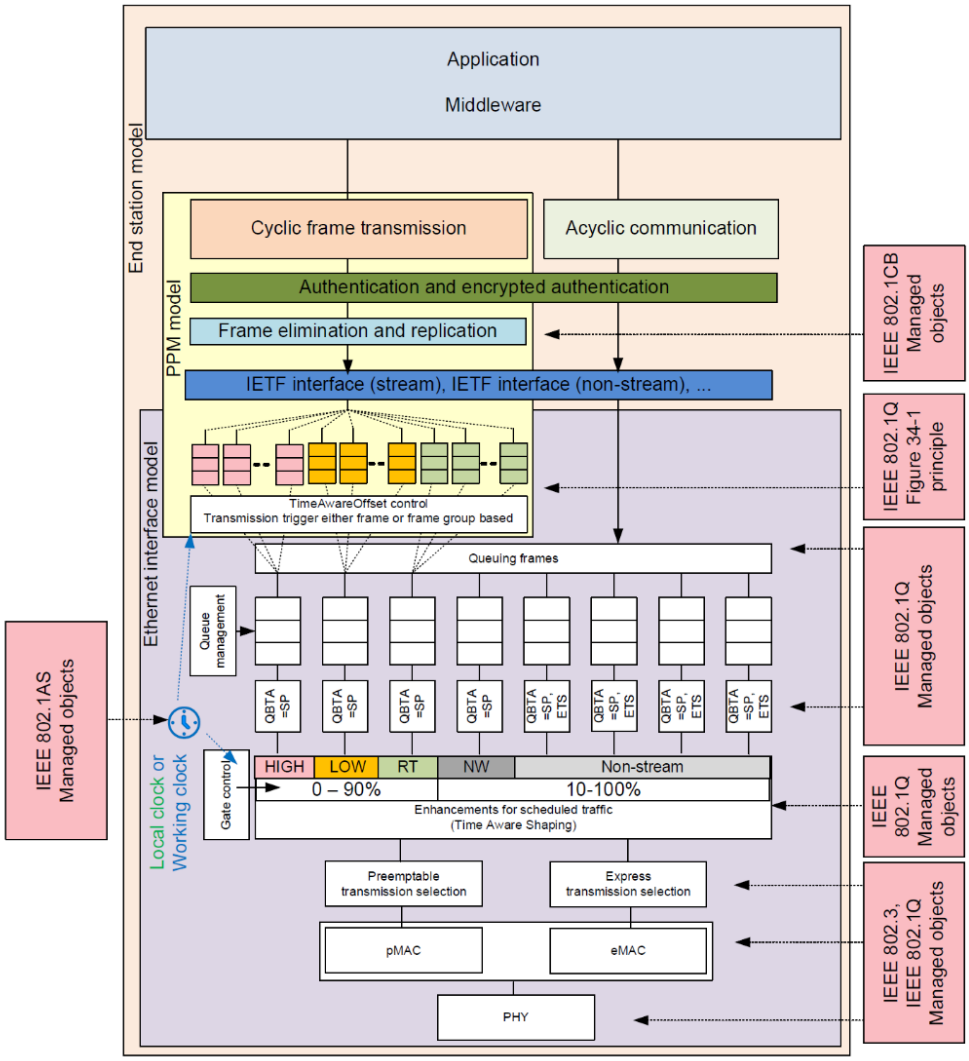



Abbildung 4: Endstation Modell nach Profinet-Standard 2.4 [11]

## 4.1 Zugriff und Konfiguration der Ethernet-Netzwerkschnittstelle unter Linux

Für die Kommunikation über die Ethernet-Netzwerkschnittstelle verwendet die Testbench zwei Arten von Socketinterfaces für die Rx- und Tx-Richtung. Zum einen RAW-Sockets, welche für die Kommunikation auf OSI-Layer 2 vorgesehen und mit Standard Linux Sockets vergleichbar sind. Zum anderen AF_XDP Sockets. Diese werden auch für die Kommunikation auf OSI-Layer 2 eingesetzt. Bei XDP (Express Data Path) handelt es sich um eine Leistungsverbesserung der RAW-Socket Schnittstelle, die ab dem Linux-Kernel 4.12 grundsätzlich zu Verfügung steht. Der Unterschied zu RAW-Sockets besteht darin, dass Teile vom Netzwerkstack im Kernel umgangen werden und das Speichermanagement der Ethernet Frames in die User Space Applikation ausgelagert wird. Das hat Vorteile hinsichtlich Determinismus und Geschwindigkeit. Während die RAW Sockets für Management-Datenverkehr eingesetzt werden, werden die AF_XDP Sockets für TSN High, TSN Low, RTC und RTA Datenverkehr genutzt. Die Konfiguration der Ethernet-Netzwerkschnittstelle erfolgt anhand von ethtool und tc. Abbildung 5 zeigt die Architektur mit XDP.

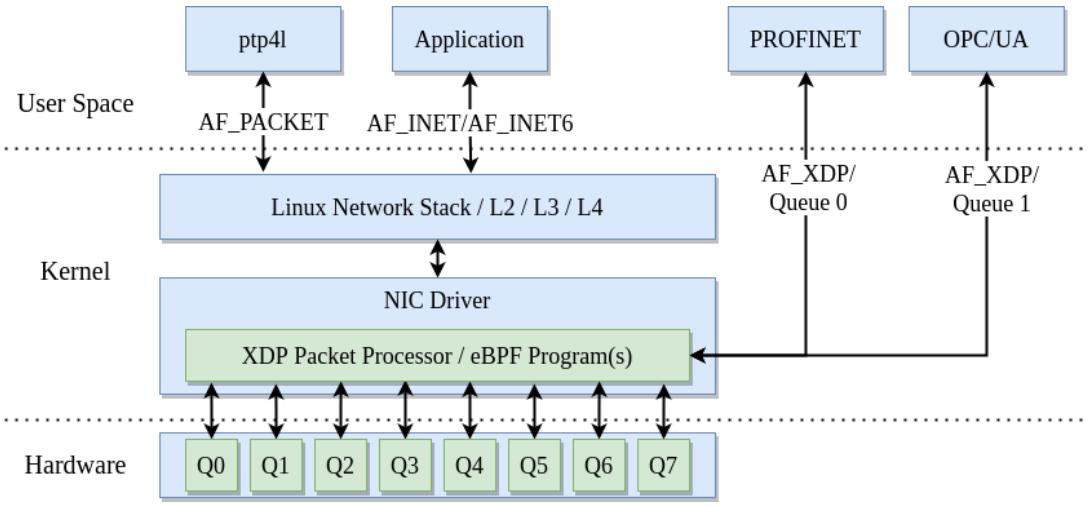



Abbildung 5: Architektur mit XDP

Die Ethernet-Netzwerkschnittstelle hat mehrere getrennte Rx- und Tx-Queues. Ebenso haben diese Queues separate Interrupts (und damit seperate Interrupt Threads bei der Verwendung eines Linux mit PREEMPT RT. Folglich kann Rx und Tx parallel ausgeführt werden. Die Applikation hingegen hat einen AF_XDP Socket pro Queue für die Kommunikation in Rx- und Tx-Richtung.

Um sicherzustellen, dass das Netzwerk und die Applikation synchronisiert werden, wird die Applikation Phc2sys verwendet. Phc2sys ist Teil Linux PTP Project [17] und kann eingesetzt werden um die Systemzeit auf eine PTP Hardware Clock (PHC) einer Netzwerkschnittstelle zu synchronisieren. Hierdurch wird ermöglicht, dass die Testbench für das Scheduling der Pakete die CLOCK_TAI als Scheduling Clock verwenden kann. Die PHC, welche als Zeitreferenz für CLOCK_TAI herangezogen wird, wird wiederrum über die Synchronisationsanwendung Linux PTP von einem Synchronisationsmaster im Netzwerk synchronisiert. Die Zeitsynchronisation mit Linux PTP wird im folgenden Abschnitt behandelt.

## 4.2 Zeitsynchronisation mit Linux PTP

Die Zeitsynchronisation ist eine der tragenden Säulen von Ethernet-TSN. Sie ermöglicht Funktionen wie zeitgesteuertes Senden, das Time Aware Shaping oder auf Anwendungsebenen die zeitliche Nachverfolgung von Ereignissen (Sequence of Events). Im Falle von Profinet over TSN wird die Zeitsynchronisation nach IEEE 802.1AS-2020 umgesetzt. Übertragen auf Abbildung 4 bedeutet dies, dass die Zeitsynchronisation für die Synchronisation der Working Clock eingesetzt wird und hier für die Time Aware Offset Control und das Time Aware Shaping eingesetzt wird.

Um diese Funktion für die Testbench umzusetzen, wurde auf die für Linux verfügbare Open Source Lösung Linux PTP zurückgegriffen. Linux PTP wird von Richard Cochran betreut, unterliegt der GNU General Public Licence und kann für die Synchronisation von Endstations eingesetzt werden. Im Kontext der Testbench wird die Synchronisation mittels IEEE 802.1AS genutzt. Die Konfiguration selbst erfolgt über eine Konfigurationsdatei, die beim Starten von Linux PTP als Parameter übergeben wird. Um die für die Synchronisation der Working Clock notwendige Genauigkeit zu erreichen, unterstützt Linux PTP die Nutzung von Hardwareuhren und der Hardwarezeitstempelung, falls dies von der genutzten Hardware und dem zugehörigen Linux Treiber unterstützt wird. [17]

# 5 Exemplarische Tests mit der Ethernet TSN-Testbench

## 5.1 Testumgebung und Topologie

Um die korrekte Funktionsfähigkeit der Testbench sicherzustellen und die erzeugten Ergebnisse verifizieren zu können, wurde die in Abbildung 7**Fehler! Verweisquelle konnte nicht gefunden werden.** gezeigte Topologie benutzt.

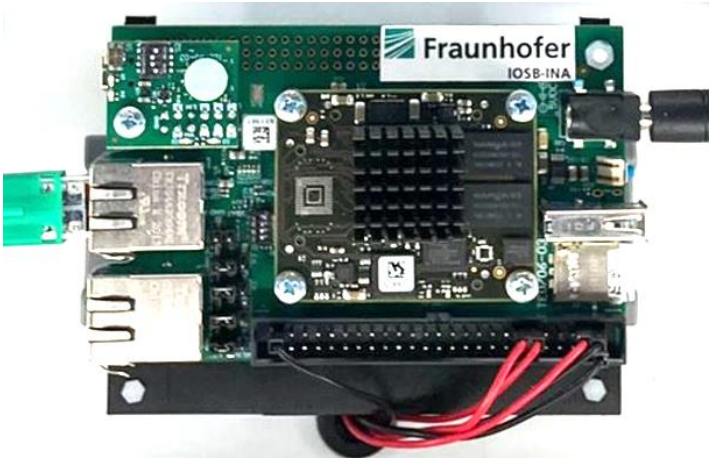



Abbildung 6: TSN-Monitor

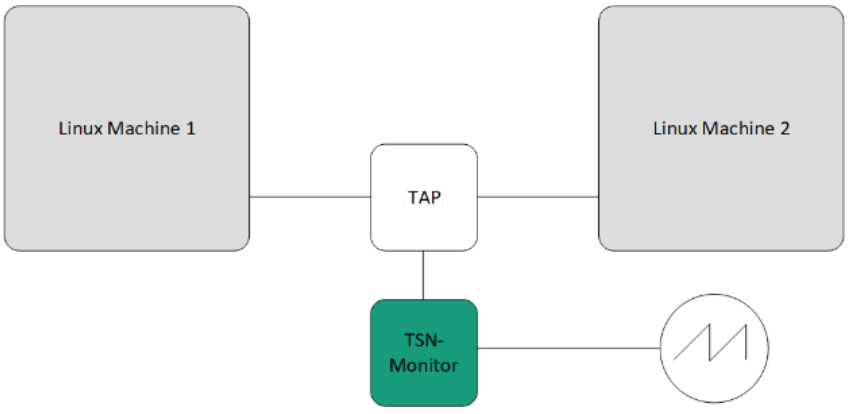



Abbildung 7: Testaufbau

Die Topologie besteht aus zwei über ein Ethernet Test Access Point (TAP) verbundenen Linux-Plattformen mit einer Intel i225 Ethernet-Netzwerkschnittstelle. Auf einer wird die Testbench betrieben und auf der zweiten die Test-Applikation. Über das Ethernet TAP werden die von der Testbench erzeugten Ethernet Frames passiv abgegriffen und an den Fraunhofer TSN-Monitor (Abbildung 6**Fehler! Verweisquelle konnte nicht gefunden werden.**) weitergeleitet. Dieser analysiert den abgegriffenen Nachrichtenverkehr und visualisiert diesen anschließend auf einem Oszilloskop. Auf die eingesetzte Messtechnik soll im folgenden Abschnitt genauer eingegangen werden.

## 5.2 Ethernet TSN-Messtechnik

Neben der vorgestellten Testbench kommt als zusätzliches Messwerkzeug der TSN-Monitor des Fraunhofer IOSB-INA zum Einsatz. Der TSN-Monitor ist eine auf einem FPGA basierte Eigenentwicklung mit dem Ziel Nachrichtenverkehr auf Basis von OSI-Layer-2 Informationen oder freien Filtern zu selektieren und auf einem Oszilloskop zu visualisieren. Eingesetzt wird der TSN-Monitor im Rahmen des am Fraunhofer IOSB-INA ansässigen TSN-Testlabor im Rahmen von Evaluierungs- und Entwicklungs-Projekten oder aber im Rahmen von Messedemonstratoren zur Veranschaulichung von TSN-Technologien wie beispielsweise im Rahmen der Hannover Messe 2022 auf dem Messestand der Profibus Nutzerorganisation oder der Firma NXP.[13][14]

Anhand der visualisierten Informationen können anschließend Funktionen wie Frame Preemption, Time-Aware Shaping oder das zeitgesteuerte Senden überprüft werden. Die Konfiguration der Filter-Kriterien erfolgt wie in Abbildung 8 gezeigt über ein Webinterface. Hier ist es möglich neben dem gesamten in eine Kommunikationsrichtung verlaufenden Nachrichtenverkehr acht weitere Filter zu konfigurieren. Filter-Kriterien sind bspw. die Quell- oder Ziel-MAC Adresse des Ethernet-Frames, der Ethertype oder aber das Vorhandensein des VLAN-Tags mit spezifischen Prioritäten. Frame-Inhalte oberhalb des OSI-Layer 2 können unter Angabe eines entsprechenden Offsets frei parametriert werden.

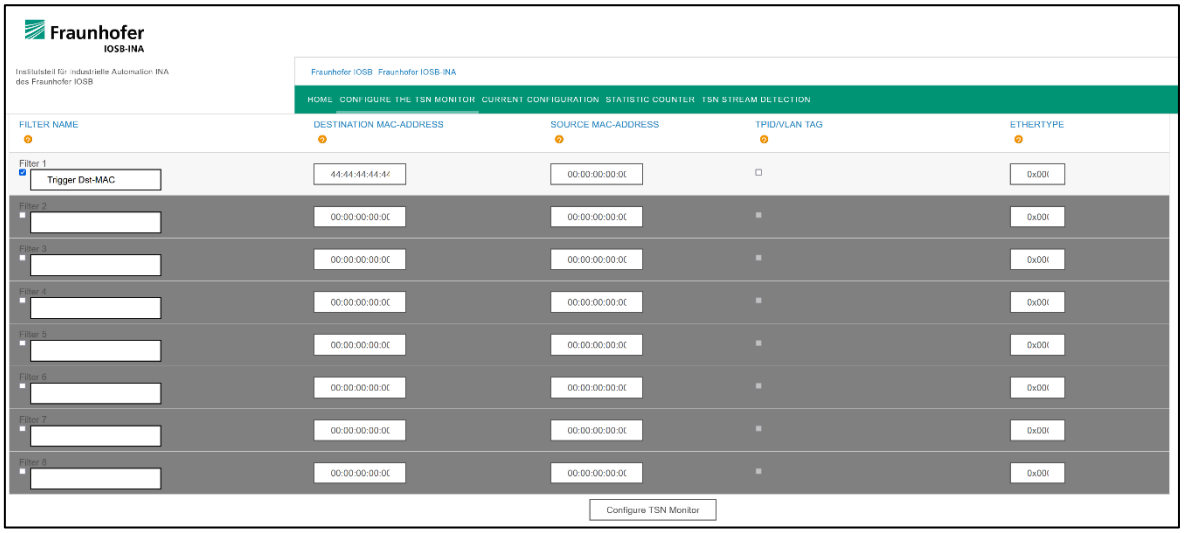


Abbildung 8: Konfigurationsoberfläche Ethernet TSN-Monitor

## 5.3 Ergebnisse Echtzeittests

Mit Hilfe der Testbench, einer Anwendung für Echtzeit-Tests [12] und der Anwendung Hackbench wurde die Echtzeit Scheduling Latenz von Linux in Kombination mit dem eingesetzten Prozessor gemessen. Hierzu wurden periodisch Timer-Threads erzeugt und dessen erwartete der tatsächlichen Aufwachzeit gegenübergestellt. Durch Hackbench wurde zusätzliche Systemlast durch miteinander kommunizierende Prozesse erzeugt. Die Messung wurde über eine Dauer von 16 Stunden durchgeführt. Das Ergebnis der Messung ist in Abbildung 9 dargestellt und zeigt eine Begrenzung der Latenzzeit auf max. 45 µs über alle CPU-Kerne.

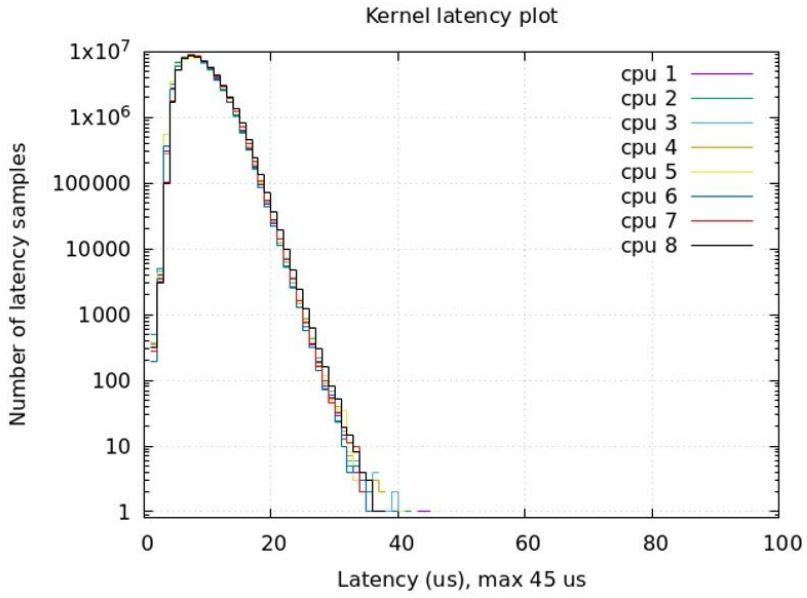



Abbildung 9: Messergebnis Echtzeit Scheduling Latenz

## 5.4 Interpacket Gap-Test

Im nächsten Schritt wurde ein Test durchgeführt, der die Interpacket-Gap überprüft. Industrielle Steuerungen, die auch in Bezug auf Ethernet-TSN in die Kategorie der Endstation eingeordnet werden können, müssen in der Lage sein Ethernet Frames in Form von Bursts zu verschicken und dabei ein konstantes Interpacket-Gap einzuhalten. Für die Durchführung der Messung wurde die Topologie aus **Fehler! Verweisquelle konnte nicht gefunden werden.** herangezogen. Für die Durchführung des Tests wurde die Testplattform durch die Testbench wie folgt parametriert: Für die Zykluszeit wurde ein Wert von 500 µs angenommen. Es wurden Burts mit 64 Ethernet-Frames erzeugt, die in verschiedenen Durchläufen eine Länge von jeweils 64 Byte, 128 Byte, 256 Byte oder 512 Byte aufwiesen. Außerdem wurde ein Test mit 256 Ethernet-Frames und einer Länge von jeweils 128 Byte durchgeführt. Das Time-Aware-Shaping wurde auf 75% Echtzeitanteil (vergleich Abbildung 4, HIGH, LOW, RT) und 25% Nicht-Echtzeit Datenverkehr (vergleich Abbildung 4, NW, Non-Stream) eigestellt. Erwartet wurde ein Frameaustausch mit einem konstanten, minimalen Interpacket-Gap bei, der es zu keinerlei Verlusten oder Beschädigungen von Ethernet Frames kommt. Exemplarisch ist innerhalb von **Fehler! Verweisquelle konnte nicht gefunden werden.** eine Aufzeichnung zwischen beiden Linux Maschinen dargestellt, die durch den TSN-Monitor aufgezeichnet wurde. Zu sehen ist die Datenübertragung von vierundsechzig 512 Byte langen Ethernet-Frames mit einer konstanten Interpaket-Gap.

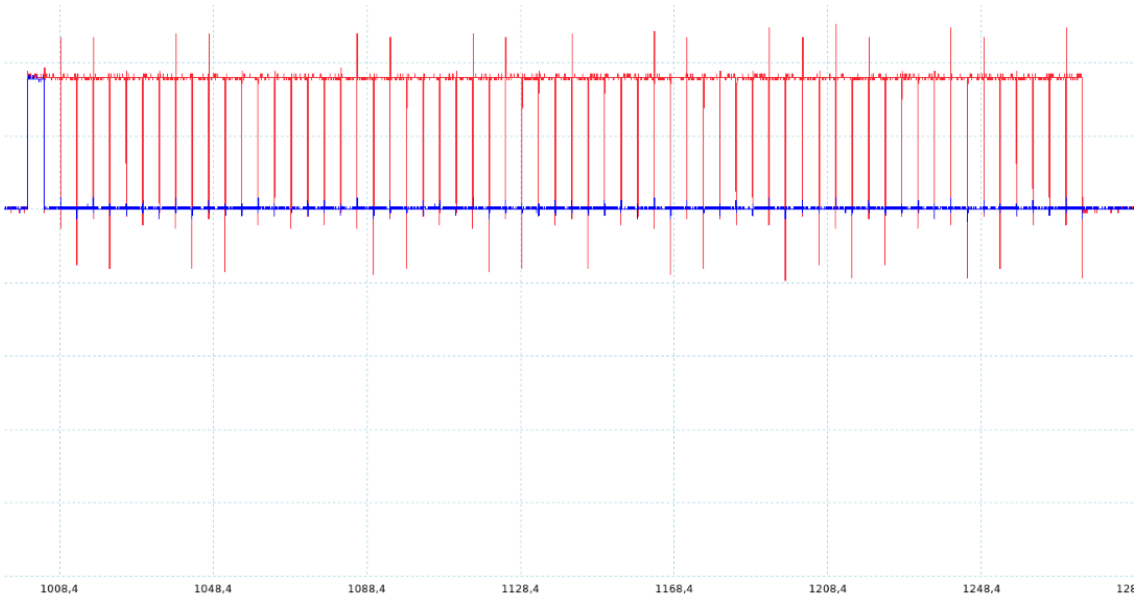



Abbildung 10: Messergebnis

## 5.5 Frame Preemption-Tests

Abschließend wurden die Testbench und der TSN-Monitor genutzt, um die TSN-Funktion Frame Preemption zu prüfen. Frame Preemption ist eine Funktion, die innerhalb der Hardware umgesetzt wird und nach entsprechender Konfiguration Ethernet-Frames niedriger Priorität unterbricht, sollte ein höher priorisiertes Ethernet-Frame übertragen werden müssen. Ist das Ethernet-Frame mit der höheren Priorität übertragen worden, wird die Übertragung des niedriger-prioren Ethernet-Frames fortgesetzt. Dadurch, dass es aktuell noch keine Schnittstelle im Linux-Kernel gibt, die es erlaubt Frame-Preemption zu konfigurieren, wurde ein Patch durch den Hersteller der Netzwerkschnittstelle zu Verfügung gestellt, um die Funktion testen zu können. Aufgabe der Testbench war es dann die unterlagerte Hardware entsprechend zu parametrieren und die Ethernet-Frames zu erzeugen. Die Parametrisierung wurde wie folgt durchgeführt: Es wird ein Echtzeit-Ethernet-Frame mit einer entsprechenden Priorität erzeugt und die Konfiguration der Hardware so durchgeführt, dass dieses das Frame ist, das die Übertragung unterbrechen kann. Des Weiteren wurden weitere Ethernet-Frames (Best Effort Traffic) generiert und mit einer gegenüber dem Echtzeit-Ethernet-Frame geringeren Priorität versehen, sodass diese unterbrochen werden dürfen. Die Einstellungen für das Time Aware Shaping wurden so durchgeführt, dass alle Gates über den gesamten Kommunikationszyklus geöffnet sind. Das Ergebnis der Messung ist in Abbildung 11 dargestellt:

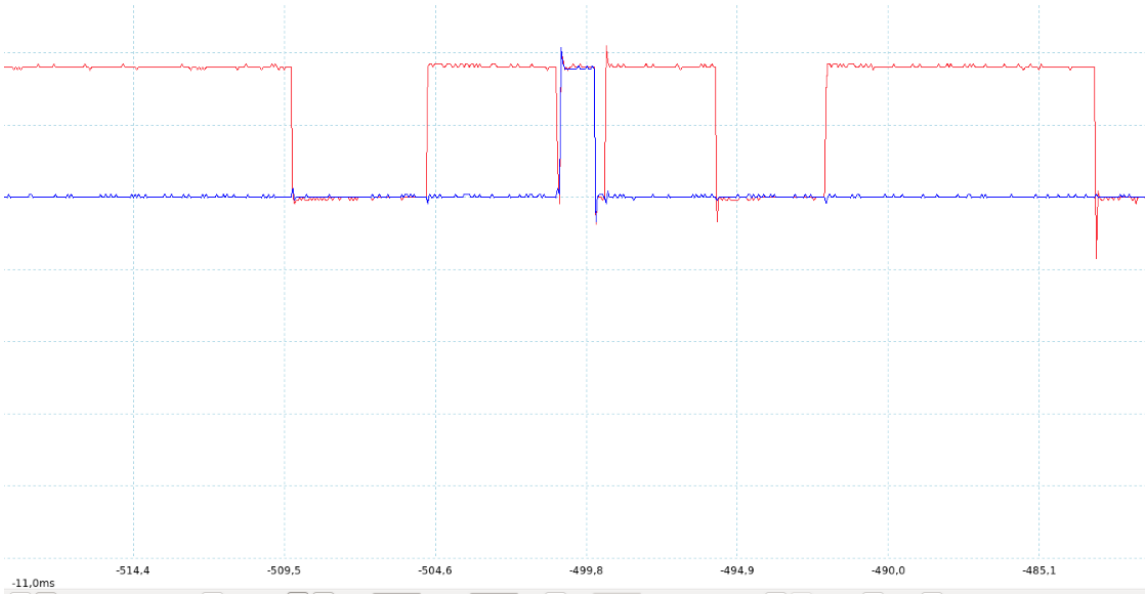



Abbildung 11: Messergebnis Frame Preemption

Der durch den TSN-Monitor visualisierte Signalverlauf des Echtzeit-Ethernet-Frames (blaues Signal) und des Best Effort Traffics (rotes Signal) zeigen die Funktion von Frame-Preemption. Während der Übertragung des Best Effort Frames tritt der Sendezeitpunkt des Echtzeit Ethernet Frames ein. Das sich aktuell im Sendevorgang befindliche Best Effort Frame wird bei Erreichen des nächsten vielfachen eines minimal langen Ethernet-Frames unterbrochen und die Übertragung des Echtzeit-Ethernet-Frames durchgeführt. Im Anschluss, nach vollständigem Versand des Echtzeit-Ethernet-Frames, wird der noch vorhandene Rest des Best Effort Frames übertragen.

# 6 Zusammenfassung und Ausblick

Die zukünftige Integration von klassischen OT-Systemen und -Services mit der IT-Welt und der Nutzung von z.B. IEEE Standards und Time Sensitive Networking (TSN) erfordert nicht nur bei den Spezifikationen, sondern auch bei den Implementierungen eine offene Standard-Architektur in Bezug auf Hardware und Software. Als offene Software-Lösung bietet sich Linux an und ist bereits in vielen Unternehmen etabliert. Es ist heute aber häufig unklar mit welcher Hardware und welchen Linux-Versionen und mit welchen Treibern welche Echtzeiteigenschaften und welche Funktionen realisiert werden können und welche Komponenten wie zueinander kompatibel genutzt werden können. In diesem Papier wurden deshalb Testwerkzeuge beschrieben, die zur Validierung von Ethernet-TSN-Hardware und Ethernet-TSN-Software, genutzt werden können. Es wurden die Architektur einer Ethernet-TSN-Testbench dargelegt und exemplarische Testergebnisse gezeigt, die die Sendeleitung von Hardware-Software-Plattformen und den Test der Funktion Preemption zeigen. Mit einem Werkzeug, dem TSN-Monitor, wurde die richtige Funktion auf der Ethernet-Leitung validiert. Die Ethernet-TSN-Testbench soll Quellcode-offen verfügbar sein, um die Einführung von Ethernet-TSN in Komponenten und Unternehmen und somit in den Markt der industriellen Kommunikationstechnik zu vereinfachen und zu beschleunigen und die Interoperabilität zu fördern. Über die offene Bereitstellung wird sichergestellt, dass die Testbench von vielen Herstellern verwendet und von der Community funktional weiterentwickelt werden kann. Hier sind Features wie Realtime-Security, Constraint-Random Pattern und vieles mehr denkbar.

# 7 Literaturverzeichnis


[1] Schriegel, Sebastian; Jasperneite, Jürgen: Migrationskonzept zur Einführung von Ethernet TSN in die Feldebene. In: at – Automatisierungstechnik, De Gruyter, 2021.

[2] IEEE Std 802-2014: Overview and Architecture, 2014.

[3] IEEE Std 802.1AB-2016: Station and Media Access Control Connectivity Discovery, 2016.

[4] IEEE Std 802.1AC-2016: Media Access Control (MAC) Service Definition, 2016.

[5] IEEE Std 802.1AS-2020: Timing and Synchronization for Time-Sensitive Applications, 2020.

[6] IEEE Std 802.1CB-2017: Frame Replication and Elimination for Reliability, 2017.

[7] IEEE Std 802.1Q-2018: Bridges and Bridged Networks, 2018.

[8] IEEE Std 802.3-2018, 2018.

[9] IEEE Standard for Ethernet IEC/IEEE 60802 Draft 1.3: Online: http://www.ieee802.org/1/files/private/60802-drafts/d1/60802-d1.pdf, 2021.

[10] Friesen, Andrej; Biendarra, Alexander; Schriegel, Sebastian: Guideline PROFINET over TSN V1.3, Profibus International, 2021.

[11] PROFINET Specification IEC 61158 (V2.4) MU3, Profibus User Organization (PNO), 2021.

[12] Schriegel, Sebastian: Kompatibilitätsverfahren für Profinet-Hardware mit Ethernet Time Sensitive Networks. In: Technologien für die intelligente Automation, 1. Aufl. 2022 Edition, Springer Vieweg, ISBN 3662647419, 7. Januar 2022.

[13] Schriegel, Sebastian; Jasperneite, Jürgen: A Migration Strategy for Profinet towards Ethernet TSN-based Field Level Communication. In: IEEE Industrial Electronics Magazine, 2021.

[14] J. Kacur und C. Williams, „Suite of real-time tests“, [Online]. Available: https://git.kernel.org/pub/scm/utils/rt-tests/rt-tests.git. [Zugriff am 03 02 2022]

[15] Fraunhofer IOSB-INA, „Fraunhofer TSN-Messtechnik zeigt Funktionsprinzip von PROFINET TSN“, [Online]. Available: https://www.iosb-ina.fraunhofer.de/de/aktuelles_news/2022/profinet-tsn.html. [Zugriff am 18.07.2022]

[16] Fraunhofer IOSB-INA, “Interoperabilität und Vorteile von Ethernet TSN auf Hannover-Messe gezeigt“, [Online]. Available: https://www.iosb-ina.fraunhofer.de/de/aktuelles_news/2022/ethernet-tsn1.html. [Zugriff am 18.07.2022]

[17] The Linux PTP Project, [Online]. Available: http://linuxptp.sourceforge.net/. [Zugriff am 19.07.2022]

[18] https://www.kernel.org/doc/html/v5.0/networking/af_xdp.html, 2022

# Speed advisor for buses to cross signaled intersection via V2I communication

Feng Xie [1], Viju Sudhi[1], Tim Ruß[1] and Arne Purschwitz[2]

**Abstract:** Due to a rising pressure of traffic jams and environment pollution, improving the traffic efficiency plays undoubtedly a critical role in modern intelligent transport system. Obviously, stop-and-go behaviors at intersections waste a large amount of energy and reduce traffic efficiency. Therefore, this paper proposes a speed advisory system to help buses adapt speeds from the closest bus stop to the intersection, aiming to go through intersections without any stop. Different from typical GLOSA (Green Light Optimal Speed Advisory) for private vehicles, which provides statistically several fixed static speed modes, the proposed GLOSA for buses will calculate appropriate speed ranges every second. For validation, 20 groups of GLOSA and non-GLOSA buses are simulated respectively based on an intersection in Hamburg. The results show that these GLOSA buses can save waiting time at the intersection by 98.95%.

**Keywords:** V2I communication, public transport, GLOSA

## 1 Introduction

Nowadays, Vehicle-to-Infrastructure (V2I) communication plays an important role in Cooperative Intelligent Transport Systems (C-ITS), which is constructed with Vehicle AdHoc NETworks [KFC19]. As an important application of V2I communication, Green Light Optimal Speed Advisory (GLOSA) systems can firstly acquire information such as current position and signal plans via broadcasts of RSUs (Road Side Units). Normally, the messages in intelligent transport systems are broadcasted in forms of SPAT (Signal Phase and Timing Message), DENM (Decentralized Environmental Notification Message), CAM (Cooperative Awareness Message), MAP (Map Data Message), SREM (Signal Request Extended Message), SSEM (Signal Request Status Extended Message) and so on [KFC18, ET16]. Secondly some data of vehicles will be processed, such as residue time to the upcoming green signal, distance to the intersection, and current velocity. Finally, the speed advisor can tell the driver an optimal crossing speed to go through the intersection without any stop. It has been proven that GLOSA plays a crucial role in the reduction of CO2 and fuel consumption [SIC19, EHG13, Ma19], by means of reducing travel time and waiting time at intersections [XNC19]. According to the algorithms, GLOSA systems can be categorized into single-segment and multi-segment systems[SM20].

[1] ifak - Institut für Automation und Kommunikation e.V., Verkehr und Assistenz, Werner-Heisenberg-Str.1, 39106 Magdeburg, Feng.Xie@ifak.eu
[2] Hamburg Verkehrsanlagen GmbH, arne.purschwitz@hhva.de

For single-segment approaches, Suramardhana et al. proposed six mobility models with different levels of accelerations and decelerations for a driver-centric GLOSA [SJ14]. The examination experiments of this GLOSA system proved that the waiting time could be reduced by 23.9% on average [SJ14]. Considering platoon-cases, Stebbins et al. raised up a GLOSA algorithm integrated with trajectory functions, which could be further used for autonomous driving [St17]. After evaluation, applied in this situation, the GLOSA system could save time by 30~50% and reduce fuel consumption by 15~20% [St17]. From an innovative point of view, Suzuki et al. designed a new indication of recommended speed by presenting GO and NOGO rectangular bars on the road by means of head-up display (HUD) [SM18]. This GLOSA system was realized by visual technologies, focused on available distances in the corresponding signal phases, which can help drivers keep appropriate speeds in an easier way. It has a high potential to reduce CO2 emissions and improve fuel efficiency by avoiding inappropriate decelerations, however, the travel time would not change significantly [SM18].

From an overall view of a whole route, which is normally divided into several segments by traffic signals in series, Seredynski et al. put forward a multi-segment GLOSA to minimize the total fuel consumption or travel time [SMK13]. Compared with single-segment GLOSA in off-peak hours, this innovative multi-segment method was proved to be a better solution with significant advantages [SMK13]. Considering different energy consumption models of electric vehicles, Simchon et al. developed a dynamic GLOSA system for e-vehicles, which could be implemented for real-time cases [SR20]. Taking into account the trade-off between the travel time and energy consumption, an optimal speed planning could be generated by the optimization calculation model. By means of simulation, this dynamic GLOSA for e-vehicles could result in a reduction of energy consumption by 50% and save more than 6% of the travel time [SR20]. Combined with autonomous driving and multi-segment algorithms, a novel R-GLOSA system was put forward by Nguyen et al., applying the communication between RSUs [Ng16]. Compared with traditional single-segment and multi-segment approaches, the proposed R-GLOSA was proved to outperform in terms of travel time and fuel efficiency [Ng16].

As one of the main modes of transportation, public transport attracts a lot of attention. Especially for buses where routes are settled and schedules are strict, there is a high demand for a GLOSA system to improve fuel efficiency and reduce waiting time at intersections. Generally, in order to maintain service regularity, holding and stop skipping are adopted by operators [LSV18]. Combining a holding criterion and speed advisory, a kind of hybrid controller for buses was raised by Laskaris et al. [LSV19], namely a combination of GLOSA and GLODTA (Green Light Optimal Dwell Time Advisory). But the limitation was that all signal programs have to be fixed time. Later, the authors further enhanced the bus holding control by means of the communication of C-ITS [LSV20]. It is the first research to study the combination of GLODTA, GLOSA, and TSP (Transit Signal Priority) at a line level [LSV20]. The results of the evaluation and simulation of the selected bus line in Luxembourg presented that this controller could cut down significantly the requests for TSP to keep regularity, and meanwhile, avoid influences on other traffic flows [LSV20]. Colombaroni et al. [CFI20] designed a GLOSA system taking into

account the TSP of buses, then simulated this model-based method for a tram line in Rome. After simulations of different bus priority rules, this GLOSA system was proved to be able to not only reduce the delay of bus riders, but also improve the whole efficiency of the total traffic [CFI20].

In the previous research, GLOSA systems were designed for private vehicles, neglecting the dwell time of buses at stops. The typical driver-centric GLOSA usually adopts several concrete accelerations or decelerations to represent diverse driving modes, which could be different from the practical cases for all vehicles [SJ14]. In many cases, once the speed recommendation is provided, the calculation might not be adjusted in real-time according to the actual velocity [SM19, Lu17]. Generally, in most conventional GLOSA systems, an optimal passing speed is only provided, which is difficult for drivers to maintain strictly and exactly [SM19].

As a result of the limitations of GLOSA for buses (GLOSA-Bs) in present research, this paper will design a novel speed advisor, considering both the waiting time at bus stops and optimal speed to pass through the signaled intersections. Via communication with the C-ITS in real time, this novel GLOSA-B will firstly tell the bus driver an appropriate waiting time at the stop to close the door and departure. Then when the bus leaves the stop, a recommended range of driving speed will be provided instead of a concrete index, in order to help the driver manage the speed more flexibly and easily. This proposed GLOSA-B will update the optimal range of speed every second, which could therefore, simplify largely the algorithm of GLOSA with a neglect of actual acceleration or deceleration. Because if the driver is aware of the difference between the practical speed and optimal range, then the driver will adapt driving behavior flexibly instead of obeying a defined acceleration. Different from typical GLOSA-Bs, this dynamic adapted GLOSA-B will focus only on the closest stop to the intersection rather than at a line level, in order to reduce the cost of communication in the whole traffic network.

The rest of this paper is structured as follows. Section 2 introduces the applied communication mechanism in ITS. Section 3 formulates the GLOSA-B algorithm. Furthermore, in order to evaluate the algorithm, a simulation and comparison of both GLOSA and non-GLOSA buses is proceeded in Section 4. Related conclusions are in Section 5.

## 2 Communication Architecture in ITS

There are many wireless technologies that could be used to realize the communication between several vehicles as well as between vehicles and infrastructure. V2X via IEEE 802.11p has become mature, new specification releases are frequently tested and, maybe most importantly, there are several 11p ready devices on the market that can be utilized in custom use cases. This technology and all corresponding specifications are also called ITS-G5.

In this work, standardized messages MAPEM and SPATEM (European versions of MAP and SPAT) are adopted for the GLOSA-B. Two RSUs have been installed in the test field to send SPATEM: one is for the precise traffic light status (current color), another one is for the precise forecast of the color switching time. The OBU is responsible for merging these data.

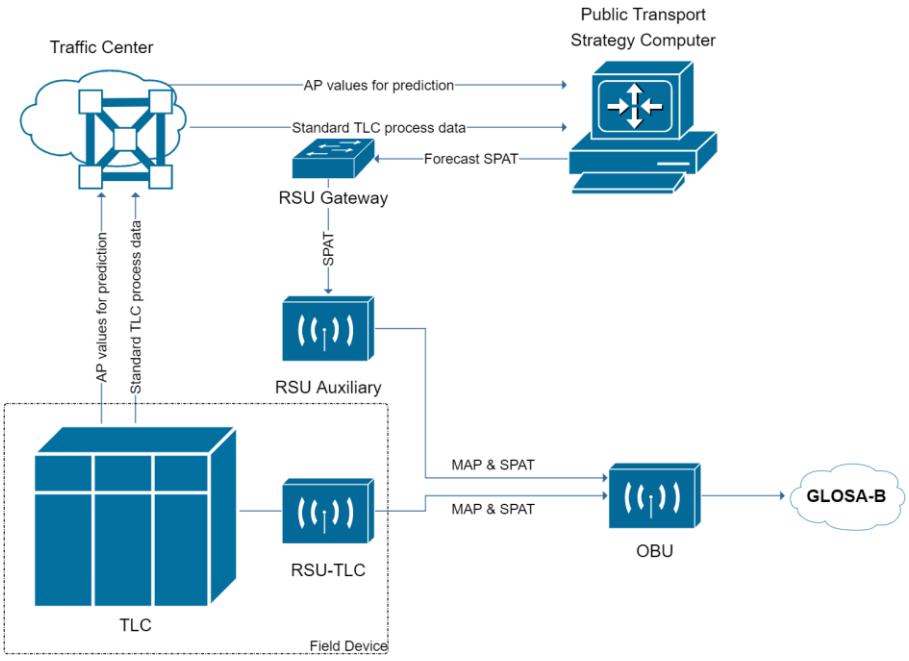



Fig. 1: V2I communication in use case

The communication mechanism of the dynamic adapted GLOSA-B in this paper is structured in Fig. 1. As presented in the architecture, the field device is constructed by the TLC (Traffic Light Controller) connecting with a RSU, which is called RSU-TLC in this

work. The TLC will firstly send related application values with some standard TLC process data to the traffic center, then the traffic center will transfer the required data to the public transport strategy computer as inputs for further process. After computing, a forecast SPAT message will be generated and transferred to a RSU auxiliary. Finally, the current SPAT message will be sent by RSU-TLC, while the forecast SPAT message will be broadcasted by the RSU auxiliary simultaneously. An OBU (On-Board Unit) will be implemented on a bus to receive the current & forecast SPAT, and MAP messages of target intersections. After merging and processing, the OBU will send required data to the GLOSA-B system.

## 3 Algorithm interpretation of GLOSA-B

As described in Fig. 2, via the communication with the RSUs, messages such as MAP, SPAT, and forecast SPAT could be obtained as inputs to the GLOSA-B application. Additionally, the bus route information is assumed to be available in OBUs.

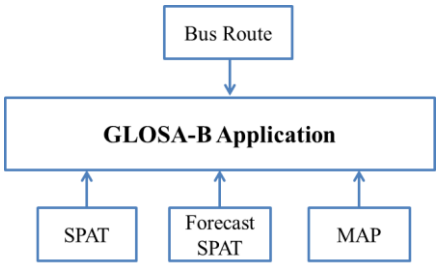



Fig. 2: Required data of GLOSA-B

The communication starts with obtaining MAP messages. Each MAP message describes the position of different lanes for different intersections. For every second, the algorithm checks which lane is the closest to the current GNSS (Global Navigation Satellite System) position (assuming it to be always available). Once the current lane is identified, the coordinate of the corresponding intersection will be found out. The signal group, connected to the lane is also retrieved. With this information, the event state corresponding to this signal group and its end times, both minimum and maximum, are retrieved from the SPAT message. When the SPAT message gives the timings of the current signal, the forecast SPAT will provide information about the overall signal cycle of the signal group.

The application starts when the bus approaches the bus stop closest to the target intersection and runs as long as the bus hasn't crossed the intersection. Despite performing the same computations, the algorithm differs according to the positions of the bus. It depends on whether the bus is at the stop or moving from the bus stop to the intersection.

**Algorithm 1** GLOSA-B algorithm

1: Find the relevant intersection $int$
2: Compute distance to $int$, $distToInt$
3: Find the relevant signal group $sg$
4: Find the current signal in $sg$, $s$
5: Set $timeToGreen$ as the time remaining for the next GREEN in $sg$
6: Compute $desiredSpeed$
7: Set $currSpeed$ as the current speed (from GNSS or Vmin)
8: Refine $desiredSpeed$ (refer Algorithm 2)
9: **if** $currSpeed$ is in range of $desiredSpeed$ **then**
10: Proceed
11: **else if** $currSpeed < desiredSpeed$ **then**
12: Accelerate
13: **else if** $currSpeed > desiredSpeed$ **then**
14: Decelerate

Since the relevant intersection and signal group are obtained from the MAP message, distance from the current position to the intersection is computed by means of the bus route data. As described in Algorithm 1, the remaining time for the next GREEN is computed from the signal cycles given by forecast SPAT. The current speed is obtained from the GNSS message if the bus is moving. Otherwise, it is assumed to be the minimum permissible speed if the bus is at the bus stop. As presented in (1), the desired speed is computed considering the distance to the intersection and the remaining time for the next GREEN, where the time needed to close the doors for passengers to board and deboard is assumed to be 10s.

$$desired\ speed = \frac{distance\ to\ intersection}{time\ to\ GREEN - time\ to\ close\ doors} \quad (1)$$

However, there could be cases when the desired speed is not in the permissible range and providing this speed advisory shall prove infeasible considering the real-world application of GLOSA. This demands a second phase of computation, when the desired speed is further refined and brought under the permissible bounds. As defined in Fig. 3, *Vmin* and *Vmax* are the minimum and maximum permissible speeds, *V1* and *V2* are the minimum and maximum limit of the computed desired speed. If the desired speed is not within the range of (*Vmin*, *Vmax*), the non-adhering bound(s) of the desired speed is capped to the permissible bound(s), which is shown in Fig. 3 (b) ~ (d). If the desired speed is much lower than *Vmin*, it implies the bus has to wait at the bus stop for a period of time, at least until *V2* is greater than *Vmin*. Similarly, when the desired speed is much greater than *Vmax*, the algorithm will prompt the driver to wait at the bus stop as long as *V1* is less than *Vmax*. In both cases, shown in Fig. 3 (e) and (f), the desired speed is re-computed based on the next possible signal cycle. These steps are detailed in Algorithm 2, where lines 2~4 and 10~12 apply only when the bus is at the bus stop. When the bus is moving from the bus stop to the intersection, the speed computation will not include waiting periods.

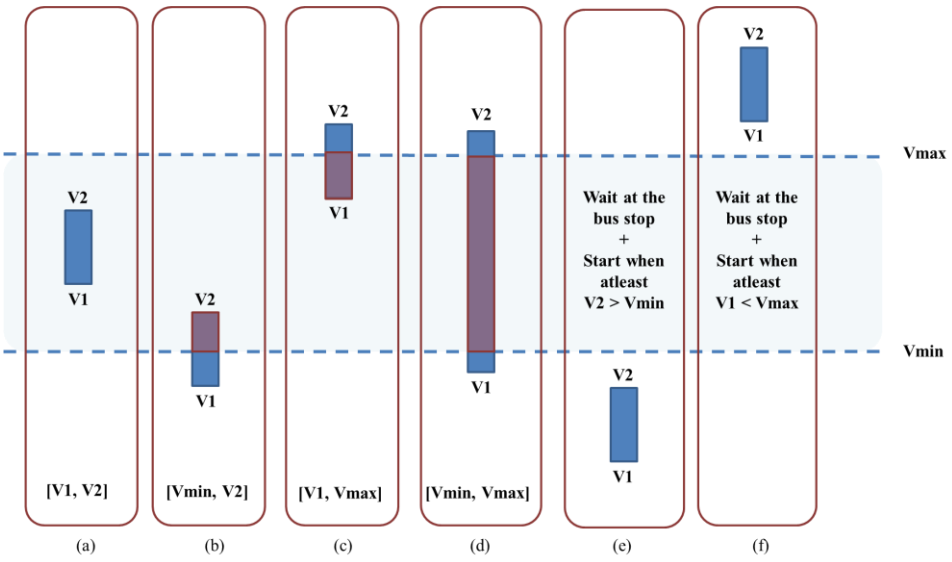



Fig. 3: Scenarios of computing and refining desired speed ranges. Desired speed is within the permissible speed range in (a). Capping the speed at *Vmin* and/or *Vmax* is necessary for (b) ~ (d), the shaded region is the re-computed desired speed. Waiting at the bus stop and checking the next cycle of the signal is necessary for (e) and (f)

**Algorithm 2** Refine desiredSpeed

**Require:** desiredSpeed (V1, V2) is not in range (Vmin, Vmax)

**if** $V1$ <Vmin **then**
  **if** $V2$ <Vmin **then**
    Check for the next GREEN in $sg$
    Wait at the bus stop, at-least until V2 > Vmin
  **else if** $V2$ is in range (Vmin, Vmax) **then**
    Set $desiredSpeed$ as (Vmin, V2)
  **else**
    Set $desiredSpeed$ as (Vmin, Vmax)
**if** $V2$ >Vmax **then**
  **if** $V1$ >Vmax **then**
    Check for the next GREEN in $sg$
    Wait at the bus stop, at-least until V1 < Vmax
  **else if** $V1$ is in range (Vmin, Vmax) **then**
    Set $desiredSpeed$ as (V1, Vmax)
  **else**
    Set $desiredSpeed$ as (Vmin, Vmax)

Once the desired speed is refined, the speed advisory can be provided in terms of

a. ‘Proceed’ if the current speed is in the range of desired speed.
b. ‘Accelerate’ if the current speed is less than the desired speed.
c. ‘Decelerate’ if the current speed is greater than the desired speed.

For Case b and c, the bus driver should also be notified of the desired speed range and the current speed, with which the bus can go through the intersection in GREEN time.

## 4 Simulation and comparison of GLOSA and non-GLOSA buses

To validate the proposed GLOSA-B algorithm, a simulation setup was developed and experimented with 20 different runs of GLOSA and non-GLOSA buses through the route of Bus 26 in Hamburg, the route with coordinate-points is shown in Fig. 4. The coordinate of the bus stop is (53.6025272, 10.1277754) and the intersection is located at (53.5998134, 10.1298669). The distance from the starting point of the trip (53.6034413, 10.1271376) to the intersection is approximately 432m and the distance from the bus stop to the intersection is about 352m.


Fig. 4: Intersection for simulation

Considering the actual case and German traffic rules, the permissible speeds for the buses are set to be *Vmin* = 5 m/s and *Vmax* = 13.89 m/s. As depicted by Fig. 5, the concerned signal group ‘K1’ at the intersection is assumed for the simulation to be cyclic in the received SPAT messages, where GREEN lasts for the first 55 seconds, then YELLOW for 3 seconds, followed by 32 seconds of RED. One signal cycle lasts for 90 seconds and it repeats in the same order throughout the simulation. The starting times for simulation were selected randomly and uniformly in the whole signal program.

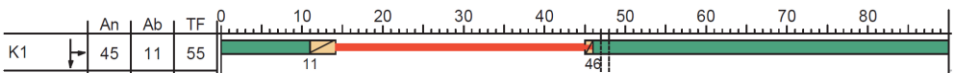
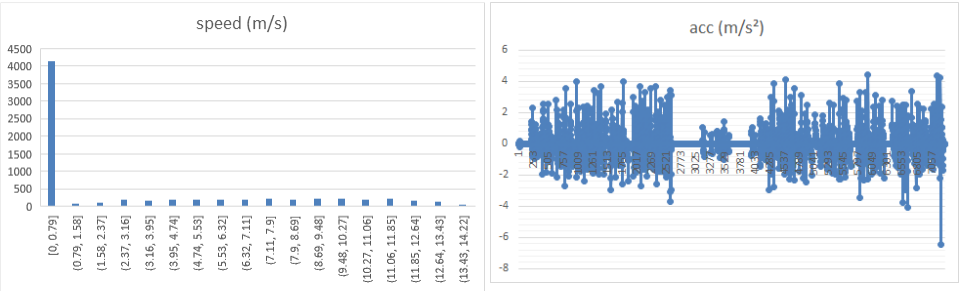



Fig. 5: Signal plan for K1

It is also assumed that the necessary inputs to the GLOSA-B algorithm are fed every second of the simulation without fail. This includes MAP, SPAT, forecast SPAT and GNSS from the RSUs and bus route information in the OBU.

For simulation, a GPS logger was implemented on a vehicle following the Bus 26 to record the speeds, which are shown in Fig. 6 (a). Excluding the most values below 0.79 m/s, a speed of 7.5 m/s is selected as a reachable medium speed for simulation with a high occurrence rate. Then the acceleration data is generated in Fig. 6 (b). After analysis, a group of reachable and acceptable parameters are selected for GLOSA and non-GLOSA buses, which are presented in Table 1.




(a) (b)

Fig. 6: Speed record (a) and acceleration data (b) of Bus 26 in Hamburg

| **Sorts of buses** | **Parameters** |
|---|---|
| GLOSA bus | adjusts according to the speed advisory<br>gradual acceleration < 3.5 *m/s²*<br>gradual deceleration < 2.5 *m/s²* |
| Non-GLOSA bus | does not follow the speed advisory<br>maximum velocity of 7.5 *m/s*<br>constant acceleration at 3.5 *m/s²*<br>constant deceleration at 2.5 *m/s²* |

Tab. 1: Precondition of simulated buses

### 4.1 Simulation results

It was observed, as shown in Fig. 7, the GLOSA buses took an average of 43 seconds to cross the intersection while non-GLOSA buses took around 54 seconds on average. There is no surprise that the non-GLOSA buses had to wait at the intersection for around 9.5

seconds on average for the GREEN signal. This amounts to about 10.6% of the total signal cycle time. As shown in Fig. 8, only 2 out of 20 GLOSA buses reached the intersection with 1 second remaining for GREEN, all the other buses reached the intersection when the signal had already been GREEN. This tiny error could be eliminated simply by narrowing down the recommended speed range.

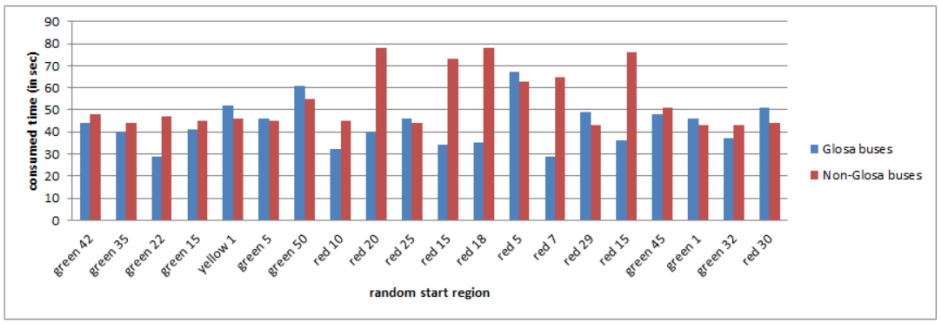



Fig. 7: Crossing time consumed by simulation buses

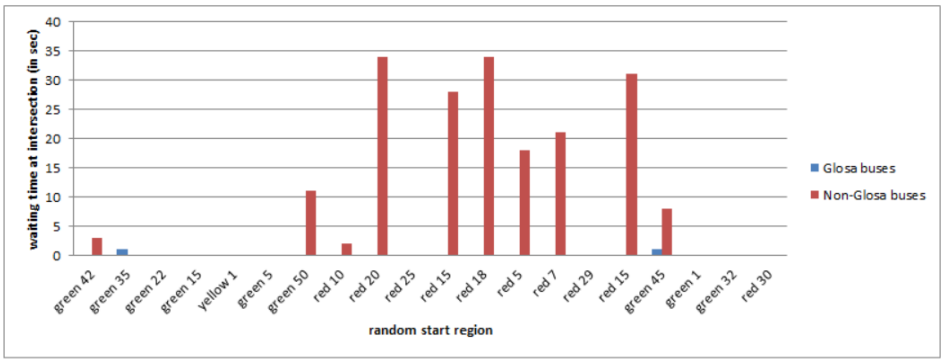



Fig. 8: Waiting time of simulation buses at the target intersection

## 5 Conclusion

Based on existing GLOSA systems for vehicles, this paper presents a GLOSA-B algorithm, considering the waiting time of buses at the closest stop to the intersection. By means of communication with C-ITS, GLOSA-B can obtain the SPAT, forecast SPAT, MAP, and further route information as inputs. According to the starting and ending time of GREEN or next GREENs, the algorithm will calculate possible time to close the doors, and appropriate speed range to guarantee the bus goes through the intersection without stopping. After simulation and comparison of GLOSA and non-GLOSA buses, the results

show that, only 2 out of 20 GLOSA buses reached the intersection 1 second earlier than the start of the next GREEN signal. With the assistance of GLOSA-B, it saves 98.95% waiting time at the intersection, which could be adapted to 100% by only adjusting the recommended speed range slightly.

The proposed GLOSA-B in this paper was proved to be able to not only remind the bus driver of closing time of doors at the bus stop, but also help avoid unnecessary stops at the traffic signal at a large level. Therefore, it is obvious that GLOSA-B can reduce the energy consumption successfully. Compared with non-GLOSA buses, the travel time of GLOSA buses from the bus stop to the intersection was also reduced on average. For further validation, the GLOSA-B system is expected to be implemented on real buses and put into use in practical conditions, considering actual random cases like traffic jams. However, the influence of GLOSA-B on the rest traffic (e. g. private cars) is still not clear, a further research needs to be proceeded to adapt the GLOSA-B for all traffic flows. But there is a fact that due to the development of IoT technologies in ITS, GLOSA will be more widely applicated for all traffic participants (i. e. private cars, pedestrians, cyclists etc.). Therefore, in this case, the adverse impact of GLOSA-B on the rest traffic will be eliminated in the future transportation network.

Literaturverzeichnis


[KFC19] Karoui, M.; Freitas, A.; Chalhoub, G.: Comparative Evaluation Study of GLOSA Approaches Under Realistic Scenario Conditions. In: Palattella M., Scanzio S., Coleri Ergen S. (eds) Ad-Hoc, Mobile, and Wireless Networks. ADHOC-NOW 2019. Lecture Notes in Computer Science, Springer, Cham, vol. 11803, pp. 407-419, 2019.

[KFC18] Karoui, M.; Freitas, A.; Chalhoub, G.: Efficiency of Speed Advisory Boundary fINder (SABIN) strategy for GLOSA using ITS-G5. In: 7th IFIP/IEEE International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN). Toulouse, France, pp. 1-6, 2018.

[ET16] ETSI: TS 103 301-Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Facilities layer protocols and communication requirements for infrastructure services. No. v1.1.1, 2016.

[SIC19] Sharara, M.; Ibrahim, M.; Chalhoub, G.: Impact of Network Performance on GLOSA. In: 16th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, pp. 1-6, 2019.

[EHG13] Eckhoff, D.; Halmos, B.; German, R.: Potentials and limitations of Green Light Optimal Speed Advisory systems. In: IEEE Vehicular Networking Conference, Boston, MA, pp. 103-110, 2013.

[Ma19] Marumo, Y.; Yamazaki, K.; Suzuki, H.; Michitsuji, Y.: Driver Assistance System at Signalized Intersection by Indicating Predicted Signal Aspects on Road. In: Proc. of the 5th Int. Symp. on Future Active Safety Technology toward Zero accidents (FAST-zero'19), Blacksburg, VA, USA, pp. 1-4, 2019.

[XNC19] Xie, F.; Naumann, S.; Czogalla, O.: Speed Control System for Pedestrians Crossing Signaled Intersections Time Optimally. IFAC-PapersOnLine, 52(24), pp. 82-87, 2019.

[SM20] Suzuki, H.; Marumo, Y.: Safety Evaluation of Green Light Optimal Speed Advisory (GLOSA) System in Real-World Signalized Intersection. Journal of Robotics and Mechatronics, 32(3), pp. 598-604, 2020.

[SJ14] Suramardhana, T. A.; Jeong, H. Y.: A driver-centric green light optimal speed advisory (DC-GLOSA) for improving road traffic congestion at urban intersections. In: 2014 IEEE Asia Pacific Conference on Wireless and Mobile, IEEE, pp. 304-309, 2014.

[St17] Stebbins, S.; Hickman, M.; Kim, J.; Vu, H. L.: Characterising green light optimal speed advisory trajectories for platoon-based optimisation. Transportation Research Part C: Emerging Technologies, 82, pp. 43-62, 2017.

[SM18] Suzuki, H.; Marumo, Y.: A New Approach to Green Light Optimal Speed Advisory (GLOSA) Systems for High-Density Traffic Flowe. In: 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 362-367, IEEE, 2018.

[SMK13] Seredynski, M.; Mazurczyk, W.; Khadraoui, D.: Multi-segment Green Light Optimal Speed Advisory. In: 2013 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum, Cambridge, MA, pp. 459-465, 2013.

[SR20] Simchon, L.; Rabinovici, R.: Real-Time Implementation of Green Light Optimal Speed Advisory for Electric Vehicles. Vehicles, 2, pp. 35-54, 2020.

[Ng16] Nguyen, V.; Kim, O. T. T.; Dang, T. N.; Moon, S. I.; Hong, C. S.: An efficient and reliable Green Light Optimal Speed Advisory system for autonomous cars. In: 18th Asia-Pacific Network Operations and Management Symposium (APNOMS), pp. 1-4, IEEE, Oct. 2016.

[LSV18] Laskaris, G.; Seredynski, M.; Viti, F.: Improving Public Transport Service Regularity using Cooperative Driver Advisory Systems, 2018.

[LSV19] Laskaris, G.; Seredynski, M.; Viti, F.: A real time hybrid controller for regulating bus operations and reducing stops at signals. In: 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), Cracow, Poland, pp. 1-7, 2019.

[LSV20] Laskaris, G.; Seredynski, M.; Viti, F.: Enhancing Bus Holding Control Using Cooperative ITS. In: IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 4, pp. 1767-1778, Apr. 2020.

[CFI20] Colombaroni, C.; Fusco, G.; Isaenko, N.: A simulation-optimization method for signal synchronization with bus priority and driver speed advisory to connected vehicles. Transportation research procedia, 45, pp. 890-897, 2020.

[SM19] Suzuki, H.; Marumo, Y.: A New Approach to Green Light Optimal Speed Advisory (GLOSA) Systems and Its Limitations in Traffic Flows. In: Ahram, T.; Karwowski, W.; Taiar, R. (eds) Human Systems Engineering and Design. IHSED 2018. Advances in Intelligent Systems and Computing, vol 876. Springer, Cham, 2019.

[Lu17] Luo, Y.; Li, S.; Zhang, S.; Qin, Z.; Li, K.: Green light optimal speed advisory for hybrid electric vehicles. Mechanical Systems and Signal Processing, 87, 30-44, 2017.

# Concept for Controller-Based Coexistence Management of Diverse Wireless Communication Systems

Daniel Antonow[1], André Gnad[2], Tobias Ferfers[3], Henning Trsek[1]

**Abstract:** Modern production facilities are increasingly optimising production performance and quality. This results in the need for real-time access to interconnected entities to enable the organisation and control of the entire production chain in industrial environments. Wireless communication systems are widely used to connect different machines, e. g. moving parts, which normally do not allow wired connections. The increasing use of diverse wireless systems will result in more devices using the limited shared wireless medium, leading to interference between involved wireless entities. To ensure coexistence, i. e. meeting the communication and functional requirements of all applications, coexistence management is essential. Therefore, this paper presents a new concept for automatic controller-based coexistence management, which considers three dimensions (i. e. frequency, time and location) of wireless communication. This leads to an increased number of applications being able to communicate in parallel, while still meeting their requirements.

**Keywords:** Coexistence Management; Wireless Communication Systems; Wireless Synchronization; Spectrum Monitoring; Wireless Signal Identification

## 1 Introduction

High demands on production output and product quality are leading to newly developed Industry 4.0 concepts for organizing and controlling the entire value chain over the life cycle of products. These concepts require the availability of all relevant information in real time, by interconnecting all entities involved. Therefore, automation applications require highly available, deterministic and low latency communication. In addition, highly accurate timing information has to be provided frequently to such applications. A key component of interconnecting future automation systems will be wireless communication. As a result, its use will grow despite the difficult conditions for radio propagation in industrial facilities [VD19]. Additionally, challenges for the parallel operation of wireless applications using the same shared medium, such as license-free frequency ranges that are allocated in a technology-neutral manner (e. g. Industrial-Scientific-Medical (ISM) radio bands), always arise. To solve the problems outlined above, interference-free parallel operation of wireless applications is necessary. As many applications as possible should use the limited wireless

[1] inIT – Institute Industrial IT, Ostwestfalen-Lippe University of Applied Sciences and Arts, Campusallee 12, 32657 Lemgo, Germany, daniel.antonow@th-owl.de, henning.trsek@th-owl.de
[2] ifak – Institut für Automation und Kommunikation e.V., Werner-Heisenberg-Straße 1, 39106 Magdeburg, Germany, andre.gnad@ifak.eu
[3] Fraunhofer IOSB-INA, Campusallee 1, 32657 Lemgo, Germany, tobias.ferfers@iosb-ina.fraunhofer.de

spectrum without interference. Communication requirements and the frequency channel can change dynamically, but the requirements for availability, determinism and synchronization must always be met. This condition is referred to as coexistence [IE22a]. In order to deal with maintaining interference-free operation, an automated, dynamic and efficient coexistence management is required. Therefore, this paper introduces a new concept for controller-based coexistence management of diverse wireless systems considering spectral, spatial and temporal aspects of wireless communication.

## 2 Use Cases and Requirements

In the following section generally described use cases are presented that are used to derive requirements for automatic coexistence management. Subsequently, the derived requirements are summarized in Table 1.

*Use Case A: Initial Deployment of a Wireless System*

A system integrator wants to perform an initial deployment of a wireless networked system into an existing wireless environment managed by a coexistence management system. When integrating new wireless applications into an existing controlled environment, the new wireless system should not cause interference with existing systems under any circumstances, in order to avoid communication downtime of other systems. Therefore, for the deployment process, the system integrator must be able to **R.1:** register the new wireless system with the coexistence management (e. g. Database entry), thereby enabling it to **R.2:** perform an autonomous configuration (e. g. frequency channel, transmission interval, signal strength) of the device and to **R.3:** integrate it into the existing environment without any interference. The particular configuration depends on the current coexistence conditions. Integration is complete when no interference has been detected as a result of the new wireless system and all functional constrains are met.

*Use Case B: Occurring Interference due to a Non-Managed Wireless System*

A non-managed and thus unknown wireless system, here acting as an interferer, transmits in a wireless environment which is actively monitored by a coexistence management system. With regard to the transmission behavior of the unknown wireless system, interference can occur because the interferer cannot be configured by the coexistence management. An automatic coexistence management should be able to **R.4:** detect the unknown wireless system on the basis of the continuously monitored medium and the analysis of its transmission behavior. Furthermore, the result of the analysis (e. g. used frequency channel, transmission intervals) and a comparison of the configurations of registered wireless systems (e. g. within a database) should lead to **R.5:** the conclusion that it is a non-managed wireless system. With regard to the fact that a configuration of the interferer is not possible, the transmission behavior of managed wireless systems affected by interference must be adapted in order to **R.6:** minimize any interference that occurs and subsequently ensuring coexistence.

*Use Case C: Reliable Parallel Operation of Different Wireless Technologies*

An automated guided vehicle (AGV) is deployed in a production plant in order to be used for the transport of loads. The AGV communicates via WLAN to receive corresponding transport orders and is initially registered by the administrator with the central coexistence management (refer to Use Case A). Thus, **R.7:** the AGV belongs to the managed wireless systems. After receiving a transport order, the AGV passes production lines that are equipped with local Bluetooth systems which are also managed. With regard to the shared frequencies of the 2.4 GHz ISM band, mutual interference between the two wireless technologies cannot be ruled out. A coexistence management system has to be able to **R.8:** detect any interference that occurs, in analogy to Use Case B, and initiate appropriate measures to ensure coexistence. Considering that the operation of production lines is of greater importance, it should be possible to **R.9:** regulate the availability of managed wireless systems depending on their functional conditions (i. e. prioritization). This means that the AGV changes its current frequency channel and/or is assigned fewer transmission resources (e. g. longer transmission intervals) in order to minimize the influence on the higher-priority Bluetooth system. Furthermore, the coexistence management should be able to **R.10:** regulate functionalities of the applications to ensure continued operation with fewer resources. For example, the AGV could reduce its driving speed on the instruction of the coexistence management in order to be able to react quickly enough to new transport orders. As soon as the AGV is out of range of the production lines, the allocated resources are adjusted again. In case of an equally demanded priority of several mutually influencing systems, the transmission resources and/or functionalities of all involved wireless systems have to be adjusted accordingly, so that **R.11:** all functional conditions are fulfilled. If functional conditions cannot be met by automatic control of coexistence, the coexistence management system has to **R.12:** issue a warning in any case, so that the operator has the opportunity to perform supplementary problem-solving measures. This is particularly important for safety-relevant applications and applications that cause costs in the event of failure.

| Requirement | Description |
|---|---|
| R.1 | Registration of wireless systems within coexistence management |
| R.2 | Autonomous configuration of managed wireless systems |
| R.3 | Automatic integration of new wireless systems |
| R.4 | Continuous spectral monitoring of coexistence state |
| R.5 | Automatic identification of non-managed wireless systems |
| R.6 | Solve or minimize occurring interference |
| R.7 | Handling of mobile wireless systems |
| R.8 | Wireless technology independent spectral monitoring |
| R.9 | Configurable prioritization of wireless systems |
| R.10 | Automatic configuration of application specific functionalities |
| R.11 | Monitoring of met communication and functional requirements |
| R.12 | Automated warning, if application requirements cannot be met |

Tab. 1: Summary of derived requirements for an automated coexistence management.

# 3 State of the Art

The aim of establishing coexistence is to ensure the interference-free operation of wireless applications in their environment. From a technical point of view, wireless systems can coexist by considering four dimensions, namely frequency, time, space and polarization. Separation in the domains of frequency and time usually causes the least loss of performance and initially also the lowest economic costs. Some wireless technologies provide adaptive media access mechanisms for this type of coexistence management, such as Listen Before Talk (LBT), Dynamic Frequency Selection (DFS) or Adaptive Frequency Hopping (AFH). However, these methods often lead to non-deterministic time behavior, which is essential for industrial applications. Spatial separation is rarely possible with wireless applications; radio propagation can only be spatially restricted with great effort. In addition, wireless communication systems are frequently used in mobile applications, which makes static spatial planning almost impossible. The effect of polarization separation is relatively small inside buildings or in other highly reflective environments [VD19].

**Clock synchronization** is essential in today's connected world and serves two main purposes. Firstly, clock synchronisation is needed for scheduling, e. g. when a wireless network access point and the station need to change their frequency. Secondly, it is applied to give different devices the same idea of an absolute time, which is useful in several applications (e. g. logging of events). In an industrial environment, wireless networks are exposed to a challenging environment due to temperature, pressure and humidity fluctuations. [SJ07] gives examples of environmental impact on clock synchronisation via the Precision Time Protocol (PTP) defined in [IE20]. In addition, alternating bandwidths, error rates and asymmetric communication links as well as wireless interference are challenging and influence the end-to-end delay. Latency and bandwidth are the two most important network parameters for precise network time synchronisation. [Pu20] gives an overview of several wired and wireless clock synchronization methods and protocols in current research, while also considering established protocols (e. g. NTP, PTP, Reference Broadcast Synchronization). [Pu20] defines three main criteria for clock synchronization, which are the algorithm, the achievable accuracy and the experimental conditions. Furthermore, they state that a synchronization accuracy of less than 1 µs is not achievable without special hardware and suggest implementing clock synchronization algorithms as software, if it meets the synchronization accuracy and other requirements of the application. Exemplary implementations for wireless clock synchronization are shown in [Pu20, Ma16, Ca05, Ce15], comprising currently researched and established synchronization protocols. However, since these implementations are usually restricted to the use of special hardware or software, no general solution for wireless and wired networks currently exists. Considering already existing standards and exemplary implementations, the following components can be identified. A clock synchronization protocol needs a defined state machine to determine its behavior. This includes the start / stop states and the behaviour to disconnects or signal loss. Furthermore, structure of packets or synchronization messages have to be defined. Additionally, specification of media-dependent or media-independent mechanism has to

be done. The synchronization method needs to include transmission parameters like send intervals, timing of message exchange and the extent of its synchronization (i. e. broadcast or two-message synchronization). The most important component for clock synchronization is to have access to a clock, which is adjustable by a so called servo algorithm such as PI-controller or linear regression.

**Wireless spectrum information** is required for the management of the frequency dimension. To sample spectral information, measurement hardware must be used, such as Software Defined Radios (SDRs), which can capture high-resolution In-Phase and Quadrature (IQ) raw data from frequencies of interest. Although SDRs offer high computational resources for data acquisition, most wireless devices have limited measurement capabilities. These capture the raw spectral data in form of low-resolution Received Signal Strength Indicator (RSSI) in order to detect other wireless devices. Furthermore, a suitable algorithm for signal identification is required that automatically recognises network participants regardless of their respective wireless communication technology. In addition, signals that cause interference in the present medium must be identified. Subsequently, an analysis of the transmission behaviour of the existing signals can be used to ensure optimal coexistence while avoiding interference. In [SBM17], a Convolutional Neural Network (CNN) and a Neuro Fuzzy Signal Classifier (NFSC) are compared in terms of their respective signal identification capabilities, showing that the CNN outperforms the NFSC. Since the proposed CNN by [SBM17] is only capable of identifying a single signal and the associated wireless technology used in each sample, [GBM18] adapted the CNN to identify multiple signals. [SBM17, Ku18] stated that, an applied Fast Fourier Transformation (FFT) to samples prior to training and classification promised better accuracy results compared to raw IQ samples. Furthermore, [Zh19] introduced optimization techniques for the proposed CNN of [SBM17], resulting in reduced training time while also slightly decreasing accuracy.

**Coexistence management** is basically described in [IE22a]. Currently, only manual and semi-automatic methods are used in industry to achieve coexistence between different types of wireless communication systems operating in parallel. Existing automatic procedures are only able to obtain an optimal coexistence state between wireless systems using the same specific communication technology without considering other technologies utilizing the same frequencies. Regarding resource allocation, almost exclusively spectral resources have been used so far. A dynamic consideration of spatial resources was proposed in [WM18]. First approaches to exploit the temporal resource can be found in [Sc17] and [SZ18].

## 4 Controller-Based Coexistence Management Concept

The presented concept proposes an automatic coexistence management for wireless applications which use the 2.4 GHz ISM band for communication, but is also applicable for other frequency bands. The frequency range considered is available for free use and currently utilized by many different wireless technologies, therefore being widely incorporated into industrial processes. This results in heterogeneous networks with diverse wireless systems

operating on these frequencies, while also communicating in parallel. Therefore, interference between applications cannot be excluded.

The described REBAKO concept is based on the IEC 62657-4 standard entitled “Coexistence management with central coordination of radio application” [IE22b], which was issued in May 2022. Fig. 1 shows the entities of the REBAKO system and their communication relations. The coexistence management system is essentially composed of several subsystems. These consist of the central coexistence coordinator, spectrum sensors and a database that stores information on communication status, configuration data and local regulatory information. In addition, REBAKO modules are introduced to act as interfaces between subsystems and wireless devices. Furthermore, the clock synchronisation system ensures that a time-consistent data set is available. The following sections describe the shown entities and their functions in more detail.

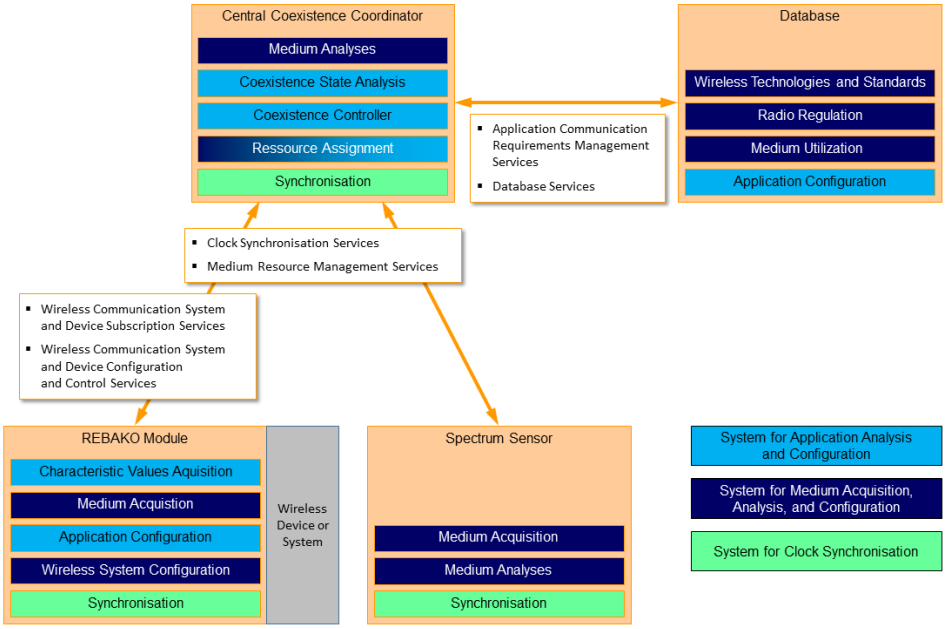



Fig. 1: Overview of the REBAKO coexistence management system concept.

## 4.1 Central Coexistence Coordinator

The central coexistence coordinator (CCC) is a unit or system for centralized management of heterogeneous wireless infrastructures for automation applications. The CCC has access to information about the status of the medium utilization and communication status of the automation applications. The central coordinator uses the obtained data about the current state of the wireless medium and other application-specific parameters, such as transmission

times, update times or error counters to evaluate the coexistence state. According to these measurements, given regulatory specifications and application requirements, the optimal configuration for the reliable functioning of all wireless applications in the operational environment is calculated in order to restore the coexistence state and subsequently ensure it in the spectral, spatial and temporal dimension. These actions result in parameters being changed within the wireless systems or applications. The parameterisation can be done via existing wireless links or via a separate wireless transmission channel. Unknown or non-manageable wireless systems cannot be influenced by the controller, but must be taken into account in the evaluation of the coexistence state. The influence of such third-party applications should be minimized as far as possible in order not to interfere with the functionalities of the devices being managed.

### 4.2 Spectrum Sensor and Medium Analysis

The current utilization of the wireless medium is measured by a SDR sensor and determined by a CNN. A deployed SDR captures the wireless traffic of the entire 2.4 GHz ISM band (i. e. 80 MHz wide) simultaneously, so that with each measurement a time consistent IQ snapshot of the relevant frequency range is created. Since the signal identification of the REBAKO system is based on Deep Learning with CNN analog to [GBM18], each snapshot is divided into eight samples corresponding to the required 10 MHz input of the implemented CNN. Before classification, a FFT is applied to each sample. Subsequently, the CNN classifies existing signals in each of the eight samples, in order to process one 80 MHz snapshot. Similarly to [GBM18], the CNN model is trained on the basis of different synthetic signal profiles. Here, a respective class represents a wireless technology and its used frequency channel. Classification results are further analysed for identifying and assigning exact wireless applications. Analysis consists of comparing transmission behaviour (e. g. used frequency channel) from each signal with entries within the REBAKO database. A wireless system is considered non-manageable if either no corresponding entry for its transmission behaviour could be found or it has been identified but does not have a configurable interface (e. g. smartphone). Furthermore, spectral interference detection is performed by comparing signal transmission profiles (e. g. frequencies, transmission intervals) and identifying overlaps.

### 4.3 Clock Synchronisation

The REBAKO system requires a synchronization of all registered network devices. This results in challenges of synchronizing a heterogeneous network environment, which includes different wireless transmission technologies and additional applications utilizing wired communication. Since there is no general solution of a clock synchronization protocol for wired and wireless network participants, the approach is to adapt an existing protocol standard to the requirements of the REBAKO network environment. The basis is the Precision

Time Protocol (PTP) [IE20] in all profiles and variants. PTP considers many scenarios like a boundary clock that synchronizes different time domains and is extended by different profiles (e. g. 802.1AS). The goal is not a complete implementation of the standardized protocol for wireless technologies, but a toolbox to apply the identified components for time synchronization from Sect. 3 to the respective requirements and capabilities of the wireless transmission technology (e. g. Bluetooth and Wi-Fi). An abstract depiction of the adapted PTP network architecture can be seen in Fig. 2. Since PTP defines one grandmaster for synchronizing the time of all connected network participants (i. e. slaves), extension are made in order to apply this basic master-slave concept via different transmission technologies. Moreover, PTP defines two port roles: A master port (M) provides its time in the network and a slave port (P) synchronizes to a master port. The idea is to group slave ports together depending on their used communication technology. A dedicated master port of each group is connected to the grandmaster (e. g. via Ethernet), which acts as the central clock of the REBAKO network. These dedicated master ports will then act as the grandmaster of their specific group or sub-network and synchronize all their connected devices, thus acting as a gateway to the main grandmaster.

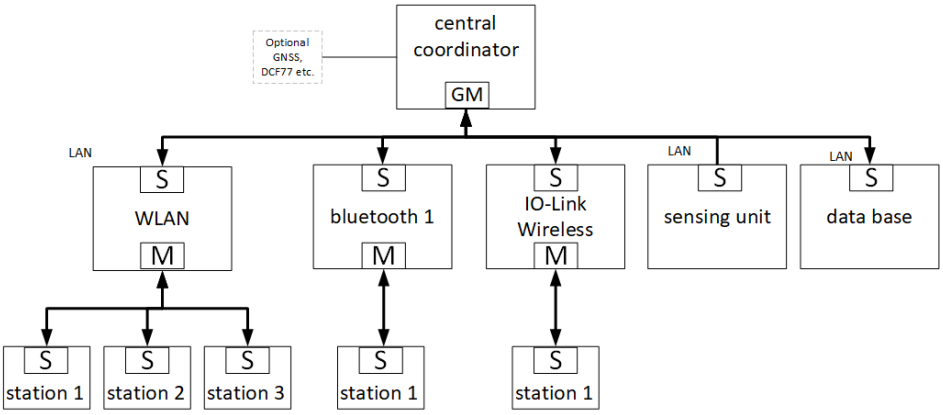



Fig. 2: Abstract depiction of the PTP protocol for the heterogeneous REBAKO system.

### 4.4 REBAKO Module

The REBAKO module operates essentially as an adapter used to connect wireless components or spectrum sensors to the central coexistence management system. With the help of this module it is possible to use standardized services for communication between the components of the REBAKO system. The implementation of the REBAKO module maps the services and protocols between the REBAKO system and the connected wireless devices. Unfortunately, the hardware and software interface to the wireless devices with application functions and their internal parameter structures are currently not standardized. Technology- and manufacturer-specific services and protocols are used for reading relevant information and

for setting application- or transmission-relevant parameters. According to the current state of technology, the implementation must be adapted for each integrated component.

### 4.5 Database

The database contains information about known wireless technologies and standards. This information supports the identification of spectrum users. In addition, detected wireless applications can be stored with additional information such as position, signal strength and other transmission properties. Furthermore, the database contains regulation information of the location of operation, such as legal requirements or regulations specified by the system operator. This allows, for example, the exclusion of frequency ranges from coexistence management. Data on medium usage is also included, as well as properties of the wireless-based applications, such as priorities.

### 4.6 Services

Data is exchanged between the components of the REBAKO coexistence management system using the services specified in [IE22b]. The following service groups are defined:

1. Application communication requirements management services
2. Wireless communication system and device subscription services
3. Wireless communication system and device configuration and control services
4. Medium resource management services
5. Database access services

The specification includes the services, attributes and their corresponding structure. Although, being independent of communication protocols, further principle communication sequences are described. Additionally, the REBAKO system includes clock synchronisation services and coexistence algorithms, which are not specified in [IE22b]. The concept defines additional services that enable event-controlled transmission if threshold values of parameters are exceeded or a warning was issued by a REBAKO module. Therefore, providing an option for outsourcing coexistence management functionalities to REBAKO modules and connected wireless devices in order to reduce heavy processing loads of the central coexistence manager caused by the computationally intensive information processing.

## 5 Demonstrator for Concept Validation and Evaluation

In REBAKO, an overall demonstrator shall be realised by the involved research institutions, which will also be used for validation of the developed concepts in realistic industrial

environments (i. e. *SmartFactoryOWL* and *Digitales Anwendungszentrum Mobilität. Logistik. Industrie.*). Components of the demonstrator are various wireless applications, as depicted in Fig. 3. These include a control application with IO-Link wireless, an AGV with WLAN communication, an application that controls signal LED's via PROFINET IO and Bluetooth as well as a unit for recording and analyzing the medium utilization. All wireless systems are using the 2.4 GHz ISM band for data transmission. The wireless access of the manageable wireless systems shall be coordinated by the REBAKO system. Vector signal generators (VSG) or SDR platforms (e. g. USRP X300), which allow the generation of defined interference signals, are also part of the demonstrator. In addition to the adaptation of wireless transmission behaviour in the dimensions of frequency and time, the possibilities by controlling the application behaviour will also be demonstrated. For example, it will be shown that an AGV can change its velocity or track accordingly when communication interference is detected.

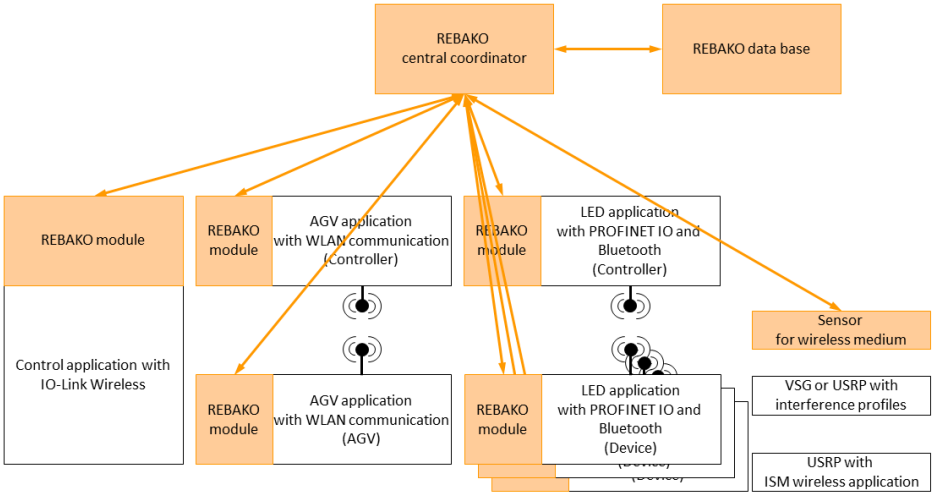



Fig. 3: Concept of the REBAKO demonstrator for validation.

## 6 Conclusion

This paper presents a new concept for controller-based coexistence management of wireless systems operating in the 2.4 GHz ISM band. To the best of our knowledge it is one of the first systems which considers three dimensions of wireless communication (i. e. spectral, spatial, temporal) for coexistence and resource allocation. Additionally, the evaluation of the current coexistence state considers application-specific parameters such as transmission times and number of lost messages. As a result, a larger number of wireless applications can meet their functional requirements while operating in parallel. Clock synchronisation via an adapted PTP protocol ensures that all network participants are synchronized regardless

of their communication technologies. Furthermore, spectral sensing is performed by SDR sensors, while signal identification and interference detection is performed with CNN by classifying SDR samples and subsequent comparison of results with wireless profiles in a database. Furthermore, this paper introduces the concept of REBAKO modules. These modules enable standardized communication between different wireless systems via defined services, such as registration with the CCC, management of medium resources and database access. Although the purpose of REBAKO modules is to connect wireless components to the coexistence management, the lack of standardization of hardware and software interfaces requires customized implementations for each integrated device. A demonstrator will be realized and deployed in an industrial environment to validate the developed concept.

## Acknowledgement

The work in this paper was carried out as part of the project “Reglerbasiertes Koexistenzmanagement verschiedenartiger Funkkommunikationssysteme” (REBAKO) IGF-Project No.: 21529 BG/2 of the German Forschungskuratorium Maschinenbau e.V. (FKM). The project is supported via the German Federation of Industrial Research Associations (AiF) in the program for Industrial Collective Research (IGF) by the Federal Ministry of Economic Affairs and Climate Action (BMWK) on the basis of a decision by the German Bundestag.

## Bibliography

[Ca05] Casas, Roberto; Gracia, Héctor J; Marco, Alvaro; Falco, Jorge L: Synchronization in wireless sensor networks using bluetooth. In: Third International Workshop on Intelligent Solutions in Embedded Systems, 2005. IEEE, pp. 79–88, 2005.

[Ce15] Cena, Gianluca; Scanzio, Stefano; Valenzano, Adriano; Zunino, Claudio: Implementation and evaluation of the reference broadcast infrastructure synchronization protocol. IEEE Transactions on Industrial Informatics, 11(3):801–811, 2015.

[GBM18] Grunau, Sergej; Block, Dimitri; Meier, Uwe: Multi-label wireless interference classification with convolutional neural networks. In: 2018 IEEE 16th International Conference on Industrial Informatics (INDIN). IEEE, pp. 187–192, 2018.

[IE20] IEEE: IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. IEEE Std 1588-2019 (Revision of IEEE Std 1588-2008), pp. 1–499, 2020.

[IE22a] IEC: IEC 62657-2:2022, Industrial networks - Coexistence of wireless systems - Part 2: Coexistence management. IEC Webstore, 2022.

[IE22b] IEC: IEC 62657-4:2022, Industrial networks - Coexistence of wireless systems - Part 4: Coexistence management with central coordination of wireless applications. IEC Webstore, 2022.

[Ku18] Kulin, Merima; Kazaz, Tarik; Moerman, Ingrid; De Poorter, Eli: End-to-end learning from spectrum data: A deep learning approach for wireless signal identification in spectrum monitoring applications. IEEE access, 6:18484–18501, 2018.

[Ma16] Mahmood, Aneeq; Exel, Reinhard; Trsek, Henning; Sauter, Thilo: Clock synchronization over IEEE 802.11—A survey of methodologies and protocols. IEEE Transactions on Industrial Informatics, 13(2):907–922, 2016.

[Pu20] Puttnies, Henning; Danielis, Peter; Sharif, Ali Rehan; Timmermann, Dirk: Estimators for time synchronization—survey, analysis, and outlook. IoT, 1(2):398–435, 2020.

[SBM17] Schmidt, Malte; Block, Dimitri; Meier, Uwe: Wireless interference identification with convolutional neural networks. In: 2017 IEEE 15th International Conference on Industrial Informatics (INDIN). IEEE, pp. 180–185, 2017.

[Sc17] Schulze, Darina; Rauchhaupt, Lutz; Kraetzig, Marko; Jumar, Ulrich: Coexistence plant model for an automated coexistence management. IFAC-PapersOnLine, 50(1):355–362, 2017.

[SJ07] Schriegel, Sebastian; Jasperneite, Jürgen: Investigation of industrial environmental influences on clock sources and their effect on the synchronization accuracy of ieee 1588. In: 2007 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication. IEEE, pp. 50–55, 2007.

[SZ18] Schulze, Darina; Zipper, Holger: A Decentralised Control Algorithm for an Automated Coexistence Management. In: 2018 IEEE Conference on Decision and Control (CDC). IEEE, pp. 4187–4193, 2018.

[VD19] VDI/VDE: VDI/VDE 2185 Part 4, Radio-based communication in industrial automation - Metrological performance rating of wireless solutions for industrial automation applications. VDI/VDE-Richtlinien. Beuth Verlag GmbH, 2019.

[WM18] Wiebusch, Nico; Meier, Uwe: Evolutionary resource allocation optimization for wireless coexistence management. In: 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA). volume 1. IEEE, pp. 1197–1200, 2018.

[Zh19] Zhang, Xiwen; Seyfi, Tolunay; Ju, Shengtai; Ramjee, Sharan; El Gamal, Aly; Eldar, Yonina C: Deep learning for interference identification: Band, training SNR, and sample selection. In: 2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC). IEEE, pp. 1–5, 2019.

# 5G mmWave für industrielle Kommunikation

**RF-Test und Ergebnisse**

Ulrich Sinn[1]

**Abstract:** 5G mmWave-Technologie bietet sich wegen ihrer technischen und physikalischen Eigenschaften für den Einsatz in einer Reihe von industriellen Anwendungen an. Verlässliche praktische Informationen über den Einsatz solcher Funksysteme in industrieller Umgebung liegen jedoch noch nicht vor. Deshalb wurde im Rahmen eines Tests der Signalpegel des 5G-NR-Signals in industriellen Fertigungsumgebungen untersucht. Zum Einsatz kam ein Small Cell-Gerät im Testmodus, von dessen Signal der Wert SS-RSRP als Indikator herangezogen wurde. Bei LOS und leichtem OLOS boten die Signalpegel gute Voraussetzungen für die Kommunikationen eines realen Endgerätes. Bei starkem OLOS und NLOS ist nur eine eingeschränkte Kommunikation bzw. keine Kommunikation zu erwarten. Die Entfernung spielte in den untersuchten Bereichen bis 70 m keine entscheidende Rolle, solange keine Hindernisse die Signalausbreitung behinderten. Reflexionen – zufällig entstanden oder gezielt eingesetzt – können durch Hindernisse entstehende Signaleinbrüche kompensieren.

**Keywords:** 5G, mmWave, Industrie

## 1 Einleitung

Drahtlose Kommunikation, insbesondere auf Basis von Funktechnik, kommt seit vielen Jahren in verschiedensten industriellen Anwendungen zum Einsatz. Die Zahl der eingesetzten Funksysteme nimmt ständig zu, und mit Konzepten wie Industrie 4.0 hat sich der Zuwachs noch beschleunigt. Bestimmte Anforderungen wie Flexibilität und Mobilität lassen sich nur mit Funktechnik praktikabel umsetzen, so dass sowohl der Bedarf an Funksystemen als auch deren Leistungsfähigkeit weiter ansteigen.

Dem zunehmenden Einsatz stehen jedoch verschiedene Probleme entgegen, insbesondere der Mangel an Spektrum, Koexistenzkonflikte durch den Betrieb in lizenzfreien Bändern sowie die limitierte Unterstützung von Echtzeitkommunikation.

Die Mobilfunktechnologie 5G ist u. a. angetreten, diese Probleme zu lösen und die etablierten drahtlosen Technologien zumindest zu ergänzen. 5G zeichnet sich u. a. dadurch aus, dass es in lizensiertem Spektrum und damit ohne Koexistenzprobleme betrieben wird und echtzeitfähige Kommunikation mit Zykluszeiten bis zu 1 ms realisierbar macht. Darüber hinaus kommt mit 5G die Möglichkeit, private Netze für vertikale Märkte und Industrien, wie z. B. die industrielle Fertigung, aufzubauen, was zum einen durch

[1] Siemens AG, DI PA TI CE, Gleiwitzer Straße 555, 90475 Nürnberg, ulrich.sinn@siemens.com

Architekturoptionen und zum anderen durch die zunehmende Verfügbarkeit von Spektrum für ebendiese Anwendungen ermöglicht wird.

Im Vergleich mit der Vorgängertechnologie 4G (LTE), die Bänder unterhalb von 6 GHz adressiert, ist 5G ab Release 17 für Frequenzen von bis zu 71 GHz spezifiziert. Fast alle Netze arbeiten jedoch im Bereich unterhalb von 4 GHz, nur wenige oberhalb von 24 GHz und damit im Bereich der mm-Wellen (mmWave). Dies gilt insbesondere für private Netze. Es ist jedoch zu erwarten, dass der kommerzielle Einsatz von mm-Wellen-Systemen zunehmen wird, und eine Reihe von Gründen sprechen für einen Einsatz auch in der Industrie.

Verlässliche praktische Informationen über den Einsatz von Funksystemen im mm-Wellen-Bereich in industrieller Umgebung liegen jedoch noch nicht vor. Es wurde deshalb im Rahmen der hier dargestellten Arbeiten untersucht, wie sich ein 5G mmWave-System in industriellen Fertigungsumgebungen verhält.

# 2 Einsatz von 5G mmWave-Technologie in der Industrie

## 2.1 5G mmWave

3GPP hat 5G für bis zu 71 GHz spezifiziert (ab Rel. 17, vorher bis zu 52,6 GHz) und dafür mit FR1 und FR2 zwei Frequenzbereiche definiert. Während FR1 mit 410 MHz – 7,125 GHz den sub-7-GHz-Bereich abdeckt, reicht FR2 von 24,25 GHz bis 71 GHz. FR2 wird oft auch als 5G mmWave bezeichnet.

Für FR2 nutzbares Spektrum befindet sich je nach Land in zwei breiten Bereichen zwischen 24 und 29 GHz sowie 37 und 43 GHz. In der EU ist es der Bereich 24,25 – 27,5 GHz, der mit 5G genutzt werden kann – in Deutschland sowohl für öffentliche als auch nichtöffentliche lokale Netze. Identische bzw. ähnliche Frequenzbereiche sind auch in vielen Ländern bereits zugeteilt oder verfügbar.

## 2.2 Vor- und Nachteile

Funksysteme im mm-Wellen-Bereich sind bisher wenig verbreitet, da sie durch eine hohe Dämpfung – sowohl im Freiraum als auch beim Durchdringen von Materie – und infolgedessen geringe Reichweite gekennzeichnet sind.

Für industrielle Anwendungen verspricht 5G mmWave jedoch eine Reihe interessanter Vorteile, wie z. B.:

- Hohe Datenrate (durch breite Kanäle bis 1 GHz)
- Geringe Latenz (sehr kurze Symbole und Pakete durch erweiterte 5G Numerology)

- Reduzierte Interferenz (durch geringe Reichweite und Antennen mit Beamforming)
- Hohe Genauigkeit von Lokalisierung und Positionierung (durch hohe Bandbreite und zeitliche Auflösung)
- Zusätzliches Spektrum

Den Vorteilen stehen jedoch auch mehrere potenzielle Nachteile gegenüber:

- Begrenzte Reichweite (durch hohe Dämpfung und geringe Durchdringung von Materialien)
- Erhöhte Verlustleistung (durch hohe Datenrate und große Zahl an TX/RX-Pfaden)
- Erhöhte Kosten (wegen Komplexität und aufwändigerer Infrastruktur zur Abdeckung größerer Flächen, Technologie noch in der Anlaufphase)

### 2.3 Potenzielle Anwendungen und Einsatzbereiche

5G mmWave-Technologie bietet sich wegen ihrer technischen und physikalischen Eigenschaften für den Einsatz in einer Reihe von industriellen Anwendungen an. Beispiele zeigt die folgende Liste:

- Abdeckung von Flächen oder Punkten mit sehr hohen Anforderungen an die Datenrate (z. B. AR/VR, Video-Anwendungen, Software-Downloads)
- Flächen mit einer großen Zahl an Roboterzellen oder Maschinen (die begrenzte Reichweite erlaubt eine hohe Dichte an Verbindungen pro Flächeneinheit)
- Anwendungen mit niedriger Latenz (z. B. Roboter, Produktionsmaschinen, Antriebe)
- Bei hohen Anforderungen an die Genauigkeit von Lokalisierung und Positionierung (z. B. AGVs, kooperative Roboter)

Darüber hinaus existieren weitere mögliche Vorteile. So lassen sich Verbindungen über sub-6 GHz und mmWave komplementär zueinander konzipieren, z. B. durch unterschiedliche UL/DL-Schemata, um den heterogenen Anforderungen in einer industriellen Produktion gerecht zu werden. Einzelne, besonders Ressourcen-intensive, Anwendungen werden vorzugsweise über den mmWave-Bereich abgewickelt. Auch der Zugang zu Spektrum kann sich für den mm-Wave-Bereich einfacher als für den sub-6-GHz-Bereich gestalten. Wegen der begrenzten Reichweite werden damit keine Flächennetzwerke aufgebaut, und Industrieanlagen sind für Netzbetreiber üblicherweise keine relevanten Hotspots.

# 3 Test

## 3.1 Testsystem

Für den Test war ursprünglich der Aufbau eines kleinen 5G SA DC-Netzes (Stand-Alone Dual Connectivity) angedacht. Dieses ließ sich für den Testzeitraum jedoch nicht realisieren, da weder die erforderliche Infrastruktur noch geeignete Endgeräte kommerziell zur Verfügung standen.

Deshalb wurde der Test mit Hilfe eines 5G Small Cell-Gerätes durchgeführt, das in einem Testmodus betrieben wurde. Als Empfangsgerät kam ein Messgerät für 5G-Signale zum Einsatz (Viavi CellAdvisor 5G).

Die Small Cell arbeitete dabei wie folgt:

- Zyklische Übertragung von 5G NR-Signalen (PSS (Primary Synchronization Signal), SSS (Secondary Synchronization Signal), PBCH (Physical Broadcast Channel))
- Zwei Component Carrier mit jeweils 100 MHz (26,9 GHz und 27,0 GHz)
- Nur Downlink, kein Uplink
- Sendeleistung max. 40 dBm EIRP
- Kein Beamforming und Beam Steering

Die Konfiguration wurde über einen angeschlossenen PC vorgenommen.

Für die Tests wurde die Small Cell auf einem höhenverstellbaren Stativ montiert, mit dem sich eine Höhe von ca. 3,6 m sowie die Möglichkeit zur Ortsveränderung realisieren ließ. Eine feste Montage unterhalb eines Hallendachs wäre zwar praxisnäher und für die Ausleuchtung von Bereichen einer Industriehalle geeigneter gewesen, doch hätte dies einen deutlicher größeren Aufwand bedeutet und Tests in verschiedenen Hallen so gut wie ausgeschlossen.

Das Messgerät war mit einer Hornantenne ausgestattet (26.5 – 40 GHz, Gain 20 dBi, 3 dB-Bandbreite: 16.7° (E-Plane) / 18.3° (H-Plane)). Während der Tests wurden Messgerät und Antenne auf einem Laborwagen platziert, so dass sich Messungen sowohl stationär als auch mobil bequem durchführen ließen.

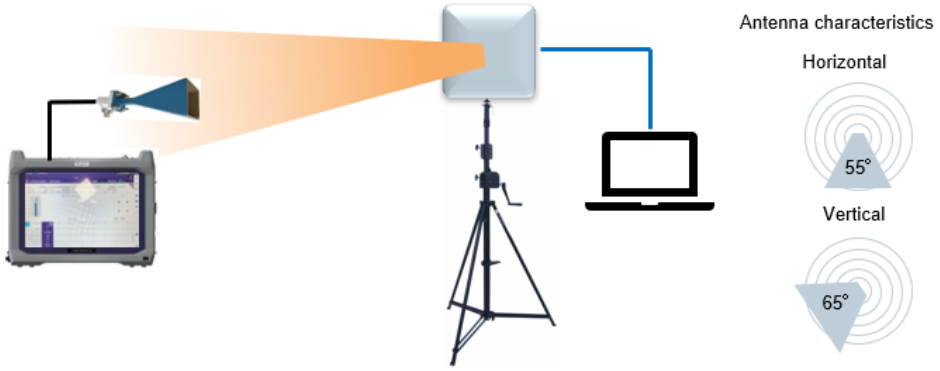



Abb. 1: Schematischer Aufbau des Testsystems

### 3.2 Durchführung der Tests und Bewertung der Messergebnisse

Während der meisten Tests wurde das Messgerät über eine längere Strecke bewegt und die Signale der Small Cell permanent aufgezeichnet. Die Antenne war dabei in Richtung der Small Cell ausgerichtet. Zusätzlich wurden an einigen Stellen Snapshots der Parameter aufgezeichnet. Dabei wurde die Antenne manuell in Richtung des stärksten Signals geschwenkt.

Als Indikator für die Qualität des empfangenen Signals wurde der Wert SS-RSRP (Synchronization Signal - Reference Signal Received Power) herangezogen. Er ist definiert als mittlere lineare Leistung der Ressourcenelemente, die sekundäre Synchronisationssignale übertragen. Dieser Leistungswert ist sehr klein und liegt auch immer unterhalb des RSSI-Wertes. Als Maximalwert wurde ein Wert von ca. -70 dBm gemessen. Werte für SS-RSRP zwischen -70 dBm und -100 dBm wurden als sehr gut bis gut gewertet; sie würden eine stabile und gute Kommunikation mit einem realen Endgerät erlauben. Werte zwischen -100 dBm und -120 dBm wurden als kritisch betrachtet, da sie eine Kommunikation nur noch mit mehr oder minder starken Einschränkungen zulassen. Werte unterhalb von -120 dBm wurden als nicht nutzbar eingestuft. Für diese Einschätzung wurden unter anderem Messungen der Firma Rohde & Schwarz herangezogen, die den SS-RSRP-Wert für Abdeckungstests von 5G-Netzen bei 28 GHz verwendeten [Si19]. Ergänzend wurden der EVM-Wert (Error Vector Magnitude) des PBCH (in Prozent) bzw. das zugehörige Konstellationsdiagramm herangezogen.

### 3.3 Durchgeführte Tests und Ergebnisse

Die Tests wurden an einem Nürnberger Standort der Siemens AG bzw. der Siemens Mobility GmbH durchgeführt, der sowohl über ein großes Spektrum an Anlagen für die mechanische Fertigung als auch verschiedene Stufen der Elektronikfertigung verfügt.

Fotos der Testorte finden sich in Abb. 2.

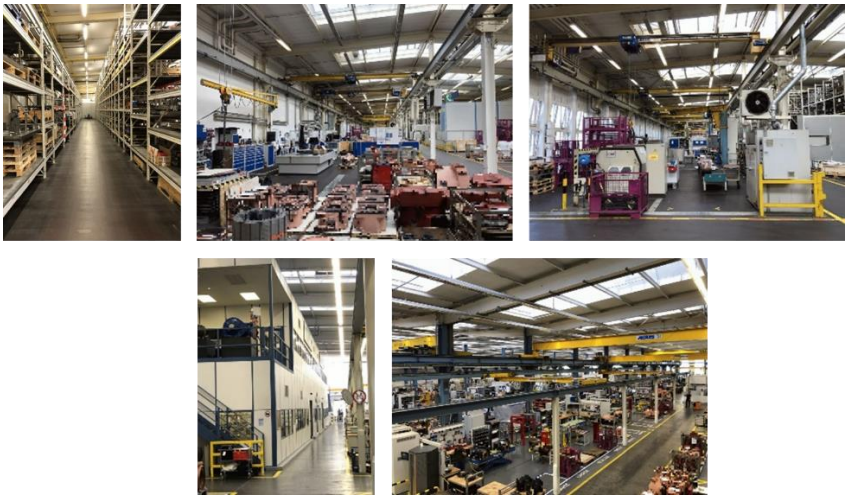


Abb. 2: Fotos der Testorte (Hochregallager, Werkhallengang (2x), Werkhalle (2x)

**Hochregallager**

Der folgende Test wurde in einem Hochregallager durchgeführt. Das Messgerät wurde dabei kontinuierlich von der Small Cell bis zur maximalen Entfernung von ca. 40 m bewegt. Im ersten Durchgang befanden sich Small Cell und Messgerät im selben Gang in Sichtverbindung (LOS, line-of-sight), während im zweiten und dritten Durchgang Small Cell und Messgerät durch eine Regal-Doppelreihe bzw. eine dritte Regalreihe getrennt waren (gestörte Sichtverbindung, OLOS (obstructed line-of-sight)). Die Small Cell war in einer Höhe von 3,5 m befestigt. Die Testanordnung zeigt Abb. 3, die Ergebnisse Abb. 4a, 4b und 4c.

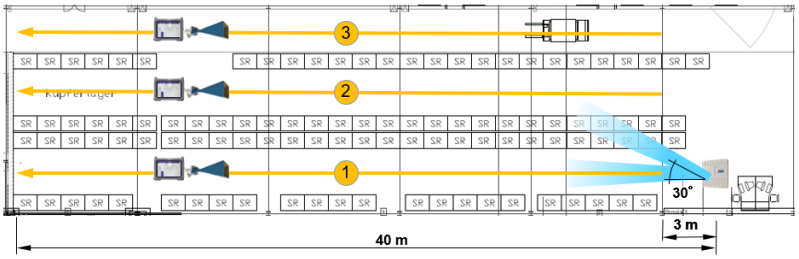



Abb. 3: Testanordnung Hochregallager

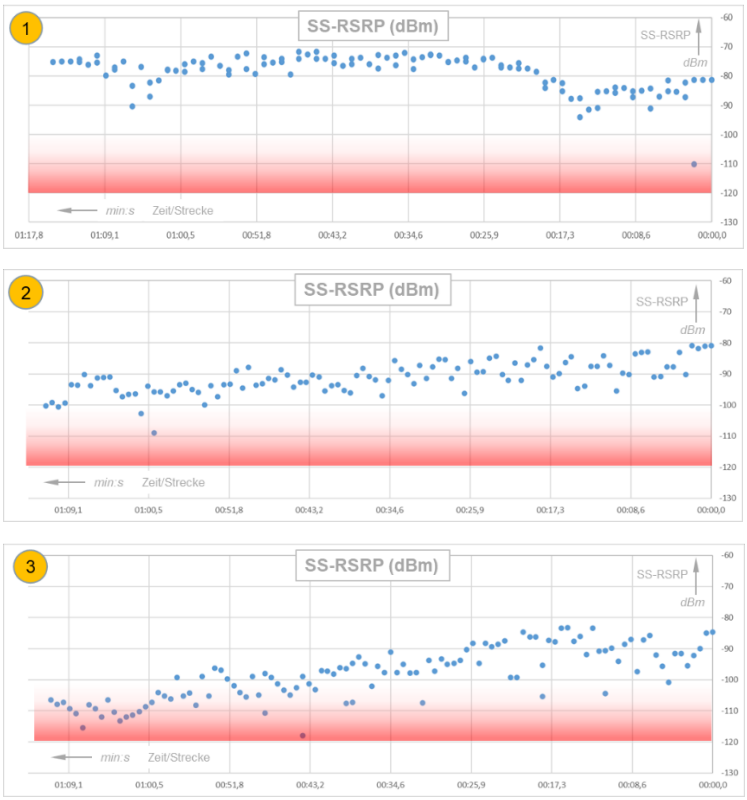



Abb. 4a, 4b, 4c: Gemessene Werte für SS-RSRP im Hochregallager

Die gemessenen Signalpegel sind korrespondierend zur Richtung des Tests dargestellt. In Test 1 (Abb. 4a) mit Sichtverbindung bleibt das Signal ab einer Entfernung von ca. 10 m über die gesamte restliche Distanz zwischen -70 und -80 dBm. Der niedrigere Signalpegel auf den ersten 10 m entsteht, da sich die Empfangsantenne dort unterhalb der Hauptkeule (Beam) der Antenne befindet. Der Einbruch zwischen 30 und 35 m ließ sich nicht erklären. Die Tests in den Nachbargängen mit gestörter Sichtverbindung führten zu deutlich niedrigeren und mit zunehmender Entfernung abnehmenden Signalpegeln. Bei Test 2 (Abb. 4b) liegen die Werte in der ersten Hälfte zwischen -80 und -100 dBm, in der zweiten Hälfte zwischen -90 und -100 dBm. Bei Test 3 (Abb. 4c) liegen die Werte in der ersten Hälfte zwischen -80 und -110 dBm, in der zweiten Hälfte zwischen -95 und -120 dBm; die Pegel sind also nochmals niedriger und die Schwankungsbreite ist höher.

Die Signalpegel lassen den Schluss zu, dass auch im benachbarten Gang die Kommunikation möglich ist. Dies gilt – mit Einschränkungen – auch für den übernächsten Gang. Hier ist wegen einiger Dropouts mit -110 dB und dem zum Ende hin sehr niedrigen Pegel eine zum Teil gestörte und am Ende abbrechende Kommunikation zu erwarten.

### Werkhallengang

Ein weiterer Test wurde entlang eines Gangs einer langgestreckten Werkhalle, die mit verschiedenen Fertigungseinrichtungen bestückt war, durchgeführt. Gemessen wurde, mit zunehmender Entfernung, sowohl auf dem Gang als auch an vier Stellen zwischen den Fertigungseinrichtungen. Die Small Cell war in einer Höhe von 3,5 m befestigt – ausreichend, um einen Teil der Hindernisse zu überstrahlen. Die ersten 20 m bestand LOS, anschließend – bedingt durch die Höhe der Fertigungseinrichtungen von ca. 2 m – OLOS bzw. NLOS (non-line-of-sight). Die Testanordnung zeigt Abb. 5, die Ergebnisse Abb. 6.

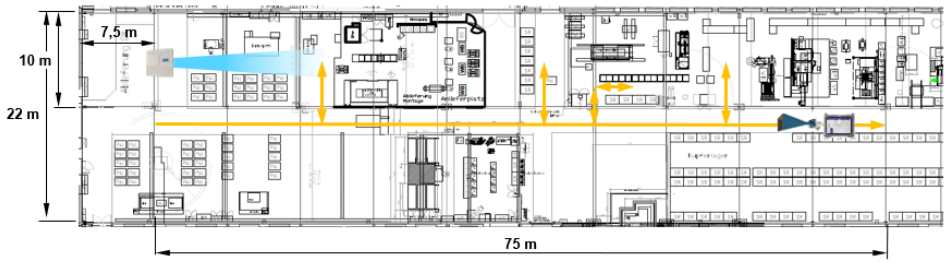



Abb. 5: Testanordnung Werkhallengang

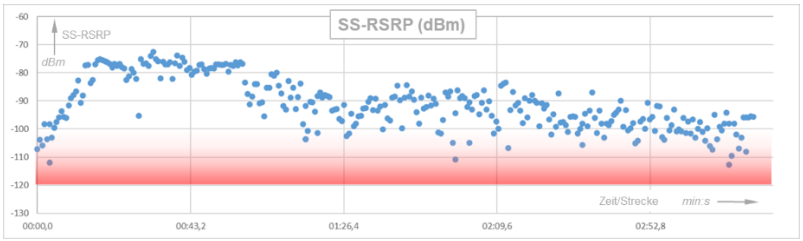



Abb. 6: Gemessene Werte für SS-RSRP im Werkhallengang

Der Signalpegel liegt die ersten 20 m zwischen -70 und -80 dBm (die ersten Meter wieder niedriger, weil außerhalb der Hauptkeule der Antenne) und nimmt mit zunehmender Entfernung kontinuierlich ab. Ab Beginn der Hindernisse bei ca. 20 m nimmt die Varianz des Signalpegels zu: Von 20 m – 50 m liegen die Werte zwischen -80 und -110 dBm, von 50 m – 75 m zwischen -90 und -110 dBm, wobei ab ca. 60 m ein weiterer Abfall zu beobachten ist. Messungen direkt im Fertigungsbereich führten zu keinen abweichenden Ergebnissen und sind in Abb. 6 nicht zu identifizieren.

Die Messergebnisse lassen erwarten, dass deutlich über den LOS-Bereich hinaus bis zur Grenze des Tests bei 75 m eine Kommunikation möglich ist. Mit wachsender Entfernung nimmt jedoch die Zahl der Stellen zu, an denen das Signal unten -100 dBm fällt und damit Probleme oder Abbrüche zu erwarten sind.

**Werkhalle**

In diesem Test wurde untersucht, welche Teile einer kompletten Werkhalle abgedeckt werden. Dazu wurde die Small Cell in der Mitte einer Längsseite auf einer Empore in einer Höhe von 5 m platziert. Die Halle (Höhe: 9,3 m) war gefüllt mit kleineren Maschinen und manuellen Arbeitsplätzen. Unterhalb der Decke befanden sich Metallschienen für Portalkrane sowie Träger für Medienleitungen.

Für den Test wurde der Signalpegel in mehreren Gängen aufgezeichnet, die sowohl innerhalb als auch außerhalb des Abdeckungsbereichs der Small Cell lagen. Die Ausbreitungsbedingungen lassen sich wie folgt charakterisieren: Gang 1 LOS/NLOS, Gang 2 OLOS, Gang 4 NLOS, Gang 5 LOS/OLOS. Die ungehinderte Ausbreitung in die Halle war durch die unterhalb der Decke montierten Hindernisse stark eingeschränkt. Die Testanordnung zeigt Abb. 7, die Ergebnisse Abb. 8a – 8d.

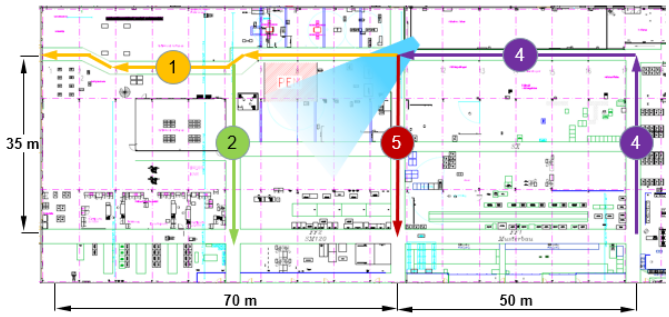
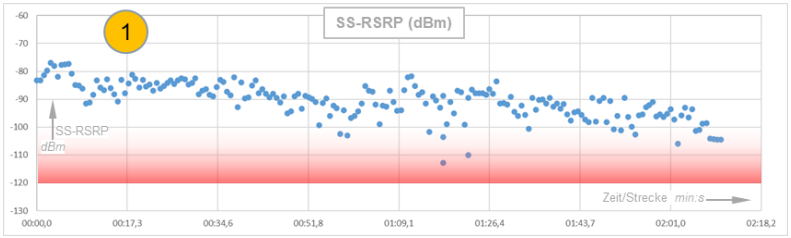
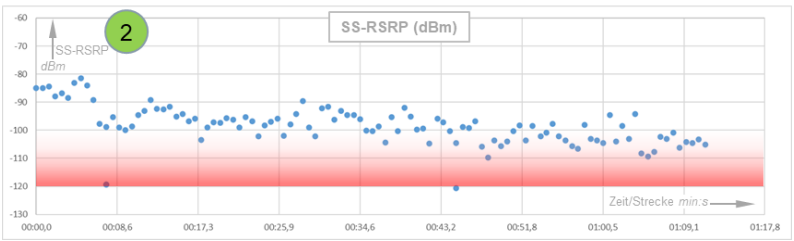



Abb. 7: Testanordnung Werkhalle



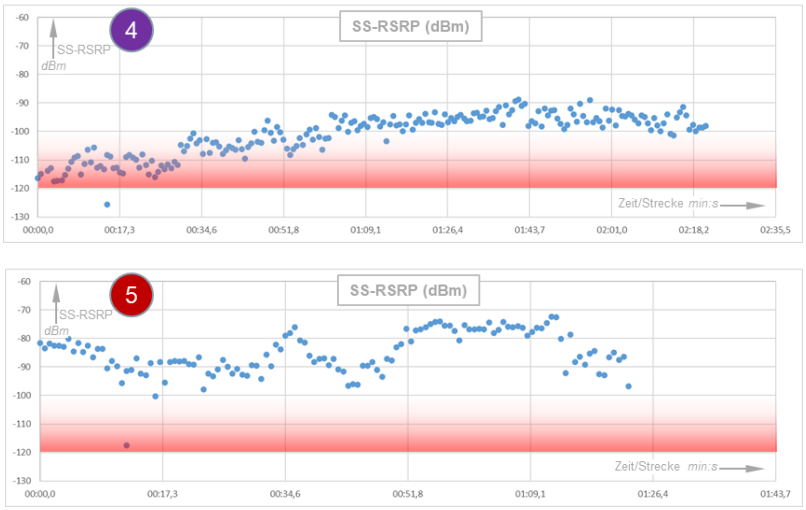



Abb. 8a, 8b, 8c, 8d: Gemessene Werte SS-RSRP in der Werkhalle

Bei Gang 1 ist ein hoher Signalpegel (-80 – -90 dBm) für die ersten 30 m (LOS) zu erkennen, der anschließend (NLOS) abfällt und eine deutlich höhere Varianz aufweist (-80 und -112 dBm). In Gang 2 (OLOS) ist der Signalpegel insgesamt niedriger, weist aber keine große Varianz auf; Schwankungen entstanden durch die wechselnden Abschattungen. Gang 4, nicht im Abdeckungsbereich, hat wenig überraschend einen anfangs sehr niedrigen Pegel, der mit abnehmender Entfernung kontinuierlich ansteigt und eine geringe Varianz zeigt. In Gang 5 (LOS/OLOS) ist der Signalpegel wechselnd, aber konstant über -100 dBm und weist lokal eine geringe Varianz auf.

**Reflexionen**

Auch der Einfluss und die Nutzung von Reflexionen wurde untersucht.

Im Flur eines Bürogebäudes wurden Small Cell und Empfänger unter NLOS-Bedingungen platziert. Als Reflektor kam eine gebogene Stahlwand (Sichtschutz) zum Einsatz. Mit seiner Hilfe konnte auch ohne Sichtverbindung ein Signal mit einem hohen Pegel von -76 dBm und sauberem Konstellationsdiagramm empfangen werden (siehe Abb. 9).

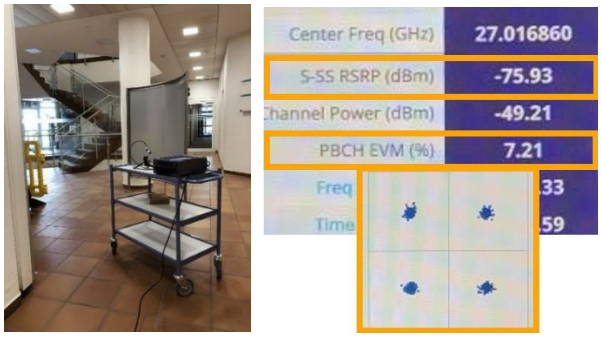



Abb. 9: Stahlwand als Reflektor, Messwerte

In einer Werkhalle wurde die Small Cell in einer Höhe von ca. 3,5 m platziert und der Signalpegel in einem Gang gemessen, der durch ca. 2,5 m hohe Bearbeitungszentren gebildet wurde (Abb. 10). In diesem Gang bestand keine direkte Sichtverbindung zur Small Cell. Während der ersten Messung war die Abstrahlrichtung der Small Cell nach vorn gerichtet. Die gemessenen Signalpegel sind in Abb. 11 als rote Punkte dargestellt. Für die zweite Messung war die Abstrahlrichtung der Small Cell in einem Winkel von 45° zur Decke gerichtet. Die Decke befand sich in einer Höhe von 11 m und bestand aus profiliertem Metallblech. Die gemessenen Signalpegel sind in Abb. 11 als blaue Punkte dargestellt und liegen (bis auf eine Messung) zwischen 0 dBm und 20 dBm höher als die der ersten Messung.

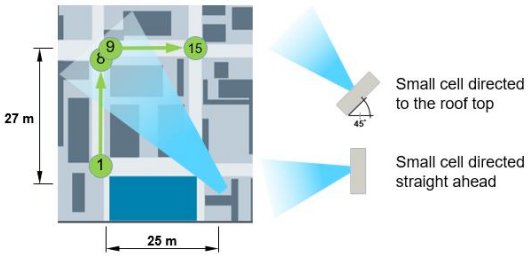



Abb. 10: Testanordnung Maschinengang

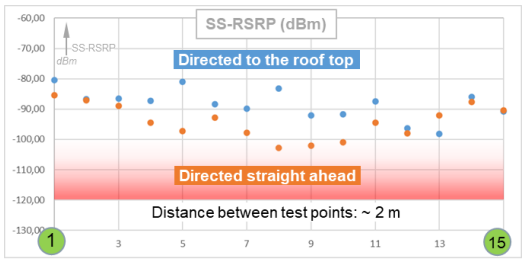



Abb. 11: Gemessene Werte SS-RSRP Maschinengang

Die Ergebnisse der beiden Tests zeigen, dass durch Reflexionen, die sich auch gezielt einsetzen lassen, deutliche Verbesserungen der Signalpegel erreichbar sind.

## 4 Bewertung und Ausblick

Im Ergebnis lag der Pegel des SS-RSRP-Signals bei LOS konstant zwischen -70 und -80 dBm und leichtem OLOS zwischen -80 und -100 dBm und bot damit gute Voraussetzungen für die Kommunikationen eines realen Endgerätes. Bei starkem OLOS und NLOS lagen die Pegel gewöhnlich zwischen -90 und -110 dBm, bei schwierigen NLOS – besonders bei größerer Entfernung – zwischen -110 und -120 dBm und darunter. Mit diesen Pegeln ist nur eine eingeschränkte Kommunikation bzw. ein Abbrechen zu erwarten.

Hohe Pegel wurden erwartungsgemäß in der Hauptabstrahlrichtung der Antenne gemessen, gute bis ausreichende aber auch außerhalb. Die Entfernung spielte in den untersuchten Bereichen bis 70 m keine entscheidende Rolle, solange keine Hindernisse die Signalausbreitung behinderten. Reflexionen – zufällig entstanden oder gezielt eingesetzt – können durch Hindernisse entstehende Signaleinbrüche kompensieren.

In einem nächsten Schritt sollen Tests mit einem realen 5G-mmWave-System folgen. Damit lassen sich z. B. Einflüsse auf die Ende-zu-Ende-Kommunikation (z. B. Datendurchsatz, Latenz) untersuchen und eine Korrelation mit den Ergebnissen dieser Studie überprüfen. Auch Funktionen wie Beamforming und Beam Steering könnten dann zum Einsatz kommen.

Literaturverzeichnis

[Si19] Rohde & Schwarz, www.mobilewirelesstesting.com/wp-content/uploads/2019/12/5G_real_world_use_cases.pdf, Stand: 18.07.2022

# Latenzoptimierte, kontaktlose Datenübertragung für Industrial Ethernet

Jan Letmade[1], Roland Hess[2], Stefan Witte[3]

**Abstract:** Kontaktlose Daten- und Energieübertragung kann in der Industrie viele Vorteile gegenüber herkömmlichen Steckverbindern bieten. Dies gilt insbesondere dann, wenn die Steckverbindung trenn- oder verdrehbar sein soll, da Steck- oder Schleifkontakte konstruktionsbedingt eine begrenzte Lebensdauer und Resistenz gegen Umwelteinflüsse aufweisen. In dieser Arbeit wird ein System zur kontaktlosen Datenübertragung für Industrial Ethernet im 60,5 GHz Band vorgestellt. Das System nutzt *Commercial off-the-shelf* (COTS) Komponenten und erreicht damit eine völlig transparente Übertragung von 100BASE-TX Ethernet mit einer festen Latenz von lediglich 5,27 µs, womit es sich für Echtzeit- und Motion-Control-Anwendungen eignet.

**Keywords:** Kontaktlose Datenübertragung; Industrial Ethernet

## 1 Einleitung

Kontaktlose Datenübertragung kann im industriellen Umfeld einige Vorteile gegenüber konventionellen Steckverbindern bieten. Da kein physischer Kontakt hergestellt wird, kann die Verbindung besser gegen Umwelteinflüsse geschützt werden und es tritt kein Verschleiß auf, wie dies sonst an Kontaktflächen der Fall wäre. Kontaktlose Datenübertragung erlaubt zudem Rotationsfreiheit und geringfügige Verschiebung, was die Verbindung auch resistent gegenüber Vibrationen macht. Dies ermöglicht Anwendungen wie Beispielsweise virtuelle Kupplungen zwischen autonomen Fahrzeugen oder die Kommunikation mit wechselbaren Werkzeugen eines Industrieroboters.

Das in dieser Arbeit vorgestellte Funksystem nutzt den ST60A2 von *STMicroelectronics*. Dies ist ein 60,5 GHz V-Band Transceiver welcher mit ASK-Modulation eine Datenrate von bis zu 6,25 Gbit/s erreicht. Mit diesem Chip existieren bereits Produkte, welche eine Vollduplex-Übertragung über verschieden polarisierte Antennen erreichen. Damit sind jedoch zwei Hardwaretypen nötig, damit Beispielsweise die horizontal polarisierte Antenne bei einem Gerät an TX und bei dem gegenüberliegenden Gerät an RX angeschlossen ist. Dies macht das System weniger flexibel einsetzbar, weshalb in dieser Arbeit das Zeitduplexverfahren (TDD) genutzt wurde.

[1] OWITA GmbH, Campusallee 6, 32657 Lemgo, jan.letmade@owita.com
[2] OWITA GmbH, Campusallee 6, 32657 Lemgo, roland.hess@owita.com
[3] Technische Hochschule OWL, Campusallee 12, 32657 Lemgo, stefan.witte@th-owl.de

## 2 Anforderungen

Da Industrial Ethernet momentan die am häufigsten verwendete Netzwerktechnologie ist [Ca], wurde 100BASE-TX Ethernet als Schnittstelle für die kontaktlose Datenübertragung gewählt. Die Daten der Ethernet-Schnittstelle sollen dabei vollkommen transparent übertragen werden, um maximale Protokollkompatibilität zu gewährleisten. Da Protokolle wie Profinet Zykluszeiten von 250 µs oder niedriger erlauben [In], sollte die Latenz nur einige Mikrosekunden betragen. Die Latenz sollte zudem konstant sein und nur wenig Jitter aufweisen, um die Genauigkeit von Protokollen wie *Precision Time Protocol* (PTP) nicht zu sehr zu beeinträchtigen [Sc09].

## 3 Berechnung von Latenz und Datenrate

Um eine konstante Latenz zu erreichen, sollen beide Kommunikationsteilnehmer die Daten abwechselnd in gleichlangen Zyklen austauschen. Die Zykluszeit $t_S$ und damit die minimal erreichbare Latenz kann daher mit folgender Formel berechnet werden

$$t_S = 2 \cdot (t_{Switch} + N \cdot t_{Bit}) \quad , \tag{1}$$

wobei $t_{Switch}$ die Umschaltzeit des Transceivers von RX auf TX bzw. TX auf RX ist. Die effektive Bitrate $R_{eff}$ der Übertragung ist aufgrund der Umschaltzeit niedriger als die Bitrate des Systems ($R = {}^1/_{t_{bit}}$) und kann mit folgender Formel berechnet werden.

$$N = \frac{R_{eff} \cdot t_{Switch}}{1 - \frac{R_{eff}}{R}} \tag{2}$$

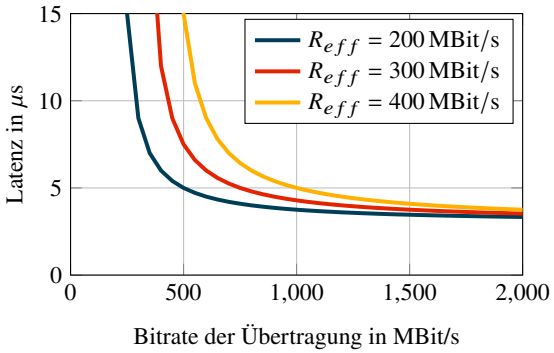



Abb. 1: Latenz der Datenübertragung in Abhängigkeit zur Bitrate für verschiedene effektive Bitraten

In Abbildung 1 ist dies für verschiedene effektive Datenrate dargestellt. Es ist ersichtlich, dass bei $R_{eff} = 200\,\text{MBit/s}$ für eine Latenz von 5 µs bereits eine Bitrate von 500 MBit/s ausreicht. In diesem Fall ist allerdings keine zusätzliche Kapazität für Kodierung oder

Protokoll-Overhead vorhanden. Bei $R_{eff}$ = 400 MBit/s wird für dieselbe Latenz bereits eine Bitrate von 1 GBit/s benötigt. Da sich die Latenz in einer realen Implementierung durch Encoder, Decoder und die *Ethernet Physical Layer*s (PHYs) noch weiter erhöht, wurde eine Bitrate >1 GBit/s gewählt.

## 4 Hardware

In Abbildung 2 ist die dafür entworfene Hardwarearchitektur schematisch dargestellt. Der PHY ist über das *Reduced Media Independent Interface* (RMII) an das *Field Programmable Gate Array* (FPGA) angeschlossen. Die Daten werden im FPGA kodiert und über die MIPI D-PHY Schnittstelle mit einer Datenrate von 1,2 GBit/s an den Transceiver übertragen. MIPI D-PHY und *Scalable Low Voltage Signaling* (SLVS) weisen ähnliche Spannungslevel auf und können daher direkt verbunden werden. Die vom Transceiver empfangenen Daten werden vom FPGA über zwei *Low Voltage Differential Signaling* (LVDS) Schnittstellen mit einer Bitrate von jeweils 1,2 GBit/s eingelesen. Die Abtastung der beiden Schnittstellen ist um 180° Phasenverschoben, womit die Abtastung effektiv mit 2,4 GBit/s erfolgt. Aufgrund der unterschiedlichen Spannungslevel der Schnittstellenstandards wurde ein Meticom MC20001 SLVS auf LVDS Umsetzer verwendet.

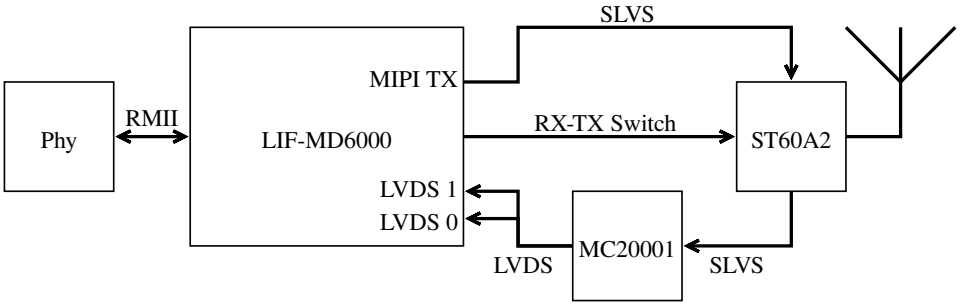



Abb. 2: Schematische Darstellung der verwendeten Hardware

## 5 Verbindungsaufbau

Bevor beide Komkunikationsteilnehmer Daten austauschen können, müssen diese sich synchronisieren und aushandeln, wer mit der Datenübertragung beginnt. Dies wird durch ein einfaches Protokoll erreicht. Beide Kommunikationsteilnehmer senden immer wieder Hello-Nachrichten. Empfängt der andere Kommunikationsteilnehmer diese Nachricht, so antwortet dieser mit einer Reply-Nachricht. Der erste Kommunikationsteilnehmer sendet dann das erste Datenpaket, womit die zyklische Datenübertragung beginnt. Alle Nachrichten (Hello, Reply, Daten) haben dieselbe Länge, da dies die Implementierung vereinfacht und den Verbindungsaufbau nur geringfügig verlängert.

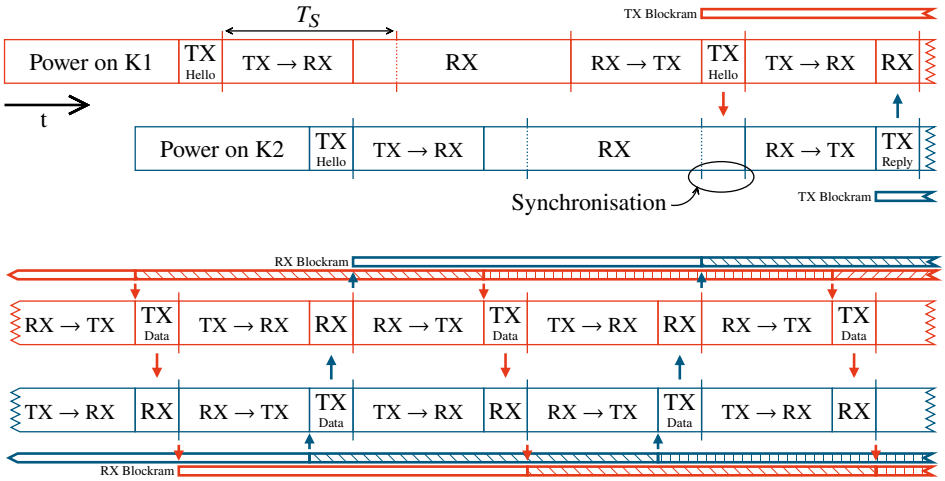



Abb. 3: Darstellung des zeitlichen Ablaufs des Verbindungsaufbaus und der anschließenden Datenübertragung zwischen beiden Kommunikationsteilnehmern (rot und blau).

In Abbildung 3 ist dies genauer dargestellt. Beide Kommunikationsteilnehmer (K1 und K2) senden nach dem Einschalten Hello-Nachrichten, welche zunächst nicht empfangen werden. Die zweite von K1 gesendete Hello-Nachricht wird jedoch von K2 empfangen, worauf dieser mit einer Reply-Nachricht antwortet. Das Empfangen der Nachricht führt zudem zur Synchronisation der Zykluszeiten von K1 und K2. Mit dem Senden der Hello- und Reply-Nachrichten beginnen K1 und K2 die Daten der Ethernet-Schnittstelle im Blockram zu puffern. So kann K1 nach dem Empfang der Reply-Nachricht mit dem Senden von Daten beginnen. Beide Kommunikationsteilnehmer tauschen danach abwechselnd Daten aus.

Während des Verbindungsaufbaus muss entschieden werden, ob in einem Zeitabschnitt eine Hello-Nachricht gesendet wird, oder empfangen wird. Hierfür wurden zwei Methoden entwickelt und optimiert. Bei der ersten Methode wird zufällig entschieden, ob eine Hello-Nachricht gesendet wird. Dies hat den Vorteil, dass keine individuelle Konfiguration der Geräte nötig ist. Die Methode kann optimiert werden, sodass die durchschnittliche Verbindungsdauer minimiert wird, allerdings kann nicht garantiert werden, dass eine Verbindung in einem definierten Zeitraum zustande kommt. Daher wurde ebenfalls ein deterministischer Verbindungsaufbau untersucht. Hierfür sind die Zeitabschnitte in jedem Gerät explizit einprogrammiert, womit eine maximale Verbindungsdauer garantiert werden kann.

### 5.1 Zufällig

Vor dem Verbindungsaufbau können sich die Geräte in einer von vier verschieden Ausgangssituationen befinden. So können die Geräte entweder zum gleichen oder zu unterschiedlichen

Zeitpunkten eingeschaltet werden und beim Einschalten können sich diese entweder bereits in Reichweite oder außerhalb der Reichweite befinden. Werden beide Geräte gleichzeitig eingeschaltet, sind diese (annähernd) synchronisiert, womit die RX und TX Zeitabschnitte übereinander liegen. Die geringste Verbindungsdauer kann in diesem Fall erreicht werden, wenn beide Kommunikationsteilnehmer in jedem Zeitabschnitt mit einer Wahrscheinlichkeit von ½ eine Hello-Nachricht senden. Damit ergibt sich eine durchschnittliche Verbindungsdauer von $1T_S$. In den drei verbleibenden Situationen sind die Kommunikationsteilnehmer in den meisten Fällen jedoch nicht synchronisiert. Damit die Verbindung dennoch zustande kommt, sollten die Empfangszeiten zwischen den Hello-Nachrichten länger und unterschiedlich lang sein, entsprechend $X : \Omega \rightarrow \mathbb{R}$ mit $\mathrm{Im}(X) = \{n \cdot T_S | n \in \mathbb{N}, n_{min} \leq n \leq n_{max}\}$. Um die optimale Strategie für alle Situationen zu finden, wurden die durchschnittliche Verbindungsdauer von unterschiedlichen Strategien und $n_{max}$ in verschiedenen Situationen durch eine Monte-Carlo Simulation ermittelt. Dabei wurde auch untersucht, ob nach dem Einschalten immer mit dem Senden einer Hello-Nachricht begonnen werden sollte oder nur in 50% der Fälle. Da das zufällige Senden nach $\mathrm{Im}(X)$ und das Einschalten der Geräte außerhalb der Funkreichweite äquivalent sind, ist nur eine Kurve eingezeichnet. Die Verbindungsdauer wird ab dem Zeitpunkt wo beide Geräte eingeschaltet und in Reichweite sind und bis zu dem Zeitpunkt wo eine Hello-Nachricht empfangenen wurde gemessen.

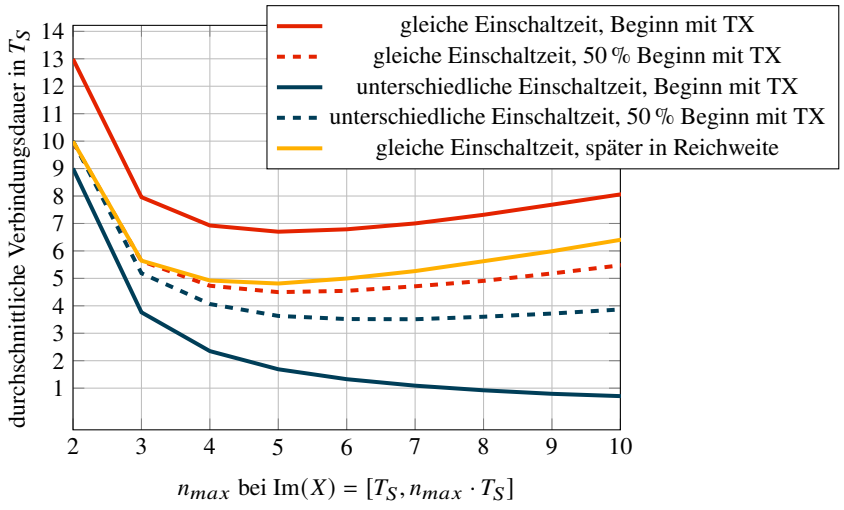



Abb. 4: Vergleich der Verbindungsdauer unter Verwendung verschiedener Strategien in unterschiedlichen Situationen

In Abbildung 4 sind die Ergebnisse zu sehen. Es zeigt sich, dass das direkte Senden einer Hello-Nachricht nach dem Einschalten vorteilhaft ist, wenn die Geräte zu unterschiedlichen Zeitpunkten eingeschaltet werden, jedoch nachteilig ist, wenn die Geräte gleichzeitig eingeschaltet werden. Wird nur in 50% der Fälle mit dem Senden einer Hello-Nachricht begonnen, liegen die Ergebnisse jedoch wesentlich näher beieinander. Wird direkt mit dem zufälligen Senden begonnen, ergibt sich eine geringfügig höhere Verbindungsdauer. Der

beste Kompromiss für alle Situationen ist sicherlich, in 50% der Fälle mit dem Senden einer Hello-Nachricht zu beginnen und $n_{max} = 5$ zu verwenden. Damit erfolgt die Verbindung in 90% der Fälle in weniger als $13T_S$, in 99% der Fälle in weniger als $27T_S$ und in 99,99% der Fälle in weniger als $54T_S$.

### 5.2 Deterministisch

Da für den zufälligen Verbindungsaufbau keine maximale Verbindungsdauer garantiert werden kann, wurde zudem ein deterministischer Verbindungsaufbau entwickelt. Dafür soll in jedem der Geräte eine feste Sequenz eingespeichert sein die definiert, wann eine Hello-Nachricht gesendet wird. Diese Sequenz muss bei beiden Geräten verschieden sein, da sich diese ansonsten nicht in allen Situationen verbinden können.

Tab. 1: Angewendete Regeln zur Generierung einer Sequenz der Länge 4.

| Zahl | Binärzahl | Äquivalent bei Verschiebung | aufeinanderfolgende 1-Bits |
|---|---|---|---|
| 1 | `0001` | - | n |
| 2 | `0010` | 1 | n |
| 3 | `0011` | - | y |
| 4 | `0100` | 1 | n |
| 5 | `0101` | - | n |
| 6 | `0110` | 3 | y |
| 7 | `0111` | - | y |
| 8 | `1000` | 1 | n |
| 9 | `1001` | 3 | y |
| 10 | `1010` | 5 | n |
| 11 | `1011` | - | y |
| 12 | `1100` | 3 | y |
| 13 | `1101` | 11 | y |
| 14 | `1110` | 7 | y |
| 15 | `1111` | - | y |

Die Sequenz kann als Binärzahl angesehen werden, wobei eine „1“ das Senden einer Hello-Nachricht und eine „0“ das Empfangen darstellt. Für eine Sequenzlänge $k$ gibt es daher $2^k - 1$ verschiedene Sequenzen, bei denen mindestens eine Hello-Nachricht gesendet werden würde. Damit ein Verbindungsaufbau möglich ist, muss nach jedem Senden einer Hello-Nachricht in den Empfangsmodus umgeschaltet werden, damit die Antwort nicht verpasst wird. Daher eigenen sich alle Sequenzen mit aufeinanderfolgenden 1-Bits nicht. Als aufeinanderfolgend zählen auch das erste und letzte Bit, da die Sequenzen wiederholt werden. Zuletzt müssen die Folgen invariant gegenüber Verschiebungen sein. Ansonsten könnten sich K1 mit Sequenz `0001` und K2 mit Sequenz `0100` nicht verbinden, wenn K1 $2T_S$ später als K2 eingeschaltet werden würde. Dies ist in Tabelle 1 beispielhaft für $l = 4$ dargestellt. Lediglich die Sequenzen `0001` und `0101` erfüllen beide Regeln und können

verwendet werden. Bei $l = 16$ gibt es aber bereits 35 verschiedene Sequenzen, bei $l = 24$ sind 351 Sequenzen vorhanden und bei $l = 32$ ergeben sich 4115 verschiedene Sequenzen. Die maximale Verbindungsdauer beträgt $l \cdot T_S$.

## 6 Datenübertragung

Für die zyklische Datenübertragung müssen die Daten en- und dekodiert werden. Die gesamte Signalverarbeitungskette dafür ist in Abbildung 5 dargestellt. Im Folgenden werden die Funktionen der einzelnen Blöcke genauer erläutert.

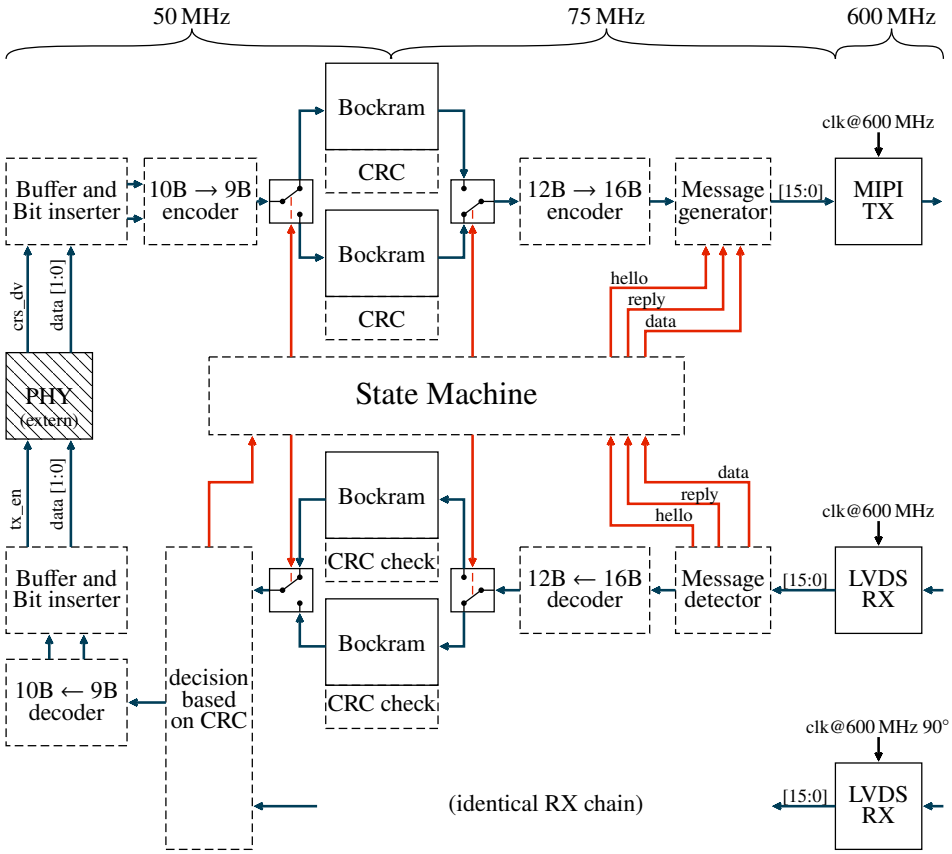



Abb. 5: Blockschaltbild der Signalverarbeitungskette vom PHY bis zum Transceiver. Blaue Pfeile: Datenfluss, rote Pfeile: Steuerleitungen, Blöcke mit durchgezogener Linie: Hardware Module, Blöcke mit gestrichelter Linie: Verilog Module

### 6.1 10B/9B Kodierung

Der Ethernet-PHY ist über RMII an das FPGA angeschlossen. Da *Carrier Sense* (CRS) und *Data Valid* (DV) im Multiplexverfahren in abwechselnden Taktzyklen auf einer Datenleitung liegen, ist jeweils ein DV-Bit pro vier Datenbits vorhanden. Jeweils acht Datenbits und zwei DV-Bits werden dann in ein 9 Bit Codewort codiert. Dies ist möglich, da bei nicht gesetztem DV-Bit die Datenbits ignoriert werden können. Theoretisch wäre hierfür auch eine 55B/45B Kodierung möglich, was deutlich effizienter wäre, allerdings würde sich eine höhere Gesamtlatenz ergeben, da die Anzahl der pro Zyklus übertragenen Bits ein Vielfaches der Länge beider Kodierer sein muss.

### 6.2 12B/16B Kodierung

Experimentell zeigte sich, dass ein hoher DC-Anteil im Datenstrom zu einer deutlich verbesserten Übertragung auf dem Funkkanal führt. Um dies sicherzustellen, werden lediglich die Symbole `01`, `10` und `11` verwendet. Mit acht dieser Symbole ergeben sich $3^8 = 6561$ verschiedene Codewörter, womit 12 Bit kodiert werden können. Dies ist erneut nicht die effizienteste Möglichkeit (dies wäre eine 19B/24B Kodierung), bietet allerdings aus dem obengenannten Grund die niedrigste Gesamtlatenz.

### 6.3 CRC

Das FPGA bietet keine Möglichkeit zur Taktrückgewinnung aus dem Datenstrom. Daher werden zwei identische Empfangskanäle verwendet, welche das Eingangssignal 180° Phasenverschoben zueinander abtasten. Damit ist sichergestellt, dass mindestens ein Empfangskanal den Datenstrom korrekt empfangen hat, vorausgesetzt auf dem Funkkanal sind keine Bitfehler aufgetreten. Die durch ungünstige Phasenlage bei der Abtastung entstandenen Bitfehler auf dem anderen Empfangskanal führen meistens zu illegalen Codewörtern, womit der gute Empfangskanal in vielen Fällen identifiziert werden kann. Um die Zahl an unerkannten Fehlern jedoch noch weiter zu reduzieren, wird zusätzlich ein *Cyclic Redundancy Check* (CRC) verwendet. Hierfür wurde folgendes primitives Polynom gewählt, da es bei der hier verwendeten Datenlänge von 468 Bits alle Fehler bis zu einer Hammingdistanz von 5 erkennt und bei einer Hammingdistanz von 6 nur 5292 Fehler unerkannt bleiben[Ko].

$$x^{32} + x^{30} + x^{28} + x^{27} + x^{25} + x^{19} + x^{14} + x^{11} + x^8 + x^7 + x^6 + x^5 + x^2 + x + 1 \qquad (3)$$

Der CRC-Wert wird über die 10B/9B Kodierten Daten berechnet, während diese in den Blockram geschrieben werden. Beim Senden wird der CRC-Wert an die Daten angehängt und ebenfalls 12B/16B Kodiert. Im Empfangskanal wird der CRC-Wert ebenfalls berechnet während die Daten in den Blockram geschrieben werden und am Ende des Datenpaketes überprüft. Stimmt der CRC-Wert nur bei einem Empfangskanal, so werden dessen Daten verwendet, ansonsten werden die Daten des ersten Empfangskanals verwendet.

### 6.4 Message Generator

Das Format der gesendeten Nachrichten ist in Abbildung 6 abgebildet. Alle Nachrichten haben einen 16-Bit Header, womit der Anfang einer Nachricht erkannt wird. Dieser ist in zwei Felder von je 8-Bit unterteilt, womit der Nachrichtentyp definiert wird. Jedes Feld kann eines von zwei Symbolen annehmen: $s_1$ = `10001110` oder $s_2 = \overline{s_1}$. Die Symbole können auch Zeitverschoben nicht in den Daten auftreten, womit der Anfang einer Nachricht eindeutig identifiziert werden kann. Eine Hello-Nachricht nutzt den Header $[s_1, s_1]$, eine Reply-Nachricht $[s_1, s_2]$ und eine Daten-Nachricht $[s_2, s_1]$. Der Message Detector erkennt anhand dieser Symbole den Anfang einer Nachricht und verschiebt die einkommenden Bits entsprechend.

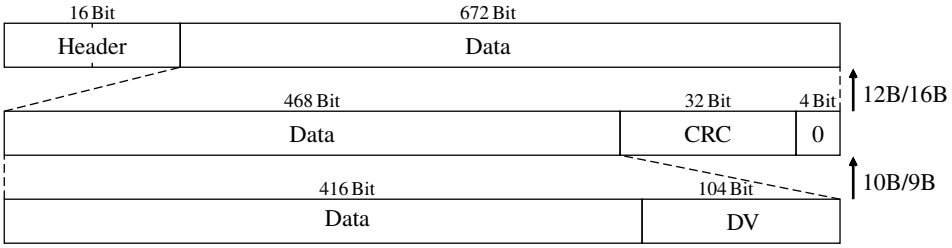



Abb. 6: Aufbau der gesendeten Daten nach allen Kodierungsschritten (nicht maßstabsgetreu)

Bei Daten-Nachrichten folgen nach dem Header die kodierten Daten. Dazu werden 416 Datenbits und 104 DV-Bits zunächst 10B/9B kodiert. Daran wird die 32-Bit CRC und 4 Bit Padding angehängt. All dies wird 12B/16B kodiert und an den Header angehängt, womit die Nachricht mit einer Gesamtlänge von 688 Bits entsteht. Bei Hello- und Reply-Nachrichten folgen auf den Header 672 0-Bits.

### 6.5 Buffer

Da die Referenztaktgeber nicht synchronisiert werden, laufen diese mit leicht unterschiedlichen Frequenzen. Dies führt in der zyklischen Verbindung nicht zu Fehlern, da beide Seiten aufeinander warten. Das Funksystem läuft damit jedoch mit dem langsameren der beiden Taktgeber, womit auf der Seite mit dem schnelleren Taktgeber nach einiger Zeit zu wenig Daten für die RMII Schnittstelle vorhanden sind. Die Daten werden daher in einem zwei Bit langen FIFO-Buffer gepuffert. Zwischen den Ethernet-Frames (DV = `0`) werden dann, wenn nötig 0-Bits eingefügt, um den Buffer gefüllt zu halten. Bei einem Referenztaktgeber mit einer Frequenzstabilität von ±25 ppm ist dies jedoch minimal alle 5000 Bytes nötig.

### 6.6 State Machine

Der Ablauf des Verbindungsaufbaus sowie der Datenübertragung wird durch eine State Machine gesteuert. Das Ablaufdiagramm ist in Abbildung 7 dargestellt.

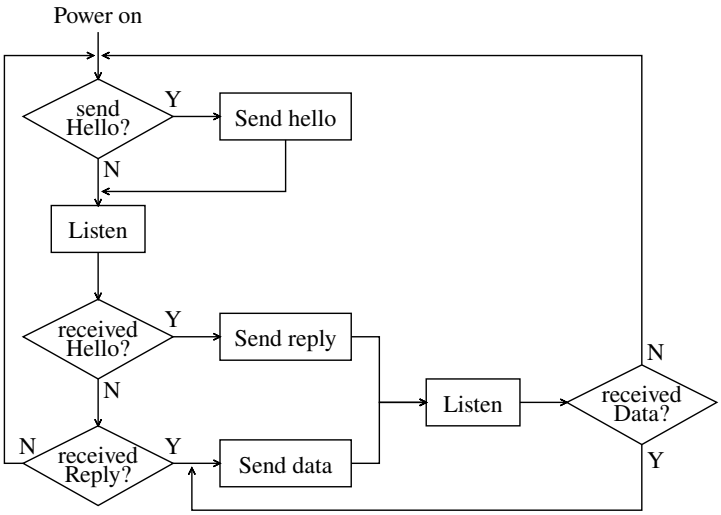


Abb. 7: Ablaufdiagramm der State Machine

Nach dem Einschalten beginnt das Gerät mit dem Verbindungsaufbau. Nach den oben beschriebenen Methoden wird entschieden, ob eine Hello-Nachricht gesendet wird. Danach wird in den Empfangsmodus umgeschaltet und auf eine Reply-Nachricht gewartet. Wird diese empfangen, wird ein Datenpaket gesendet. Wird stattdessen eine Hello-Nachricht Empfangen, wird eine Reply-Nachricht gesendet. Wird keine Nachricht empfangen, so wird der Verbindungsaufbau fortgesetzt. Wird nach dem Senden einer Reply-Nachricht oder eines Datenpaketes ein Datenpaket empfangen, so wird ein weiteres Datenpaket gesendet. Wird kein Datenpaket empfangen, gilt die Verbindung als unterbrochen und es wird ein erneuter Verbindungsaufbau versucht.

# 7 Latenz und Jitter

In diesem Abschnitt sollen Latenz und Jitter des Systems theoretisch betrachtet werden. Die Latenz ergibt sich aus der Länge des Buffers, der Datenübertragung und der Latenz des verwendeten PHYs [De19].

$$\Delta t = t_{PHY}+2\cdot t_{Buffer}+t_{Blockram}+t_{Data} = 460\,\text{ns}+2\cdot 20\,\text{ns}+4{,}16\,\mu\text{s}+610\,\text{ns} = 5{,}27\,\mu\text{s} \quad (4)$$

Für den Jitter des Systems sind hauptsächlich die nicht synchronisierten Referenztaktgeber verantwortlich. Durch die verwendeten Buffer kann sich die Latenz um bis zu 20 ns verringern, bevor dies durch Einfügen von 0-Bits korrigiert wird. Diese Latenzsprünge machen sich als Jitter bemerkbar. Der Jitter könnte durch Synchronisation der Referenztaktgeber über die Header in den Datenpaketen reduziert werden.

## 8 Zusammenfassung

In dieser Arbeit wurde eine kontaktlose Datenübertragung für Industrial Ethernet vorgestellt. Dabei wurden die Hardware, der Verbindungsaufbau und die Datenübertragung erläutert. Für den Verbindungsaufbau wurden zudem zwei verschiedene Möglichkeiten gefunden und optimiert. Das System erreicht eine feste Latenz von lediglich 5,27 µs und eignet sich damit für Echtzeit- und Motion-Control-Anwendungen.

## Literatur


[Ca] Carlsson, T.: Industrial networks keep growing despite challenging times, https://web.archive.org/web/20220607092027/https://www.hms-networks.com/news-and-insights/news-from-hms/2022/05/02/industrial-networks-keep-growing-despite-challenging-times, Accessed: 2022-08-10.

[De19] Devices, A.: Robust, Industrial, Low Power, 10 Mbps and 100 Mbps Ethernet PHY, https://www.analog.com/media/en/technical-documentation/data-sheets/ADIN1200.pdf, Accessed: 2022-08-10, 2019.

[In] International, P.: PROFINET Systembeschreibung - Technologie und Anwendung, https://www.profibus.com/index.php?eID=dumpFile&t=f&f=82431&token=056ec23a1f6860dc926064f796d49f75c7d72a76, Accessed: 2022-08-10.

[Ko] Koopman, P.: CRC Polynomial Zoo, https://web.archive.org/web/20220627144103/https://users.ece.cmu.edu/~koopman/crc/crc32.html, Accessed: 2022-08-10.

[Sc09] Scheiterer, R. L.; Na, C.; Obradovic, D.; Steindl, G.: Synchronization Performance of the Precision Time Protocol in Industrial Automation Networks. IEEE Transactions on Instrumentation and Measurement 58/6, S. 1849–1857, 2009.

# Entwurf eines einheitlichen Energieinformationsmodells für die Übertragung von Energieinformationen auf Basis von industriell genutzten Kommunikationsstandards

Leif-Thore Reiche[1], Maxim Runge[2], Karl-Heinz Niemann[2], Alexander Fay[1]

**Abstract:** Das Monitoring von Energiebedarfen und die aktive Beeinflussung der eingesetzten Energiemenge mit Hilfe eines Lastmanagements spielen in industriellen Anlagen eine zunehmend wichtige Rolle. Um diese Aufgaben zu erfüllen, können in Energiemanagementsystemen sogenannte Energiemanagementfunktionalitäten verwendet werden. Energiemanagement-funktionalitäten basieren dabei auf sogenannten Energieinformationen, die bidirektional zwischen der Feldebene und den Applikationen des Energiemanagementsystems ausgetauscht werden. Die Besonderheit der Energieinformationen besteht darin, dass diese aufgrund der unterschiedlichen industriell genutzten Kommunikationsstandards jeweils eigene Semantiken und Syntax aufweisen. Diese semantischen Unterschiede erschweren eine Vereinheitlichung der Energieinformationen und führen zu einem hohen Engineering-Aufwand, wenn Energieinformationen unterschiedlicher Kommunikationsstandards genutzt werden sollen. Um dieses Problem zu lösen und um den Engineering- Aufwand reduzieren zu können, stellt dieser Beitrag den Entwurf eines einheitlichen Energieinformationsmodells vor, mit dem die Energieinformationen verschiedener industriell genutzter Kommunikationsstandards vereinheitlicht werden können.

**Keywords:** Kommunikation von Energieinformationen, Energieprofile, heterogene Kommunikationsprotokolle, Energiemanagementsysteme

## 1 Einleitung

Um den Energiebedarf reduzieren und die Energieeffizienz der Produktion sukzessiv steigern zu können, sind industrielle Unternehmen angehalten sogenannte Energiemanagementsysteme (EnMS) entsprechend dem ISO-Standard 50001 einzuführen [ISO 50001]. Durch diese Art von System haben Unternehmen die Möglichkeit ein Verständnis für den eigenen Energiebedarf zu entwickeln und können dadurch Maßnahmen ableiten, die darauf ausgerichtet sind, langfristig einen energieeffizienteren Betrieb im Unternehmen zu gewährleisten. Grundsätzlich setzt sich ein Energiemanagementsystem aus einem organisatorischen und einem technischen Teil zusammen. Während beim organisatorischen Teil die Prozesse und Verfahren mit einem

[1] Helmut-Schmidt-Universität, Institut für Automatisierungstechnik, Holstenhofweg 85, 22043 Hamburg, Leif-Thore.Reiche@hsu-hh.de, https://orcid.org/0000-0003-2919-8148
Alexander.Fay@hsu-hh.de https://orcid.org/0000-0002-1922-654X
[2] Hochschule Hannover, Elektro- und Informationstechnik Fakultät I, Ricklinger Stadtweg 120, 30459 Hannover, Maxim.Runge@hs-hannover.de, https://orcid.org/0000-0002-7651-4126
Karl-Heinz.Niemann@hs-hannover.de

Bezug zum Thema Energie im Vordergrund stehen, werden beim technischen Teil alle hard- und softwaretechnischen Komponenten betrachtet, die im Zusammenhang mit dem Thema Energie stehen [BEG+19]. Dieser technische Teil wird auch als technisches Energiemanagementsystem (tEnMS) bezeichnet und ist im speziellen darauf ausgerichtet, verschiedene Funktionalitäten für das Energiemanagement bereitzustellen [WNF18]. Zum einen können diese Funktionalitäten für ein Energiemonitoring genutzt werden, um so energiespezifische Informationen messen, kommunizieren, speichern und auf den höher liegenden Ebenen des Automatisierungssystems (z. B. Betriebsleitebene) darstellen zu können. Zum anderen können die Funktionalitäten für ein Lastmanagement der Anlagen genutzt werden, wodurch der Energiebedarf der Produktion mittels spezieller Betriebsmodi aktiv beeinflusst werden kann [BAL12]. Sowohl für das Energiemonitoring als auch für das Lastmanagement werden sogenannte Energieinformationen genutzt, die zwischen den Geräten der Feldebene (z. B. Frequenzumrichter, Roboter) und den Applikationen des Energiemanagementsystems über industriell genutzte Kommunikationsprotokolle ausgetauscht werden. Damit der Austausch bis zur Betriebsleitebene gelingt, erfolgt auf der Steuerungsebene eine Übersetzung (Mapping) der Energieinformationen von einer Feldbus-basierenden Semantik auf eine auf Companion Spezifikation-basierenden Semantik mit Hilfe von spezifischen Energiemanagement-Programmen [RRN+22].

Bedingt durch die verschiedene Kommunikationsprotokolle einer heterogenen Feldebene liegt keine einheitliche Semantik und Syntax für die Energieinformationen vor, sodass sich das Problem ergibt, dass die Erstellung der Energiemanagement-Programme mit einem hohen Engineering-Aufwand verbunden ist [WÜR20].

Dieser Beitrag verfolgt daher das Ziel, den Entwurf eines universellen Engieinformationsmodells (UEIM) vorzustellen. Mit Hilfe dieses Informationsmodells können Energieinformationen sematischen einheitlich beschrieben werden, um so den Engineering-Aufwand für die Bereitstellung der Energieinformationen reduzieren zu können. Auch Aufwand eines Zugriffs auf die Energieinformationen aus Sicht der Energiemangement-Applikation wird reduziert. Das UEIM bezieht sich dabei auf bestehende Standards für die Kommunikation von Energieinformationen.

Zur Erläuterung des UEIM-Entwurfs ist der Beitrag wie folgt aufgebaut: In Kapitel 2 wird der aktuelle Stand der Technik der Energieinformationsübertragung beschrieben. Das darauffolgende Kapitel 3 zeigt zum einen was unter einem Energieinformationsmodell zu verstehen ist, zum anderen wird die methodische Vorgehensweise zur Erzeugung des UEIM vorgestellt. Kapitel 4 geht auf den Entwurf des UEIM ein und stellt insbesondere die Darstellung von Energie-Messinformationen im Informationsmodell vor. Abschließend folgen eine Zusammenfassung und ein Ausblick auf zukünftige Arbeiten.

## 2 Stand der Technik

Dieses Kapitel beschreibt, wie die Kommunikation von Energieinformationen mit Hilfe eines technischen Energiemanagementsystems erfolgen kann.

## 2.1 Parallel und integrierte technische Energiemanagementsysteme

Um Energieinformationen in industriellen Anlagen übertragen zu können, werden sogenannte technische Energiemanagementsysteme (tEnMS) eingesetzt. Nach [WÜR21] zählen zu diesen Systemen alle technischen Komponenten (Hardware, Software) die für das Energiemanagement benötigt werden. Die Übertragung der Informationen kann mit diesen Systemen laut [WNF+19] entweder parallel oder integriert erfolgen. In [NIWÜ20] wird die parallele und integrierte Übertragung von den Autoren wie folgt beschrieben: Bei einer parallelen Übertragung wird parallel zu dem bestehenden Kommunikationssystem ein weiteres System installiert, das ausschließlich für die Übertragung der Energieinformationen genutzt wird. Dieses zusätzliche System hat den Nachteil, dass weitere Kosten für Planung, Equipment und Wartung der Systeme entstehen. Neben der parallelen Übertragung können die Energieinformationen auch über die bestehenden Kommunikationssysteme übertragen werden. In der Praxis werden für die Übertragung der Energiedaten sogenannte Energieprofile eingesetzt. Energieprofile wie PROFIenergy [PRO21] für PROFINET, sercos Energy [SER18] für sercos III oder CIP Energy [ODV21] für das Common Industrial Protokoll (CIP) sind darauf ausgelegt, die vorhandenen Protokolle der Kommunikationssysteme um die Möglichkeit der Energieinformationsübertragung zu erweitern. Der Vorteil dieser Lösung liegt darin, dass keine Kosten für ein zusätzliches Kommunikationssystem anfallen, da die bestehenden Kommunikationssysteme weitergenutzt werden können. Dieser Beitrag geht von einer integrierten Energieinformationsübertragung aus.

## 2.2 Ablauf der Kommunikation von Energieinformationen

Bevor Energieinformationen in den Energiemanagement-Applikationen der Betriebsleitebene genutzt werden können, müssen diese Informationen so aufbereitet werden, dass die Energiemanagement-Applikationen die Semantik der Energieinformationen interpretieren kann. In Abbildung 1 ist die bidirektionale Kommunikation von Energieinformationen von der Feld- über die Steuerungs- bis in die Betriebsleitebene dargestellt. Quellen für die Energieinformationen sind die Geräte einer heterogenen Feldebene ①. Für die Aufnahme der Messinformationen verfügen die Geräte der Feldebene über eigene Messgeräte und Energiezähler [NIWÜ20]. Die aufgenommenen Messinformationen werden anschließend über unterschiedliche Ethernet- und nicht-Ethernet-basierende Kommunikationssysteme ②, wie Feldbussysteme (z. B. PROFINET [PRO18]), Punkt-zu-Punkt-Verbindungen (z. B. IO-Link [UWJ20] oder weitere Systeme zwischen der Feld- und der Steuerungsebene ausgetauscht [RRN+21]. Die verwendeten Kommunikationssysteme weisen jeweils eigene Semantiken für die Darstellung der Energieinformationen auf.

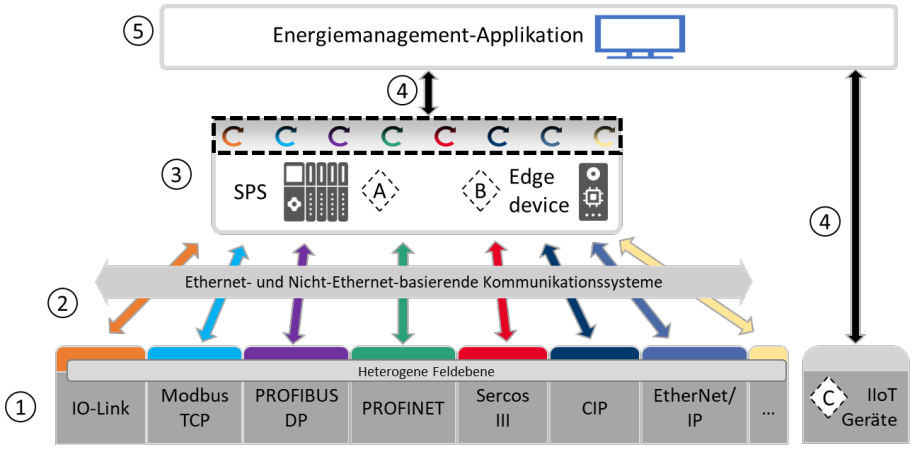



Abbildung 1 Kommunikation von Energieinformationen

Um die Energieinformationen verwenden zu können, müssen auf den speicherprogrammierbaren Steuerungen (SPS) (A) oder Edge-Geräten (B) der Steuerungsebene ③ für jedes Protokoll spezifische Energiemanagement- Programme erstellt werden. Diese Programme dienen dazu, ggf. das Datenformat der Energieinformationen anzupassen oder die Energieinformation zu skalieren, damit diese auf geeigneten Kommunikationsschnittstellen (z. B. OPC UA) abgebildet werden können [RRN+22]. Da der Engineering-Aufwand für die Erstellung dieser Programme sehr hoch ist, wurden für einzelne Protokolle eigene OPC-UA-Companion-Spezifikationen (z. B. PROFINET auf OPC UA [OPC21]) erarbeitet, durch die sich der Engineering-Aufwand reduzieren lässt. Von der Steuerungsebene können die Energieinformationen dann über eine Ethernet-basierte Schnittstelle ④ unter Verwendung von OPC UA an die Betriebsleitebene übertragen werden. Bedingt durch den Aufbau und die Anbindung an das Kommunikationsnetzwerk können IIoT-Geräte (C) die Energieinformationen direkt mit der Betriebsleitebene austauschen. Abschließend werden die Energieinformationen auf der Betriebsleitebene in den Energiemanagement-Applikationen ⑤ weiterverarbeitet, gespeichert, dargestellt und ausgewertet.

# 3 Methodisches Vorgehen zur Erzeugung des universellen Energieinformationsmodells

Der Stand der Technik zeigt auf, dass die Kommunikation von Energieinformationen mit den gegebenen Standards einen hohen Engineering-Aufwand nach sich zieht, wenn Energieinformationen aus unterschiedlichen Quellen der Feldebene entstammen und in unterschiedlichen Formaten vorliegen. In diesem Kapitel wird daher erläutert, wie dieser Aufwand mit Hilfe eines universellen Energieinformationsmodells (UEIM) reduziert werden kann. Dazu wird zuerst erklärt wo das UEIM verortet ist, was unter diesem

Informationsmodell zu verstehen ist und wie es funktioniert. Anschließend wird in den weiteren Unterkapiteln darauf eingegangen, welches methodische Vorgehen gewählt wurde, um das UEIM zu erzeugen.

## 3.1 Universelles Energieinformationsmodell

In Abbildung 2 ist dargestellt, wo das UEIM zu verorten ist. Beim UEIM handelt es sich um ein Informationsmodell, das Energieinformationen, die eine Relevanz für das Energiemanagement einer Produktion haben, eine semantisch eindeutige Bedeutung gibt. Das UEIM ist dabei technologieunabhängig.

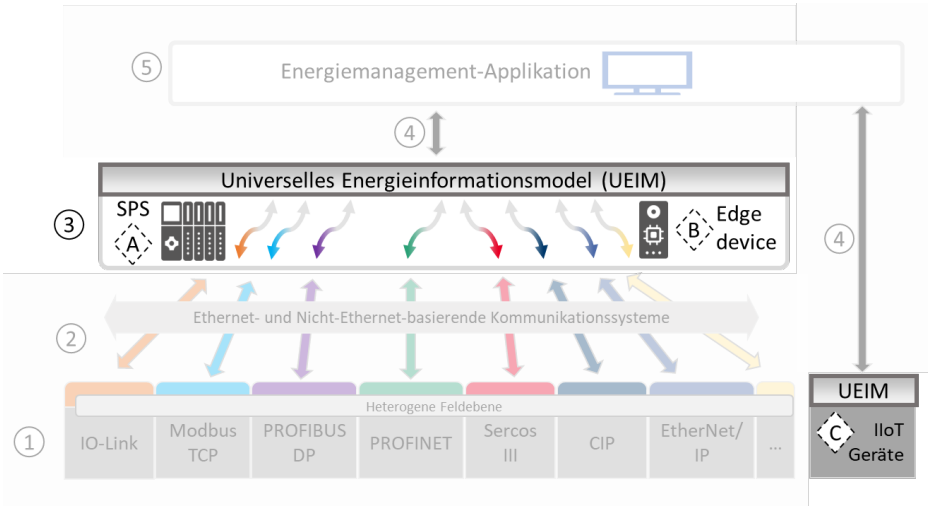



Abbildung 2 Energieinformationsmodell auf der Steuerungsebene

Mit Hilfe des UEIM ist es möglich Energieinformationen der Feldebene eindeutig zu interpretieren, unabhängig davon welchen Ursprung die Energieinformationen in der Feldebene haben. Wird das UEIM in den SPS oder Edge- Geräten der Steuerungsebene integriert, können die Energieinformationen ohne großen Mehraufwand mit einer einheitlichen Semantik zu den Energiemanagement-Applikationen auf der Betriebsleitebene übermittelt werden.

## 3.2 Methodik zur Modellierung des Energieinformationsmodells

Die Erzeugung des UEIM erfolgte mit Hilfe einer speziell ausgearbeiteten Methodik, die in Abbildung 3 dargestellt ist. Diese Methodik besteht aus 4 Einzelschritten, die sukzessive erarbeitet wurden.

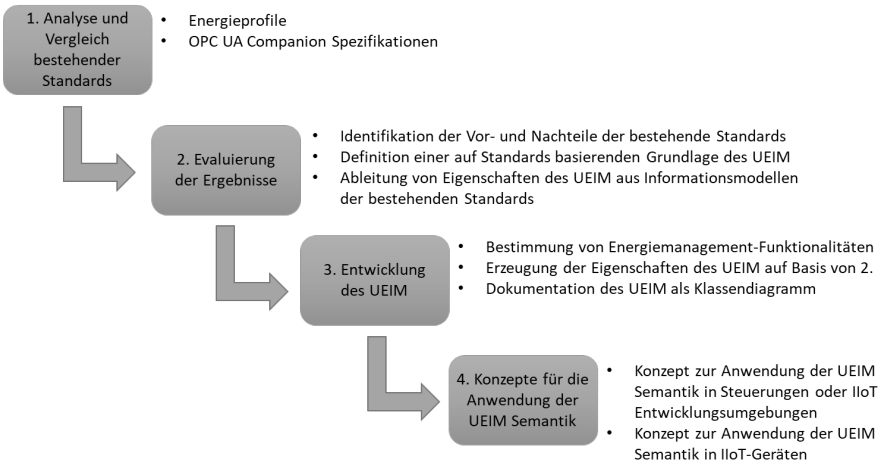



Abbildung 3 Methodisches Vorgehen zur Erzeugung des UEIM

Die Ausgangsbasis für das UEIM bildeten Standards zur Kommunikation von Energieinformationen (z. B. Energieprofile und OPC-UA-Companion-Spezifikationen), die in der Industrie bereits verwendet werden. Diese Standards wurden im ersten Schritt analysiert und anschließend miteinander verglichen. Im Rahmen der Analyse wurde unter anderem ermittelt, welche Energiemanagement-Funktionalitäten die einzelnen Standards bereitstellten und wie die Funktionalitäten semantisch beschrieben werden.

Anschließend wurden im zweiten Schritt die Ergebnisse aus Schritt 1 evaluiert. Hierbei konnten unter anderem Vor- und Nachteile der einzelnen Standards herausgearbeitet und die Frage geklärt werden, welche Energiemanagement-Funktionalitäten durch ein Energieinformationsmodell abgedeckt sein müssen. Auf Basis dieser Erkenntnisse wurden erste notwendige Eigenschaften des UEIM erarbeitet und somit eine Grundlage des UEIM definiert.

Der dritte Schritt wurde dazu genutzt, die erkannten Energiemanagement-Funktionalitäten zu bestimmen und das UEIM mit Hilfe eines Klassendiagramms auf Basis der Erkenntnisse aus Schritt 2 und den Inhalten aus [PRO22] zu modellieren.

Im abschließenden vierten Schritt wurden für die Anwendung der UEIM Semantik Konzepte ausgearbeitet, um die Funktion des Informationsmodells in Steuerungen, IIoT Entwicklungsumgebungen und in IIoT-Geräten zeigen zu können.

### 3.3 Vergleich von existierenden Standards

Um einen Einblick in die einzelnen Arbeiten des methodischen Vorgehens zur Erzeugung des UEIM zu erhalten, wird in Tabelle 1 ein Ausschnitt aus dem Vergleich von den existierenden Standards PROFIenergy, sercos Energy und CIP Energy und deren dazugehörenden Companion Spezifikationen (Schritt 1) vorgestellt.

Tabelle 1 Auszug aus dem Vergleich von existierenden Standards für die Energieinformationsübertragung

| OPC UA Companion Spezifikation des Energieprofils: |  |  |  |
|---|---|---|---|
| **Name des Messwerts:** | Active Energy Import/ Export | Supplied / Consumed Energy Odometer | Generated / Consumed Energy Odometer |
| **Datentyp:** | (optional) Signed Integer / Float32 / Float64 | FLOAT64 | Struct |

In der Tabelle sind links Eigenschaften der Energieprofile und deren OPC UA Companion Spezifikationen definiert, die miteinander verglichen werden. Der Vergleich der Messwertbezeichnung zeigt beispielsweise, dass alle Standards verschiedene Namen für den gleichen Messwert verwenden. Werden die Datentypen dieser Messwerte betrachtet, zeigen sich weitere Unterschiede. PROFIenergy kann hierfür die drei Datentypen *Signed Integer*, *Float32* und *Float64* verwenden. Sercos Energy besitzt für den gleichen Messwert nur den Datentyp *Float64* und CIP Energy wiederum verwendet einen eigens definierten strukturierten Datentyp (*Struct*).

Um im UEIM die Energieinformationen einheitlich beschreiben zu können, wurden die Eigenschaften der genannten Standards vereinheitlicht, sodass beispielsweise für das genannte Beispiel aus Tabelle 1 die Energieinformationen unter den Begriffen *Active Energy Consumed Odometer* und *Active Energy Generated Odometer* zusammengefasst wurden.

## 3.4 Integration des Energieinformationsmodells in der Praxis

Um Energieinformationen mit dem UEIM allgemein beschreiben zu können, wurden die erarbeiteten Inhalte in ein UML-Klassendiagramm überführt. Mit Hilfe des Klassendiagramms konnten die Energieinformationen anschließend durch Erzeugung einer Datei im XML-Schema in eine maschinenlesbare Form übertragen werden. Durch die Verwendung des XML-Schemas als Beschreibungsmittel für das UEIM ist es möglich die Energieinformationen in Entwicklungsumgebungen zu integrieren und diese somit auf Steuerungen oder Edge-Geräte verfügbar zu machen.

Somit können auf Steuerungen oder Edge-Geräten Messinstanzen für Geräte in der Feldebene angelegt werden, die über eine semantisch eindeutige Beschreibung von Energieinformationen verfügen.

# 4 Abbildung von Messinformationen

In diesem Kapitel wird das UEIM anhand von Teilausschnitten aus dem UML-Klassendiagramm vorgestellt. Abbildung 4 zeigt dafür in einem ersten Teilausschnitt die Klassen, mit denen die Bereitstellung der Energieinformationen (Messinformationen) erfolgen kann.

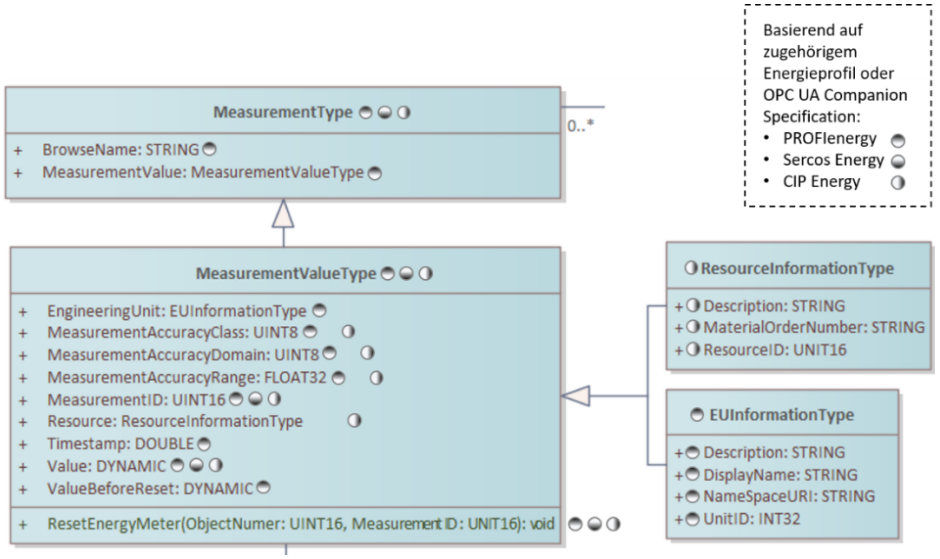



Abbildung 4: Modellierung des UEIM in einem UML-Diagramm

Für die Abbildung eines Messwertes *Value*, z. B. eines Energiezählwertes, muss eine Instanz der Klasse *MeasurementType* angelegt werden. Über das Attribut *BrowseName* kann einem Messpunkt im Feld eine eindeutige Bezeichnung zugewiesen werden. Die Bezeichnung des Messpunktes kann z. B. die Betriebsmittelkennzeichnung eines Sensors und die Benennung des Messorts enthalten. Die Klasseninstanz *MeasurementValue* der Klasse *MeasurementValueType* enthält detailliertere Informationen zu der abzubildenden Messung. Diese detaillierteren Informationen werden nachfolgend zusammengefasst:

Um dem abzubildenden Messwert (*Value*) eine Maßeinheit zuzuordnen, wurde das Attribut *EngineeringUnits* vom Typ *EUInformation* spezifiziert. In der dazugehörigen Unterklasse *EUInformationtype* sind weitere Attribute hinterlegt. Die *NameSpaceURI* dient zur Identifikation der Firma oder Normungsorganisation, die die Einheitsangaben spezifiziert haben. Jeder Einheit ist eine eindeutige *UnitID* (z. B. 4937544 für die Einheit der Messgröße Energie) zugewiesen. Weitere Angaben zur Einheit sind die Bezeichnung der Einheit in Volltext in *Description* (z. B. Kilowattstunde) und eine entsprechende Abkürzung in *DisplayName* (z. B. kWh).

Die Attribute mit dem Präfix *MeasurementAccuracy* werden zur Beschreibung von Messgenauigkeiten verwendet. Durch den Wert des Attributs *MeasurementAccuracyDomain* wird die Auswahl eines Messgenauigkeitstyps vorgenommen. Mit dem Messgenauigkeitstyp wird angegeben, ob sich die relative Messabweichung (*MeasurementAccuracyClass*) auf den Messbereichsendwert (Wert: 1) oder den aktuellen Messwert (Wert: 2) bezieht. Der zugehörige Messbereichsendwert kann ggf. im Attribut *MeasurementAccuracyRange* hinterlegt werden. Weitere Messgenauigkeitstypen ermöglichen die Norm-basierte Angabe von Messgenauigkeiten nach DIN EN 61557-12 (Wert: 3) [DIN EN 61557-12] oder DIN EN 50470-3 (Wert: 4) [DIN EN 50470-3].

Die *MeasurementID* ist eine Kennzahl, die zur eindeutigen Identifikation der zugehörigen Messgröße genutzt werden kann. Diese Kennzahl verweist auf die Zeile einer Messgrößentabelle, der Informationen wie Datentyp, Einheit oder weitere Messgrößenspezifische Informationen entnommen werden können. Die beschriebene Messgrößentabelle wurde auf der Basis der PROFIenergy-Spezifikation erstellt und um weitere Messgrößeneinträge aus den weiteren betrachteten Energieprofilen erweitert, sodass elektrische Messwerte wie beispielsweise elektrische Leistung, aber auch nichtelektrische Messwerte wie Wärmeströme abgebildet werden können.

Um eine eindeutige Zuordnung von Messungen zu einem Ressourcentyp zu ermöglichen, wird die Instanz *Resource* von *ResourceInformationType* erstellt. Dieses Merkmal des UEIM basiert auf der CIP Energy Spezifikation. Die dem Attribut *ResourceID* zugewiesene Identifikationsnummer legt einen eindeutigen Ressourcentyp, wie z. B. Druckluft fest. Die volle Bezeichnung der Ressource wird mit *Description* zugewiesen. Durch die *MaterialOrderNumber* kann die Bereitstellung einer Firmen-spezifischen Bestellnummer erfolgen.

Das Attribut *TimeStamp* kann verwendet werden, um einen gerätespezifischen Zeitstempel bereitzustellen, der mit Hilfe eines Zeitsynchronisationsprotokolls erstellt werden kann. Dadurch kann die Qualität des Zeitstempels individuell auf Geräteebene bestimmt werden.

Wenn der abzubildende Messwert einen Energiezähler darstellt, kann der Zählerwert in das Attribut *ValueBeforeReset* Attribut geschrieben werden, bevor er mit der bereitgestellten Methode *ResetEnergyMeter* zurückgesetzt wird.

Ein weiterer Teilausschnitt des UEIM ist in Abbildung 5 dargestellt. In der Abbildung sind Interface-Beispiele dargestellt, die zum automatischen Anlegen von Geräte-spezifischen Messknotenanordnungen geeignet sind. Die Klasse *MeasurementValueType* kann diese Interfaces optional implementieren, um die Bereitstellung mehrerer Messgrößen in einem Arbeitsschritt zu ermöglichen. Eine Messknotenanordnung kann zum Beispiel drei einzelne Phasenströme umfassen, die bei der integrierten Überwachung der Phasenströme eines intelligenten 3-phasigen Motorschutzschalters vorliegen. Die Interfaces wurden auf Basis der Inhalte der PROFIenergy Companion Spezifikation erstellt und um weitere Interfaces ergänzt.

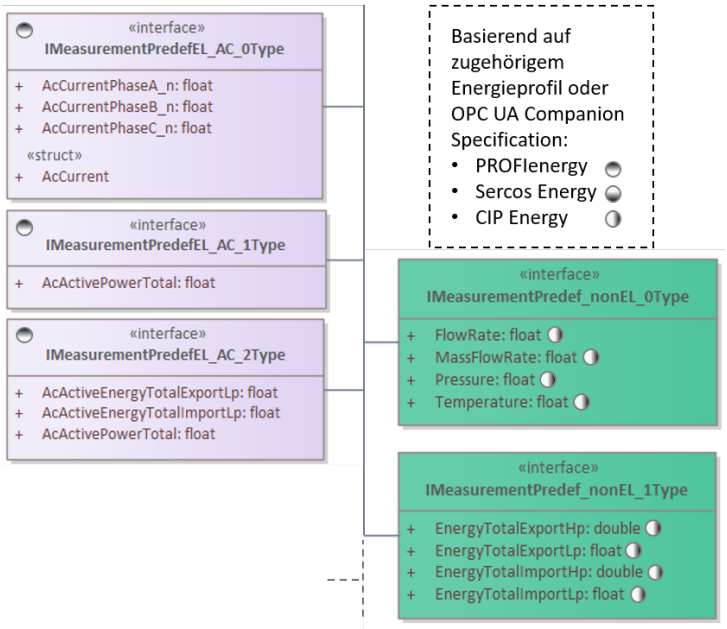



Abbildung 5: Interfaces zur Ergänzung des UML-Klassendiagramms

Zur Unterscheidung von Interfaces zur Bereitstellung von elektrischen Messgrößen (violett) und nicht-elektrischen Messgrößen (blaugrün) wurden die Interfaces mit Suffixen versehen. Das Suffix *El_AC_nType* wurde angelegt, um elektrische Messknoten-anordnungen aus Wechselstrom-Systemen bereitzustellen. Für die Vollständigkeit ist im UEIM dementsprechend auch das Suffix *El_DC_nType* für Gleichstrom-Systeme hinterlegt. Nicht elektrisch charakterisierte Interfaces werden mit dem Suffix *nonEl_nType* gekennzeichnet. Die hinzugefügten nicht-elektrischen Interfaces sind für Geräte wie Durchflussmesser (IMeasurementPredef_nonEl_0Type) und nicht-elektrische Energiezähler (IMeasurementPredef_nonEl_1Type) vorgesehen. Für die Bereitstellung von Energiezählerwerten stehen unterschiedlich große Datentypen zur Auswahl, um die Messwerte möglichst effizient bereitstellen zu können.

# 5 Zusammenfassung und Ausblick

Dieser Beitrag beschreibt den Entwurf eines einheitlichen Energieinformationsmodells (kurz UEIM), welches die Möglichkeit bietet, Energieinformationen aus einer heterogenen Feldebene zu vereinheitlichen, um so den Engineering-Aufwand für die Erstellung von Energiemanagement-Programme reduzieren zu können. Im Stand der Technik werden die Möglichkeiten aufgezeigt, wie Energieinformationen in einem Automatisierungssystem mit den bestehenden Systemen kommuniziert werden können. Des Weiteren wird erläutert, welches methodische Vorgehen gewählt wurde, um das

UEIM erzeugen zu können. Das UEIM zeichnet sich dadurch aus, dass es auf existierenden Standards basiert, da sich diese Standards in der Praxis zur Kommunikation von Energieinformationen bewährt haben. Im Anschluss wird die Modellierung des UEIM und Ausschnitte aus der Darstellung von Messinformation aufgezeigt, um so einen detaillierten Einblick in die Inhalte des Modells zu erhalten.
Nach Prüfung des UEIM durch verschiedene Arbeitskreise wird dieses Informationsmodell nun einer Joint Working Group, unter Beteiligung von VDMA, ZVEI, PI, ODVA und OPC Foundation zur Standardisierung in Form einer OPC-UA-Companion-Spezifikation zugeführt.
Zusätzlich wird in einem nächsten Schritt ein Demonstrator aufgebaut, der verschiedene energetische Anwendungsfälle aus dem automatisierungstechnischen Umfeld abbildet, um so die Funktion des UEIM bestätigen und evaluieren zu können.
Die Inhalte dieses Beitrags wurden im Rahmen des IGF-Vorhabens 21329-N (Projektname IoT_EnRG), das vom Bundesministerium für Wirtschaft und Klimaschutz (BMWK) auf Grundlage eines Beschlusses des Deutschen Bundestages gefördert wird, erarbeitet. Die Autoren bedanken sich für die finanzielle Unterstützung des Projekts.

## 6 Literaturverzeichnis

[BAL12] J. Balanowski: *Handbuch Lastmanagement: Vermarktung flexibler Lasten: Erlöse erwirtschaften-zur Energiewende beitragen: Deutsche Energie-Agentur,* 2012.

[BEG+19] J. Bränzel, D. Engelmann et al. (Hrsg.): *Energiemanagement: Praxisbuch für Fachkräfte, Berater und Manager*. 2., überarbeitete Auflage. Wiesbaden: Springer Vieweg (Springer eBook Collection), 2019.

[DIN EN 50470-3] DIN EN 50470-3. *Wechselstrom-Elektrizitätszähler - Teil 3: Besondere Anforderungen - Elektronische Wirkverbrauchszähler der Genauigkeitsklassen A, B und C,* 2022.

[DIN EN 61557-12] DIN EN 61557-12. *Elektrische Sicherheit in Niederspannungsnetzen bis AC 1 000 V und DC 1 500 V – Geräte zum Prüfen, Messen oder Überwachen von Schutzmaßnahmen,* 2016.

[ISO 50001] ISO 50001. *Energy management systems -Requirements with guidance for use,* 2018.

[NIWÜ20] K.-H. Niemann, A. Würger: *Potenziale von PROFIenergy in der Prozessindustrie,* 2020.

[ODV21] ODVA, Inc.: *Common Industrial Protocol*. 3.31. Aufl., 2021.

[OPC21] OPC Foundation: *OPC UA for Energy Management*. Release 1.00, 2021.

[PRO18] PROFIBUS and PROFINET International: *PROFINET Systembeschreibung: Technologie und Anwendung,* 2018.

[PRO21] PROFIBUS and PROFINET International: *Common Application Profile PROFIenergy: Technical Specification for PROFINET*. V1.3, 2021.

[PRO22] PROFIBUS Nutzerorganisation e.V.: *Informationsmodelle: PI-Strategie für Industrie 4.0,* 2022.

[RRN+21] L.-T. Reiche, M. Runge, K.-H. Niemann, A. Fay: *Communication of energy data in automation systems.* In: *Communication in automation: 17th IEEE International Workshop on Factory Communication Systems 2021: WFCS 2021: 9-11 June 2021, Linz, Austria.* Piscataway, NJ: IEEE, S. 45–48, 2021.

[RRN+22] M. Runge, L.-T. Reiche, K.-H. Niemann, A. Fay: *Wie können heterogene Prozessdaten automatisch in einem Energiemanagementsystem zusammengeführt werden?* In: Härle, Jäkel, Sand (Hrsg.): *Tagungsband AALE: Wissenstransfer im Spannungsfeld von Autonomisierung und Fachkräftemangel.* Leipzig: Hochschule für Technik Wirtschaft und Kultur, 2022.

[SER18] Sercos International e.V.: *Sercos III 1.3.2 Subprofile Sercos Energy*. 1.3.2-1.1, 2018.

[UWJ20] J.R. Uffelmann, P. Wienzek, M. Jahn (Hrsg.): *IO-Link - Band 1: Anwendung: Schlüsseltechnologie für Industrie 4.0*. 3. Auflage. Essen: Vulkan-Verlag GmbH, 2020.

[WNF+19] A. Würger, K.-H. Niemann, A. Fay, M. Gienke, M. Paulick: *Integriertes Anlagenengineering zur Erhöhung der Energieeffizienz.* atp magazin, Vol. 61, S. 70, 2019.

[WNF18] A. Würger, K.-H. Niemann, A. Fay: *Concept for an Energy Data Aggregation Layer for Production Sites A Combination of AutomationML and OPC UA.* In: *IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA) (*1), S. 1051–1055, 2018.

[WÜR20] A. Würger: *Automatisierte Generierung von Energiemanagementfunktionen auf Basis des PROFINET-Energieprofils: Dissertation,* 2020.

[WÜR21] A. Würger: *Optimizing the engineering of technical energy management systems.* In: Drath (Hrsg.): *AutomationML: The industrial cookbook.* 1. Auflage. Berlin: De Gruyter Oldenbourg (De Gruyter STEM), S. 539–554, 2021.

# 10BASE-T1L Plug-and-Play Architecture for I4.0 Field Devices over IO-Link

Victor Chavez [1], Jörg Wollert[2]

**Abstract:** The latest amendment to the Ethernet physical layer, known as IEEE 802.3cg or 10BASE-T1L, aims to solve the limitations of current physical mediums for industrial field devices. This standard overcomes current limitations for industrial protocols and their physical medium which have problems such as short-distance reach or reduced bandwidth for long-distance applications. In this paper, we propose an architecture to implement the 10BASE-T1L physical layer with field devices aligned to the Industry 4.0 vision. We propose the adaption of the IO-Link protocol, which works in a plug-and-play fashion. The exchange of information through 10BASE-T1L is then managed through the I4.0 asset administration shell. The last important part of this architecture is the correct interpretation of information and automatic classification of semantics from field devices to enable interoperable systems. Based on these components, we mention the requirements to realize this architecture and its implementation.

**Keywords:** 10BASE-T1L, interoperability, semantics, ontology, IO-Link, Industry 4.

## 1 Introduction

Field devices, such as sensors and actuators, are an integral part of any process that involves controlling or measuring its physical environment. As such in the industry, communication protocols and their physical layers are continuously evolving. One of the most recent physical layers that have been published in the IEEE 802.3cg Ethernet amendment. This specification increases the distance at which Ethernet can be transmitted over a single twisted pair of wires and allows transmission of power at a bandwidth of 10 Mbps.

In this sense, field devices can benefit from this standard as typical physical layers and industrial protocols have limitations regarding distance reach and bandwidth, which are

[1] Fachhochschule Aachen, Mechatronik und Eingebettete Systeme, Goethestr. 1, 52062 Aachen, chavez-bermudez@fh-aachen.de, https://orcid.org/0000-0001-6419-1641
[2] Fachhochschule Aachen, Mechatronik und Eingebettete Systeme, Goethestr. 1, 52062 Aachen, wollert@fh-aachen.de https://orcid.org/0000-0001-7576-1339

often proportionally inverse. To this matter, 10BASE-T1L Ethernet offers the possibility to overcome these limitations and reduce complexity in installations with a single pair of wires for transmission of power and data. As 10BASE-T1L is still a recent standard there are not many industrial applications in the market and as far as we know it has not been fully integrated into existent products.

In this paper, we propose a plug-and-play architecture for field devices based on the 10BASE-T1L physical layer and its integration into the well-known Industry 4.0 (I4.0) vision to enable seamless interoperability and reduction of heterogeneity at the information level. The plug-and-play functionality of this work is intended to work on the adaption of the single-drop digital communication system known as IO-Link [1]. The reason for this is that IO-Link offers already a plug-and-play communication protocol that includes diagnostics and parameterization of devices. Secondly, to achieve the I4.0 approach we propose the use of semantic technologies and their description of resources through the asset administration shell [2]

In the first section of this paper, we discuss some of the typical physical layers and their limitations compared to 10BASE-T1L Ethernet. Afterward, we mention some of the adaptions that are needed to use IO-Link over the Ethernet physical layer. In the next part of the paper, we describe how the integration of field devices into an I4.0 system takes place. In this section, we also discuss more in detail what is needed to use the asset administration shell and its integration with ontologies for field devices. The next topic of discussion is how this architecture could be implemented with off-the-shelf components and the limitations that need to be addressed. Lastly, we give our conclusions and future work to address the current research gaps.

## 2 Field-device limitations

In the industry, there are several commonly used physical layers and communication protocols for field devices. These technologies have evolved to suit the needs and requirements of the industry. However, one aspect that is still common among them is the limitations between distance reach, power transmission, and bandwidth. These three aspects are commonly tied to the properties of the physical layer and their topology. In Table 1, we show some of the most relevant protocols used for field devices to illustrate this better.

From this table, we can see how the maximum distance reached is typically limited to the maximum bitrate at which they transmit. For example, the CAN bus has a maximum distance of 1000 meters reached with the limitation of a bitrate of 50 Kbps. This also occurs similarly with Modbus RTU which is limited to 100 Kbps for a reach of 1200 meters. In comparison, IO-Link can achieve a maximum bitrate of 230.4 Kbps but is limited to a 20-meter reach. The HART protocol has the best distance reach but at the cost of a slower transmission rate. By looking at these properties we can identify a pattern where transmission speeds and distance reach is inversely proportional.

Table 1 Typical Field devices and their properties

| **Protocol** | **Characteristics** | | | |
|---|---|---|---|---|
| | ***Maximum Distance*** | ***Maximum Bitrate*** | ***Data transmission wires*** | ***Power over data Line (PoDL)*** |
| IO-Link | 20 m | 230.4 kbps | 1 | No |
| CAN | 1000 m [a] | 1 Mbps | 2 | No |
| HART | 3000 m [3] | 1.2 Kbps | 2 | Yes |
| Modbus RTU (RS4-85) | 1200 m [4] [b] | 10 Mbps | 2 | No |
| AS-Interface | 300 m [5] | 167.5 Kbps | 2 | Yes |

[a.] At a bitrate of 50 Kbps

[b.] At a bitrate of 100 Kbps

Another important point regarding, field devices is that not all of them can transmit power over their data lines. Only two of the 5 explored options transmit power over their data line (PoDL) at the expense of slower transmission rates. As it can be seen, these field devices have a tradeoff between bitrate, and power transmission as well.

## 3 IO-Link as a plug-and-play mechanism

From the last section, we saw some of the current limitations that exist for field devices. If these technologies were adapted to 10BASE-T1L we could overcome these problems. For our proposal, we propose a plug-and-play mechanism for field devices over 10BASE-T1L. From the list of options that we explored in the last section, we decided that IO-Link is the most suited to be adapted over 10BASE-T1L Ethernet. IO-Link already includes a well-defined plug-and-play mechanism for point-to-point field devices and is well known in the industry.

The main adaptions that we see are necessary for IO-Link is the initialization of communication, metadata exchange, and cycle times. IO-Link consists of a minimum of two components. An IO-Link Master, which initiates communication, and an IO-Link device, typically a sensor or actuator. From an electrical point of view, IO-Link is a 24-volt DC serial communication protocol. To initiate an exchange, the IO-Link Master sends a wakeup pulse over the data line which is detected and allows any IO-Link Device to be

ready for a request. This first part can be addressed by detecting the link status of the Ethernet physical layer.

The next part that needs to be addressed is the exchange of metadata which includes data such as vendor identification number, device number, and serial number. This information is transmitted on a sequential basis due in part due to the speed rate and maximum payload size limitations of IO-Link. A way to optimize and adapt this is proposed by transmitting one single ethernet frame with the metadata which consists of a maximum of 16 bytes, known as the direct parameter page 1.

The last part that needs more attention is the minimum cycle time for synchronous data exchange (e.g., sensor data, control signals). First, we need to calculate the round-trip latency to know what could be the minimum theoretical time to communicate with an IO-Link device over ethernet. The theoretical roundtrip time, in this case, can be calculated as follows:

$$FS = (EO + IOLP + IOLH) \quad (1)$$

$$Tx_{byte} = 8/Tx_{bit} \quad (2)$$

$$RTT = (2 * FS) * Tx_{byte} \quad (3)$$

Where:

FS = Ethernet frame size in bytes

EO= Ethernet overhead size in bytes (38 bytes)

IOLP = IO-Link Payload size (37 to 1491 bytes)

IOLH = IO-Link header (9 bytes)

RTT= Round trip time in seconds

$Tx_{bit}$= Transmission Time of one bit over 10BASE-T1L

$Tx_{byte}$= Transmission Time of one byte over 10BASE-T1L

Equation (3) defines RTT based on the transmission speed of the physical medium (10 Mbps) and the total amounts of frames to transmit. As the only value that can be changed is the size of the payload IOLP, we have calculated the theoretical RTT for different IOLP values and compiled the results in Table 2. The results show that under 578 bytes of payload, the round-trip time is below the 1-millisecond range and theoretical cycle times up to 13.4 microseconds could be achieved. If the device needs to transmit more than 1 KB of cyclic data, the maximum payload could be used and achieve less than 3 milliseconds of cycle time.

Table 2 Round Trip Time for different IO-Link Payload sizes

| IOLP Size in bytes | Round trip time In microseconds |
|---|---|
| 37 | 134.4 |
| 100 | 235.2 |
| 578 | 1000 |
| 1000 | 1675.2 |
| 1491 | 2460.8 |

# I4.0 Field device description

After discussing the physical layer details for data exchange for devices it is required to explain in detail how the data should be processed to achieve interoperability and follow the latest I4.0 guidelines. One of the core aspects of the I4.0 is the seamless exchange of information between devices and the management of information in a hardware-agnostic manner. For this purpose, in the last years, the asset administration shell has been developed for the exchange of metadata, services, and identification of physical and digital resources. The asset administration shell (AAS) proposes an I4.0 description of assets (e.g., sensors, actuators, machines) that offer an interoperability model that can be implemented for industrial applications [6][7][8].

The description offered by the AAS allows a generic way of developing sub-models for the different domains that a resource can offer. For example, a sensor can contain information about its electrical and mechanical characteristics, communication properties, and services that it offers. From a communication perspective, there is a need to model the functional behavior of the device and provide an interface that can be accessed independently of the manufacturer. At the moment, there exist some AAS templates for a general description of information such as technical data [9], and manufacturer information [10] but there is not a template to describe the functional behavior of field devices yet.

The most similar concept to describe this is the so-called capabilities submodel element of the AAS [11]. This submodel allows to semantically relate the capability of the asset to a skill, which is the detailed description of the asset function. This capability requires to be further expanded to generically categorize functions and skills of field devices.

The generic description of the functional behavior of field devices is also known as device profiles [12]. Currently, field device communication protocols such as CANOpen, IO-Link, or Profinet, have several device profiles for field devices. However, these profiles were created by different organizations and are not interoperable among different

protocols. An essential part of this architecture is to develop a generic device profile for field devices that can describe general characteristics and enables an interoperable I4.0 interface

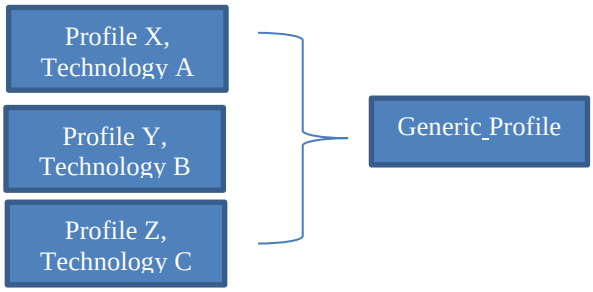



Fig. 1 Generalization of device profiles

## 4 Information management

The description of I4.0 field devices requires a mechanism to manage the assets and can update their information in real-time. Currently, there are two options to manage I4.0 assets. The first one is developed by the Industrial Digital Twin Association (IDTA), known as the AASX server [13]. The second option is the BaSyx middleware [14]. The option from the IDTA is a server that hosts I4.0 assets and has a REST, OPC UA, and MQTT interface. As information management is expected to be automated, we have chosen the BaSyx middleware, as it not only offers an AAS server but has a software development kit to programmatically generate I4.0 asset descriptions and integration to its server.

The information management can be implemented by having an agent [15]–[17] that links the low-level IO-Link 10BASE-T1L layer and the I4.0 interface assets. As new devices could be integrated into this plug-and-play architecture the I4.0 agent should be capable of generating a field device description from the IO-Link description known as IODD [18], and a knowledge database. This knowledge database which stores an unambiguous conceptualization of information is also called an ontology [19]. An ontology could describe relationships between I4.0 capabilities and field device descriptions. An ontology reasoner could infer dynamically capabilities of new devices and integrate them into the BaSyx Middleware to create a generic I4.0 asset.

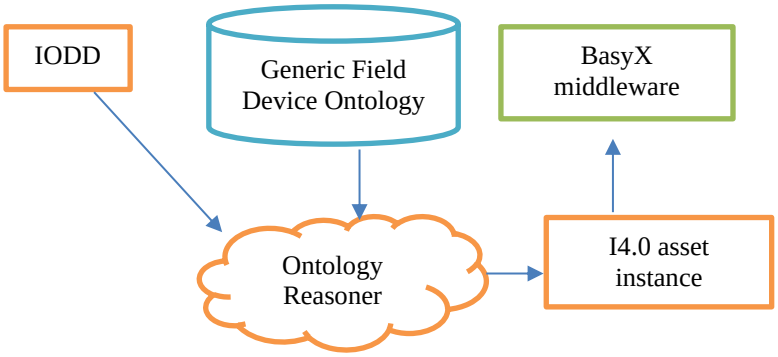



Fig. 2 Generation of I4.0 assets

An overview of the management of assets and communication exchange between field devices is shown in Figure 3. The exchange of information between the real-time level and the I4.0 interface occurs through the I4.0 AAS agent. The agent is in charge of managing any new connected devices, the description of its capabilities, and access through the communication stack. Integration with external systems or applications happens using an AAS standardized interface. At the moment there an HTTP/REST API is specified for the AAS but in subsequent versions of the API, other technologies such as OPC UA or MQTT will be supported [20].

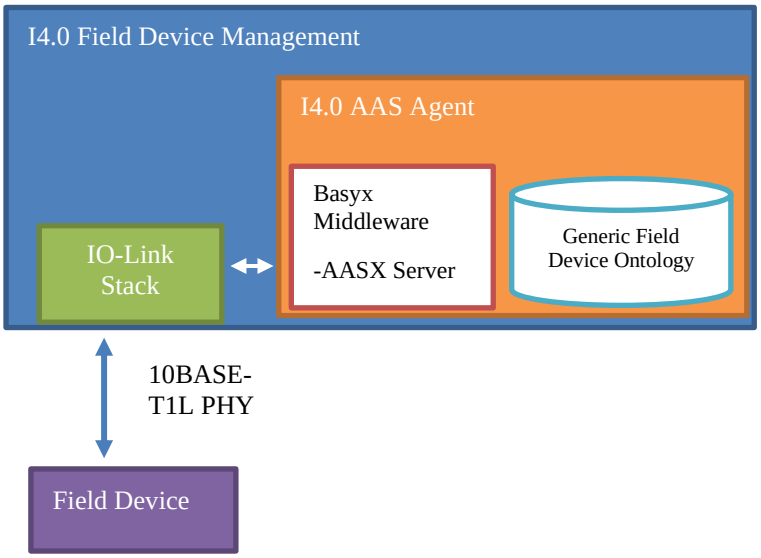



Fig. 3 Overview of I4.0 Field Device Architecture

## 5 Challenges and limitations

The proposed architecture has inherent challenges and limitations that should be addressed to develop its implementation. Real-time Ethernet communication requires a dedicated processor that can communicate with the field device in the range of microseconds up to 3 milliseconds. The second requirement is the agent that manages the I4.0 information. The BasyX middleware is intended to work with general operating systems and can be integrated into different ecosystems. One of the critical tasks is to integrate the real-time stack with a general operating system. Solutions such as processors that can have a general operating system (e.g., Linux) and a co-processor could alleviate this situation [21].

A limitation that we currently find is that the current implementations of the AAS are not intended for real-time systems. The AAS specification allows not only to get information on currently registered resources but to interact with them and a dedicated channel for real-time operations is needed. A possible solution for this could be an asynchronous channel such as the HTTP/REST API to retrieve information from the assets and a dedicated synchronous channel for real-time operations with a reserved bandwidth over the physical layer of choice.

## 6 Conclusions and future outlook

In this paper, we have explored an architecture to develop I4.0 field devices with the 10BASE-T1L Ethernet physical layer. Through the adaption of the IO-Link communication protocol, we intend to achieve plug-and-play devices that are managed by an I4.0 AAS agent. The management of information is done by developing a generic field device profile that intends to relate current field device profiles from protocols such as IO-Link, Profinet, and CANOpen. This profile should be an unambiguous conceptualization of knowledge generated through an ontology. The I4.0 AAS agent is intended to be based on the BaSyx middleware and orchestrate the generation of I4.0 assets with ontologies that describe capabilities and skills available for field devices.

The current implementation of the AAS has certain limitations that should be addressed to develop complete real-time I4.0 solutions. For the current proposal, we intend to use a dedicated co-processor for the IO-Link 10BASE-T1L communication stack and the main processor that could use a general operating system such as Linux.

Our next goals for the realization of this topic are to develop a generic field device profile ontology based on current technologies, the implementation of the IO-Link 10BASE-T1L based on our current rapid-prototyping IO-Link framework, and research more about real-time capabilities for the AAS and how they can be integrated to manage assets remotely [6].

# References


[1] IO-link Community, “IO-Link Interface and System Specification v1.1.2.” IO Link Community, Karlsruhe, 2013.

[2] Bundesministerium für Wirtschaft und Energie (BMWi), “Details of the Asset Administration Shell Part 1 - The exchange of information between partners in the value chain of Industrie 4.0 (Version 2.0).” [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/Details-of-the-Asset-Administration-Shell-Part1.html

[3] Emerson, “Avoiding HART loop interference when using the Smart Wireless THUM ™ Adapter,” 2011. https://www.emerson.com/documents/automation/white-paper-avoiding-hart-loop-interference-when-using-smart-wireless-thum-adapter-en-88318.pdf (accessed Mar. 01, 2022).

[4] Maxim Integrated, “Application Note 3884: How far and how fast can you go with RS-485?,” 2006. https://pdfserv.maximintegrated.com/en/an/AN3884.pdf (accessed Mar. 01, 2022).

[5] AS-International Association e.V., “AS-Interface Technology.” https://www.as-interface.net/en/technology (accessed Mar. 01, 2022).

[6] B. Lv, Y. Zhang, F. Yang, and J. He, “Edge Asset Management based on Administration Shell in Industrial Cyber-Physical Systems,” in *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, Oct. 2020, pp. 3817–3822. doi: 10.1109/IECON43393.2020.9255293.

[7] S. Cavalieri and M. G. Salafia, “Asset Administration Shell for PLC Representation Based on IEC 61131–3,” *IEEE Access*, vol. 8, pp. 142606–142621, 2020, doi: 10.1109/ACCESS.2020.3013890.

[8] J. Fuchs, J. Schmidt, J. Franke, K. Rehman, M. Sauer, and S. Karnouskos, “I4.0-compliant integration of assets utilizing the Asset Administration Shell,” in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2019, pp. 1243–1247. doi: 10.1109/ETFA.2019.8869255.

[9] Platform Industrie 4.0, “Submodel Templates of the Asset Administration Shell - Generic Frame for Technical Data for Industrial Equipment in Manufacturing (Version 1.1) .” https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/Submodel_Templates-Asset_Administration_Shell-Technical_Data.html (accessed Aug. 01, 2022).

[10] Plattform Industrie 4.0, “ZVEI Digital Nameplate for industrial equipment (Version 1.0)”, Accessed: Aug. 01, 2022. [Online]. Available: https://www.zvei.org/fileadmin/user_upload/Presse_und_Medien/Publikationen/2020/December/Submodel_Templates_of_the_Asset_Administration_Shell/Submodel_Templates-Asset_Administration_Shell-ZVEI-digital_nameplate.pdf

[11] A. Bayha, J. Bock, B. Boss, and C. Diedrich, “Describing Capabilities of Industrie 4.0 Components,” 2020. [Online]. Available: www.bmwi.de

[12] IEC, “IEC TR 62390:2005 Common automation device - Profile guideline.”

[13] IDTA, “C# based server for AASX packages .” https://github.com/admin-shell-io/aasx-server (accessed Aug. 01, 2022).

[14] Bundesministerium für Bildung und Forschung, “Eclipse BaSyx.” https://www.eclipse.org/basyx (accessed Aug. 01, 2022).

[15] A. Lopez, O. Casquero, E. Estevez, P. Leitao, and M. Marcos, “Towards the generic integration of agent-based AASs and Physical Assets: a four-layered architecture approach,” in *2021 IEEE 19th International Conference on Industrial Informatics (INDIN)*, Jul. 2021, pp. 1–6. doi: 10.1109/INDIN45523.2021.9557568.

[16] A. Lopez, O. Casquero, and M. Marcos, “Design patterns for the implementation of Industrial Agent-based AASs,” in *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*, May 2021, pp. 213–218. doi: 10.1109/ICPS49255.2021.9468129.

[17] L. Sakurada and P. Leitao, “Multi-Agent Systems to Implement Industry 4.0 Components,” in *2020 IEEE Conference on Industrial Cyberphysical Systems (ICPS)*, Jun. 2020, pp. 21–26. doi: 10.1109/ICPS48405.2020.9274745.

[18] I.-L. Consortium, “IO-Link Device Description v1.1.” Karlsruhe, 2011.

[19] N. Guarino, D. Oberle, and S. Staab, “What Is an Ontology?,” in *Handbook on Ontologies*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 1–17. doi: 10.1007/978-3-540-92673-3_0.

[20] Bundesministerium für Wirtschaft und Energie (BMWi), “Details of the Asset Administration Shell Part 2 – Interoperability at Runtime – Exchanging Information via Application,” 2021, [Online]. Available: https://www.plattform-i40.de/PI40/Redaktion/EN/Downloads/Publikation/Details_of_the_Asset_Administration_Shell_Part2_V1.html

[21] STMicroelectronics, “Coprocessor Management Overview STM32 MPU.” https://wiki.st.com/stm32mpu/wiki/Coprocessor_management_overview (accessed Aug. 01, 2022).

# Nahfeldmesstechnik zur Designoptimierung und Bewertung von IoT-Komponenten

Dominik Schröder[1], Sven Lange[2], Christian Hedayat[3], Ulrich Hilleringmann[4], Harald Kuhn[5]

**Abstract:** Um drahtlose Kommunikationssysteme zu charakterisieren und zu bewerten, gibt es verschiedenste Möglichkeiten. Nahfeldmessungen werden dabei vor allem zum Auffinden von EMV-Störquellen und zur Bewertung der Abstrahleigenschaften verwendet. Mit phasenbezogenen Nahfeldmessungen können Nahfeldquellen erzeugt werden, die in Simulationen für weiterführende Untersuchungen verwendet werden können. Bei der sogenannten HUYGENS-Box Methode werden alle tangentialen Feldkomponenten um das Testobjekt herum erfasst. In diesem Beitrag wird zunächst ein Überblick über die Grundlagen der Nahfeldmesstechnik gegeben und einfache Messergebnisse von Testobjekten im 2,4 GHz Bereich mit Simulationen verglichen. Darauf aufbauend wird die HUYGENS-Box Methode anhand von IoT-Komponenten dargestellt und bewertet. Untersucht werden die Fernfeld-Extrapolation und der Einfluss der Komponenten auf ihre Umgebung. Weitere, vielfältige Einsatzmöglichkeiten werden abschließend skizziert.

**Keywords:** Nahfeldmessung; IoT; Nahfeld-zu-Fernfeld; Feldsimulation; HUYGENS-Box

## 1 Einleitung

Mit der allgegenwärtig fortschreitenden Digitalisierung ziehen immer mehr elektrische Geräte in den Alltag der Internet-of-Things (IoT) ein. Vor allem in der Industrie werden neue Maschinen miteinander vernetzt und alte Geräte über Retro-Fitting aufgewertet und angebunden. Dadurch entstehen mehr und mehr drahtgebundene und drahtlose Kommunikationswege auf dem gleichen, begrenzten Raum.
Um Störungen zwischen verschiedenen Systemen zu minimieren, muss die elektromagnetische Verträglichkeit (EMV) nachgewiesen werden. Zum Nachweis der EMV-Konformität werden standardmäßig Messungen in geschirmten Fernfeldkammern durchgeführt. Dort wird geprüft, ob das Testobjekt die vorgegebenen Grenzwerte einhält. Hält das Messobjekt die entsprechenden Grenzwerte ein kann es eingesetzt werden. Fällt es jedoch durch, ist der Ursprung der Störquelle häufig unbekannt und schwer aufzufinden. Simulationen des Gesamtsystems zur Fehlersuche aufgrund des hohen Rechenaufwandes kaum möglich.

[1] Universität Paderborn, Paderborn, Deutschland schroeder@sensorik.uni-paderborn.de
[2] Universität Paderborn, Paderborn, Deutschland lange@sensorik.uni-paderborn.de
[3] Fraunhofer ENAS, Paderborn, Deutschland christian.hedayat@enas-pb.fraunhofer.de
[4] Universität Paderborn, Paderborn, Deutschland hilleringmann@sensorik.upb.de
[5] Fraunhofer ENAS, Chemnitz, Deutschland harald.kuhn@enas.fraunhofer.de

Eine alternative Methode zu Fernfeldmessungen bietet die Nahfeldmesstechnik. Bei Nahfeldmessungen werden elektrische und magnetische Felder an mehreren Punkten wenige Milli- oder Zentimeter über dem Testobjekt erfasst. Vorteile von Nahfeldmessungen sind dadurch vor allem die örtlich hohe Auflösung der Felder direkt über dem Objekt, um Quellen von EMV-Störungen direkt ausmachen zu können [Ha16].
Durch das sogenannte Oberflächen-Äquivalenz-Prinzip können die Nahfeldmessungen als Quelle für weiterführende Simulationen verwendet werden [La20]. Somit können aus den Nahfelddaten die Fernfeld-Daten berechnet und EMV-konforme Messungen abgeschätzt werden. Außerdem kann die gemessene Feldquelle für Simulationen zur Bestimmung der Wechselwirkung mit seiner Umgebung verwendet werden. Hier können Kopplungen zu anderen Geräten oder die Abstrahlung in den Raum simuliert und beurteilt werden. Denn gerade in industriellen Umgebungen und Anwendungen ist eine zuverlässige Kommunikationsverbindung und ein minimaler Störeinfluss zwischen den verschiedenen Teilnehmern wichtig.

## 2 Grundlagen der Nahfeldmesstechnik

### 2.1 Nahfeld- und Fernfeld-Definition

Fernfeldmessungen werden meistens im Abstand von 3 m zum Testobjekt (DUT) durchgeführt, während Nahfeldmessungen in unmittelbarer Nähe des DUTs erfolgen. Daher wird zuerst betrachtet wo das Fernfeld endet und das Nahfeld beginnt. Grundsätzlich kann der Raum um ein elektronisches System in drei Teilbereiche aufgeteilt werden: reaktives Nahfeld, strahlendes Nahfeld und Fernfeld. Im reaktiven Nahfeld stehen die elektrischen Strukturen in starker Wechselwirkung miteinander. Das strahlende Nahfeld, auch FRESNEL-Bereich genannt, ist der Übergangsbereich zwischen dem reaktiven Nahfeld und dem Fernfeld. Darüber hinaus befindet sich das Fernfeld, auch FRAUNHOFER-Region genannt.
Die verschiedenen Bereiche sind in Abb. 1 dargestellt. Der genaue Übergang zwischen den einzelnen Regionen ist nicht eindeutig bestimmbar, weshalb es auch unterschiedliche Definitionen in der Literatur gibt [Ta07, Ya86]. Eine häufig verwendete Definition wird im folgenden dargestellt, wobei $D$ die Länge des abstrahlenden Elements, $R$ der Abstand zum Prüfling und $\lambda$ die Wellenlänge ist [Ya86]. Das reaktive Nahfeld überwiegt im Bereich

$$R < 0.62\sqrt{\frac{D^3}{\lambda}} \tag{1}$$

gefolgt vom strahlenden Nahfeld, begrenzt zwischen

$$0.62\sqrt{\frac{D^3}{\lambda}} \leq R < \frac{2D^2}{\lambda} \tag{2}$$

und schließlich beginnt das Fernfeld ab

$$\frac{2D^2}{\lambda} \leq R, \tag{3}$$

welcher FRAUNHOFER-Abstand genannt wird [SJ17].

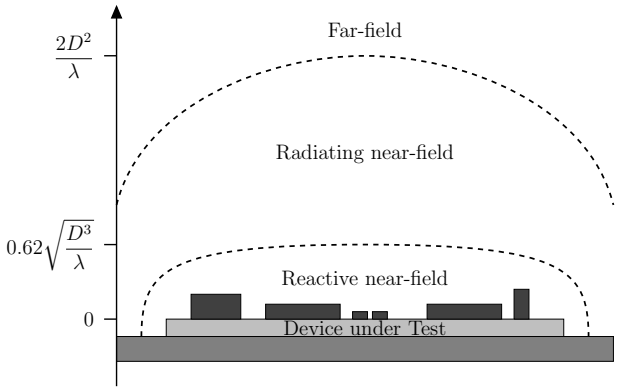



Abb. 1: Veranschaulichung der Nah- und Fernfeldregionen über einem Testobjekt

### 2.2 Nahfeldscanner-System

Im folgenden wird kurz darauf eingegangen wie ein grundlegendes Nahfeldmesssystem aussieht und welcher Nahfeldscanner für die dargestellten Messungen verwendet wird.
Der Begriff *Scanner* bezieht sich darauf, dass die elektromagnetischen Felder an verschiedenen räumlichen Punkten um den Prüfling herum erfasst werden. Dazu ist ein Nahfeldscanner mit einem mechanischen Positioniersystem ausgestattet, um die Nahfeldsonde oder das DUT selbst von einem Messpunkt zum nächsten zu bewegen. Mit einem solchen System können sowohl das elektrische als auch das magnetische Feld in allen drei Raumrichtungen auf äquidistanten, beliebigen oder adaptiven [Cl18, 35-58] Flächen gemessen werden. Gängige Oberflächen sind kugelförmig [NFG12], zylindrisch [QSE13] oder kubisch [RMH09].
Neben dem Positionierungssystem besteht ein Nahfeldscanner aus verschiedenen Nahfeldsonden, dem Messempfänger und einem Steuergerät. Um Amplitude und Phase in jedem Raumpunkt erfassen zu können sind mindestens zwei Messkanäle notwendig, um einen Phasenbezug herzustellen. Um das Referenzsignal zu erhalten, gibt es zwei grundlegende Konzepte. Entweder kann eine relativ zum DUT ortsfeste Referenzsonde eingesetzt werden. Das kann für jede Art von Prüfling durchgeführt werden. Oder das Referenzsignal wird vom DUT selbst erzeugt. Dies ist nur möglich, wenn das Signal des DUTs zugänglich ist und in einem periodischen Verhältnis zur gemessenen Emissionen steht. Der grundlegende Aufbau eines Nahfeldscanners ist in Abb. 2 dargestellt.
Die Messungen im praktischen Teil dieser Arbeit werden mit dem dem Nahfeldscanner NFS3000 durchgeführt [Ha16] . Der NFS3000 wurde am Fraunhofer-Instituts für Elektronische Elektronische Nanosysteme ENAS in Paderborn zusammen mit dem Fachgebiet Sensorik der Universität Paderborn entwickelt. DUTs können bis zu einem Volumen von

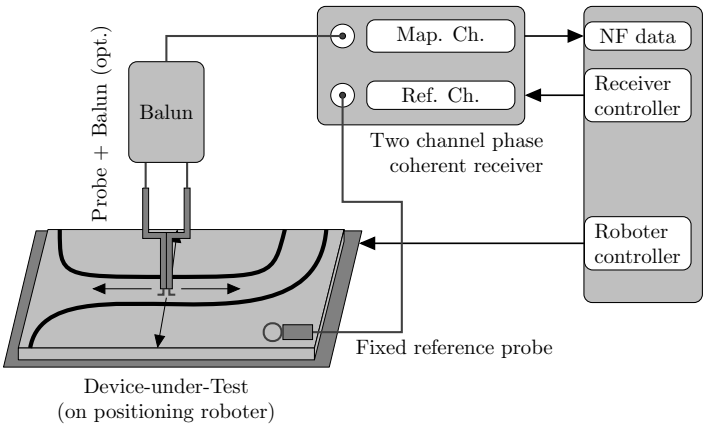



Abb. 2: Schematischer Aufbau eines einfachen Nahfeldmesssystems

500 mm × 800 mm × 500 mm (x, y, z) bei einer räumliche Schrittweite von 1 µm vermessen werden. Als Messempfänger wird ein Vektor Signalanalysator (VSA) mit zwei Messkanälen eingesetzt. Der Frequenzbereich reicht bis zu 6 GHz und in der neuesten Generation bis zu 90 GHz. Die mit dem NFS3000 verwendeten Sonden sind selbst entwickelt und somit auf das Gesamtsystem abgestimmt. Zur präzisen und automatisierten Positionierung der Nahfeldsonden wird die 3D-Kontur des DUTs optisch erfasst.

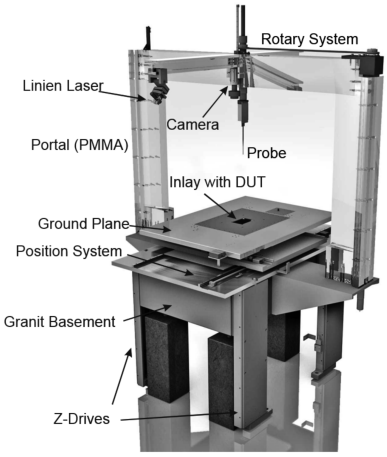



Abb. 3: Abbildung des NFS3000

### 2.3 Oberflächen-Aquivalenz-Theorem

Das Oberflächen-Aquivalenz-Theorem, auch HUYGENS-Prinzip, wird kurz theoretisch vorgestellt, da es die Grundlage für den späteren Einsatz der Nahfelddaten als Simulationsquelle ist [La20]. Es besagt, dass jeder Punkt einer sich ausbreitenden Wellenfront selbst eine Quelle von Strahlungswellen ist. Nach diesem Theorem kann eine reale, elektrische Strahlungsquelle, im besonderen ein elektrisches DUT, durch eine fiktive Menge äquivalenter Quellen ersetzt werden. Diese befinden sich auf einer beliebigen, geschlossenen Oberfläche, die das DUT umhüllt [Gi07].
Der das DUT umgebende Raum wird mit $D_0$ bezeichnet. Es wird vorausgesetzt, dass er mit einem homogenen, linearen und isotropen Material gefüllt ist. $D_0$ wird dabei durch die geschlossene Oberfläche $S$ in zwei Teilbereiche unterteilt. Die äquivalenten Quellen auf $S$ sind durch die die beiden fiktiven Stromdichten $\vec{M}$ (magnetisch) und $\vec{K}$ (elektrisch) gegeben. Diese lassen sich aus dem tangentialen elektrischen Feld

$$\vec{M} = -\vec{n} \times \vec{E}|_{\vec{r} \in S} \tag{4}$$

und dem tangentialen magnetischen Feld

$$\vec{K} = \vec{n} \times \vec{H}|_{\vec{r} \in S} \tag{5}$$

auf der Fläche $S$ bestimmen. Die Oberflächenstromdichten stellen die Felder in den beiden Unterräumen dar. Somit können Ersatzgleichungen für den inneren und äußeren Unterraum angegeben werden. Das führt zu stückweise definierten Gleichungen für die Felder in $D_\pm$, den so genannten STRATTON-CHU-Gleichungen [St07]. Die Original- und Ersatzanordnung ist in Abb. 4 dargestellt. Die Theorie zum HUYGENS-Prinzip wird in [La20] ausführlich zusammengefasst.

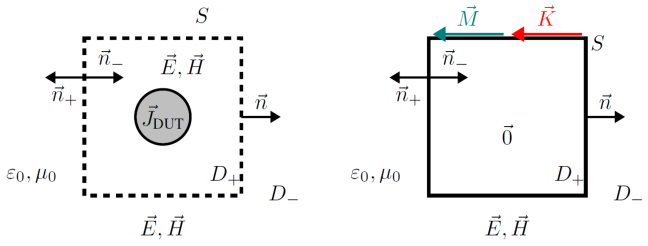



Abb. 4: HUYGENS-Box Prinzip und Ersatzanordnung mit Oberflächenstromdichten

## 3 Nahfeldmessungen von IoT-Komponenten

Im folgenden Abschnitt werden anhand von zwei Beispielen die Ergebnisse von Nahfeldmessungen dargestellt. Durchgeführt werden diese mit dem beschriebenen NFS3000. Zum

einen wird eine für IoT-Anwendungen vorgesehene Antenne bei 2,4 GHz vermessen und mit Feldsimulationen verglichen, um die Messergebnisse zu verifizieren. Zum anderen wird ein ESP32 basiertes Modul im Bluetooth-Frequenzbereich untersucht. Die begleitenden Feldsimulationen werden mit CST Studio Suite durchgeführt.

### 3.1 IoT-Antenne bei 2,4 GHz

Das CC-Antenna-DK2 Kit der Firma Texas Instruments beinhaltet verschiedenste Antennen für Frequenzbereiche bis 2,4 GHz, welche für einfache Tests und Untersuchungen geeignet sind. Da die Gerber-Dateien und Materialeigenschaften frei verfügbar und bekannt sind, eignet sich das Kit ebenfalls für Nahfeldmessungen und für Vergleiche mit Feldsimulationen. Charakterisiert wird die Antenne 11 aus dem Kit, dessen Form einer *Inverted F Antenne*

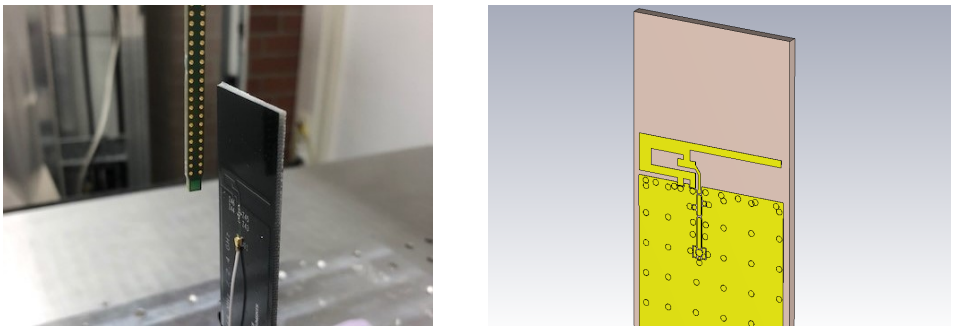


Abb. 5: Antenne 11 aus dem CC-Antenne-DK2 Kit von Texas Instruments

mit einer Resonanzfrequenz um 2,4 GHz entspricht, siehe Abb. 5. Die Antenne wird mit einem Signal-Generator betrieben. Vorteilhaft ist hierbei, dass das Sinus-Signal über einen Leistungsteiler gleichzeitig zum DUT und zum Referenz-Kanal geführt werden kann und somit eine einfache Messung der Phase möglich ist. In Abb. 6 wird beispielhaft das Ergebnis

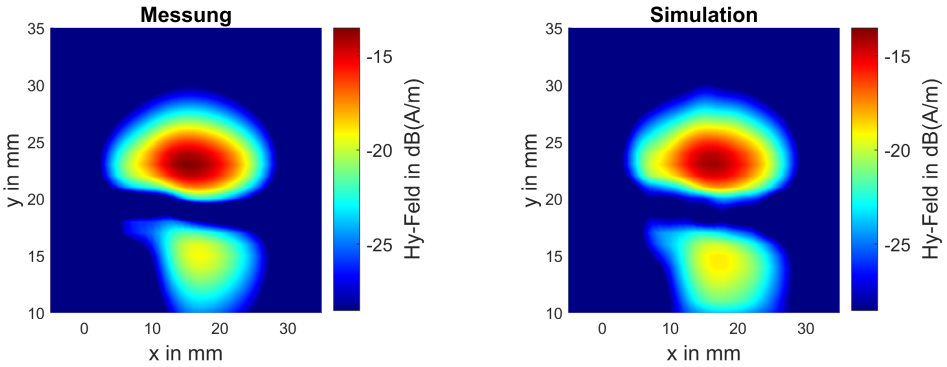



Abb. 6: Amplitude des $H_y$-Feldes der Antenne von Messung (links) und Simulation (rechts)

der Messung einer tangentialen H-Feld Komponente dargestellt. Die $H_y$-Komponente in diesem Fall ist in einem vertikalen z-Abstand von 2 mm gemessen worden. Bei rein optischer Betrachtung fallen keine großen qualitativen oder quantitativen Unterschiede auf. Der RMSE über alle Messpunkte liegt bei etwa 4 Prozent. Die weiteren Abweichungen der anderen tangentialen Feldkomponenten sind in Tab. 1 aufgelistet. Somit kann festgestellt werden, dass es eine gute Übereinstimmung zwischen Simulation und Messung für diese Antenne gibt. Neben der Amplitude kann mit einem zwei-kanaligen Nahfeldscannern auch die

| | RMSE | | RMSE |
|---|---|---|---|
| $H_x$ | 5,5 % | $E_x$ | 16 % |
| $H_y$ | 4 % | $E_y$ | 9,5 % |

Tab. 1: Abweichung zwischen Simulation und Messung der Antenne

Phase bestimmt werden. In Abb. 7 sind Simulation und Messung der Phase des $H_y$-Feldes dargestellt. Auch hier kann eine gute Übereinstimmung festgestellt werden. Lediglich dort, wo die Amplitude geringer ist, ist das Rauschen der Phase etwas größer. Im Mittel beträgt die Abweichung zwischen beiden Bildern rund 8°.

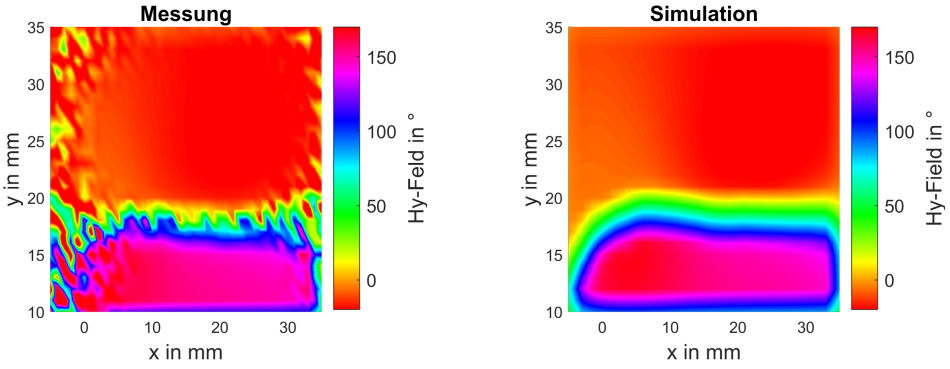



Abb. 7: Phase des $H_y$-Feldes der Antenne von Messung (links) und Simulation (rechts)

### 3.2 Messung eines ESP32-IoT-Moduls

Häufig für IoT-Anwendungen eingesetzte Module bauen auf ESP32 Mikrocontrollern auf. Im folgenden Beispiel soll ein ESP32-Entwicklungsboard im Bluetooth Low Energy (BLE) *Advertising* Modus gemessen werden. Die Antenne, die hier verwendet wird, ist eine *Meander-Line Inverted F* Antenne (MIFA), welche für den 2,4 GHz Bereich ausgelegt ist. Die BLE-Sendeleistung des ESP32 Moduls liegt bei rund 9 dBm.
Im letzten Abschnitt ist die Antenne per Signal-Generator gespeist worden, was einen direkten Zugriff auf die Phaseinformation zugelassen hat und ebenfalls für ein periodisches Signal gesorgt hat. In praktisch relevanten Komponenten ist das nicht der Fall. Die Signale und Abläufe werden durch die Komponenten selbst erzeugt und meistens gibt es keine

frei zugänglichen Testpunkte. Daher wird für solche Testobjekte die Referenz aus der abgestrahlten Leistung mit einer ortsfesten Referenzsonde gewonnen.
Der zweite entscheidende Unterschied ist die Dauer der Periodizität. Mit einem Signalgenerator wird ein kontinuierlicher Sinus erzeugt, der sich im Takt der Frequenz wiederholt. Bei realen Anwendungen ist das in der Regel nicht gegeben. So können sich Signale erst sehr langsam wiederholen oder im schlimmsten Fall sogar willkürlich sein. Für Nahfeldmessungen ist es jedoch wichtig, dass sich ein periodischer Verlauf einstellt, da an jedem räumlichen Messpunkt die gleichen elektrischen Bedingungen vorherrschen sollten.
Das ESP32-Modul gewährleistet die Periodizität über das sich kontinuierlich wiederholende *Advertisen* des BLE-Protokolls. Das Modul sendet dabei alle rund 110 ms für ungefähr 2,5 ms ein BLE-Signal, um für andere BLE-Geräte sichtbar zu sein. Um die Qualität und Zuverlässigkeit der Messung zu erhöhen sollten mehrere Perioden des sich wiederholenden Signals gemessen werden. Dazu gibt es die Möglichkeit entweder eine sehr lange Messung durchzuführen oder über mehrere, kürzere Zeiträume zu mitteln. In Abb. 8 wird das Spektrum des BLE-Signals für 1,5 s ohne Mittelung und für 0,3 s mit 5-facher Mittelung (effektiv wieder 1,5 s) verglichen. Der entscheidende Vorteil bei der zweiten Variante ist das geringere Rauschen und wird daher bei der Messung angewendet. In Abb. 9 ist der Messaufbau und

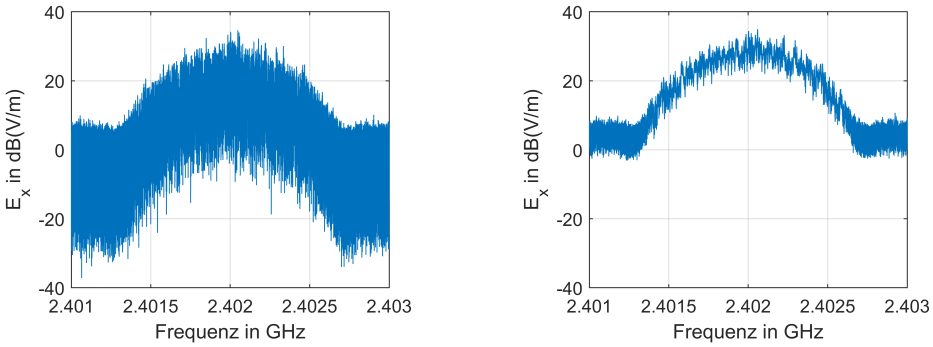



Abb. 8: ESP32 BLE-Spektrum ein einem Messpunkt ohne (links) und mit (rechts) Mittelung

das Ergebnis für das $H_x$-Feld direkt über der Antenne bei der Mittenfrequenz von 2,402 GHz des ersten BLE-*Advertising* Kanals dargestellt. Zu erkennen ist, dass die Messergebnisse zwar etwas verrauschter sind, als bei der mit dem Sinus gespeisten Antenne. Das Feld kann dennoch sehr gut gedeutet und erkannt werden.

# 4 Äquivalente Nahfeldquelle in Feld-Simulationen

## 4.1 Fernfeld-Extrapolation

Bisher wurden 2D-Bilder der Feldverteilungen einer Fläche über dem DUT dargestellt. Zur einfachen Charakterisierung und Fehlerbehebung ist das ausreichend. Wie bereits

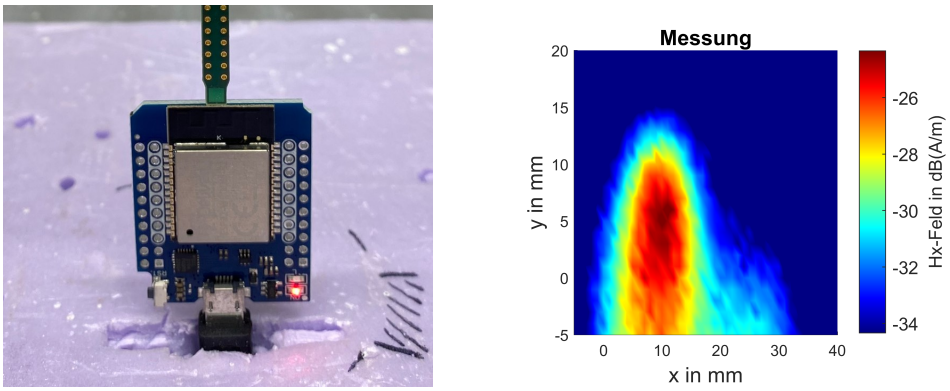



Abb. 9: Messaufbau mit dem ESP32-Modul und Ergebnis des $H - x$-Feldes

beschrieben, kann mit der Messung der tangentialen Feldkomponenten die HUYGENS-Box um das DUT herum erfasst werden.
Im einfachsten Fall kann die HUYGENS-Box dazu verwendet werden die Nahfelddaten ins Fernfeld zu extrapolieren. Dazu werden die Daten in ein Feldsimulationsprogramm wie CST reingeladen und mittels Fernfeldmonitor können die Fernfelddaten berechnet werden. Für EMV-Untersuchungen können so z.B. die EMV-Konformitätsmessungen in der Fernfeldkammer abgeschätzt werden. Bei Antennen bzw. Kommunikationsmodulen kann damit vor allem die Richtcharakteristik des Gesamtsystems bestimmt werden.
Zur Verifikation der HUYGENS-Box Methode wird die Antenne 11 aus dem Antennen-Kit verwendet. In Abb. 10 wird die Fernfeld-Simulation aus dem Simulationsmodell mit der Fernfeld-Extrapolation aus der Nahfeldmessung verglichen. Einige Unterschiede lassen sich erkennen. Vor allem die Hauptabstrahlrichtung und die grobe Form passen jedoch gut zusammen. Daher kann aus den Nahfelddaten eine gute, erste Schätzung des Fernfeldes hergeleitet werden.

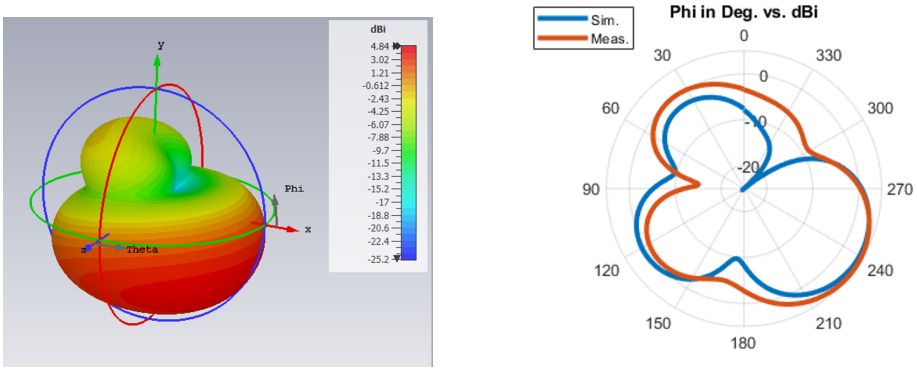



Abb. 10: Richtcharakteristik der Fernfeld-Extrapolation und Vergleich mit Simulation

### 4.2 Bewertung von Überkoppeln

Interessanter wird die Anwendung der HUYGENS-Box, wenn weitere Gegenstände in die Simulationsumgebung mit eingebracht werden. So kann z.B. das Überkoppeln von elektrischen Komponenten auf andere Gegenstände bewertet werden. In der nächsten Untersuchung wird die Nahfeldquelle des ESP32-Moduls erfasst. Diese wird dann neben einem beispielhaften Modell mit MIFA-Antenne, einigen Bauteile und Leitungen platziert. Der Aufbau ist in Abb. 11 dargestellt. An den Leitungen zum exemplarischen Modell eines ICs in der Mitte werden jeweils 50 Ω Widerstände vorgesehen, an denen die Spannung beobachtet werden kann. Einmal zur Antenne, einmal zum unteren Leitungspaar, welches ebenfalls mit 50 Ω abgeschlossen ist.
Das Ergebnis ist rechts in Abb. 11 zu sehen. Die maximale Feldstärke auf dem Modell liegt bei rund 200 $\mathrm{V\,m^{-1}}$. Die Spannungen an den Widerständen liegen bei 60 mV für die Antenne und bei 10 mV für das Leitungspaar. Damit kann veranschaulicht werden, welche Möglichkeiten sich durch die Kombination von Messung und Simulation bieten. Zum einen kann analysiert werden, wie das gemessene DUT sich elektromagnetisch verhält. Zum anderen können andere Objekte auf ihre Störanfälligkeit hin bewertet und optimiert werden.

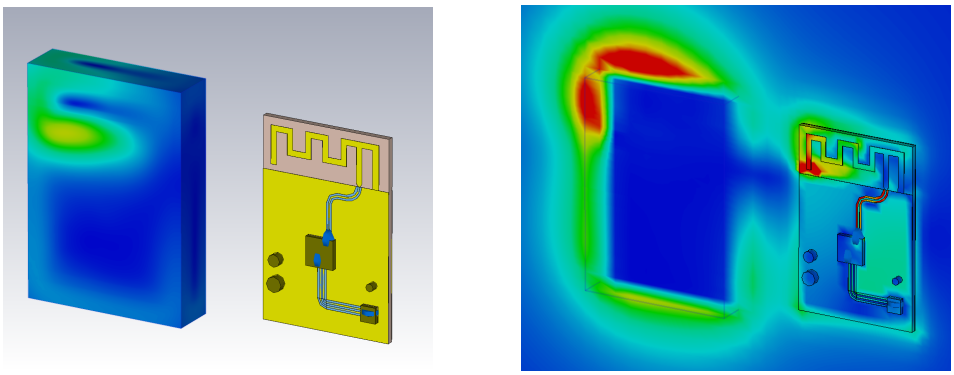


Abb. 11: Simulation des Einflusses des ESP32-Moduls auf eine andere Komponente

### 4.3 Abstrahlung in den Raum

Neben der Nahfeldinteraktion mit Gegenständen in unmittelbarer Nähe kann auch die Abstrahlung in den Raum von Interesse sein. So könnte zum Beispiel mittels Simulationen die Positionierung und Ausrichtung von Kommunikationsmodulen in Anlagen, Räumen und Hallen optimiert werden.
Zur Veranschaulichung wird die HUYGENS-Box des ESP32-Moduls in einen 10 m × 7,5 m großen Raum importiert, in dem zwei stark vereinfachte Anlagen stehen. Das Modul wird auf der einen Anlage platziert und die Abstrahlung in den Raum charakterisiert und visualisiert. In Abb. 12 ist das elektrische Feld in der z-Ebene dargestellt. Somit kann eine einfache und

schnelle, erste Bewertung durchgeführt werden. Mit einem Optimierungsverfahren kann die Position noch verbessert werden.

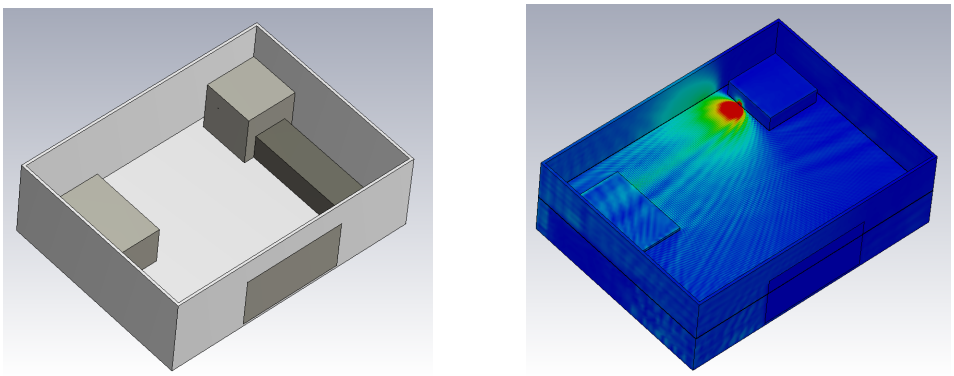


Abb. 12: Simulation der Abstrahlung des ESP32-Moduls in den Raum

### 4.4 Weitere Anwendungsgebiete

Die möglichen Anwendungsgebiete der HUYGENS-Box Methode lässt sich beliebig erweitern. So könnten zum Beispiel EMV- und Abstrahluntersuchungen in Autos oder Flugzeugen durchgeführt werden. Entweder um Komponenten optimiert zu platzieren oder Störungen im Gesamtsystem zu vermeiden. Auch die Auswirkung von Gehäusen auf die elektrischen Eigenschaften eines Systems, zum Beispiel die Änderung der Richtcharakteristik, können untersucht werden.
Ein weiteres, spannendes Gebiet behandelt den Einfluss elektrischer Komponenten auf den menschlichen Körper. Die HUYGENS-Box kann im Simulationsprogramm in der Nähe oder auf menschlichen Modellen positioniert werden und der Einfluss bzw. die elektromagnetische Strahlung in den Körper kann untersucht werden. Die sogenannten SAR-Werte zur Bewertung der Erwärmung des Körpers können somit simuliert werden.

## 5 Fazit

In diesem Beitrag wurde die HUYGENS-Box Methode vorgestellt und anhand von IoT-Komponenten beispielhaft demonstriert und bewertet. Dabei wird aus Nahfeldmessungen eine Nahfeldquelle erzeugt, die in ein Feldsimulationsprogramm importiert wird. Ausgehend von den notwendigen Grundlagen des Messsystems und der theoretischen Betrachtungen wurden Nahfeldmessergebnisse einer Antenne und eines ESP32-Moduls im 2,4 GHz Band dargestellt. Für die Antenne wurden die Messergebnisse anhand von Simulationen verifiziert und bewertet, wobei die Werte für Betrag und Phase sehr gut übereinstimmen.
Von beiden Komponenten wurden anschließend die HUYGENS-Box Daten erfasst. Die

Methode zur Kombination von Nahfeldmessung und Simulation wurde anhand der Fernfeld-Extrapolation, dem Überkoppeln auf andere Komponenten und der Abstrahlung in einen Raum mit Anlagen gezeigt. Die Fernfeld-Extrapolation der Antenne wurde durch eine Feldsimulation verifiziert, wobei eine gute, erste Abschätzung des Fernfeldes aus den Nahfelddaten erfolgen kann. Die anderen beiden Untersuchungen haben gezeigt, dass durch das beschriebene Vorgehen die Interaktion des Testobjektes mit der Umgebung dargestellt und analysiert werden kann. Abschließend wurden einige weitere Anwendungsgebiete der HUYGENS-Box Methode skizziert.
In zukünftigen Arbeiten müssen vor allem die letzten beiden Beispiele durch Messungen und reale Experimente verifiziert werden.

## Literaturverzeichnis


[Cl18] Claeys, Tim: Increasing the Accuracy and Speed of EMI Near-Field Scanning. Dissertation, KU Leuven, 2018.

[Gi07] Gibson, Walton C.: The Method of Moments in Electromagnetics. Chapman & Hall/CRC, Boca Raton, FL, 2007.

[Ha16] Hangmann, C.; Mager, T.; Khan, S.; Hedayat, C.; Hilleringmann, U.: Improved RF Design Using Precise 3D Near-Field Measurements and Near-Field to Far-Field Transformations. In: Smart System Integration - International Conference and Exhibition on Integration Issues of Miniaturized Systems. 2016.

[La20] Lange, Sven; Schröder, Dominik; Hedayat, Christian; Hangmann, Christian; Otto, Thomas; Hilleringmann, Ulrich: Investigation of the Surface Equivalence Principle on a Metal Surface for a Near-Field to Far-Field Transformation by the NFS3000. In: 2020 International Symposium on Electromagnetic Compatibility - EMC EUROPE. S. 1–6, 2020.

[NFG12] Noren, P.; Foged, L. J.; Garreau, P.: State of the art spherical near-field antenna test systems for full vehicle testing. In: 2012 6th European Conference on Antennas and Propagation (EUCAP). S. 2244–2248, March 2012.

[QSE13] Qureshi, M. A.; Schmidt, C. H.; Eibert, T. F.: Adaptive Sampling in Spherical and Cylindrical Near-Field Antenna Measurements. IEEE Antennas and Propagation Magazine, 55(1):243–249, Feb 2013.

[RMH09] Reinhold, C.; Mager, T.; Hedayat, C.: Three dimensional near field scanning measurement techniques for improved RF design. In: SAME 2009, 12th Edition. Sophia Antipolis, France, 2009.

[SJ17] Selvan, K. T.; Janaswamy, R.: Fraunhofer and Fresnel Distances : Unified derivation for aperture antennas. IEEE Antennas and Propagation Magazine, 59(4):12–15, Aug 2017.

[St07] Stratton, J.A.: Electromagnetic Theory. IEEE Press Series on Electromagnetic Wave Theory. Wiley, 2007.

[Ta07] Tankielun, A.: Data Post-Processing and Hardware Architecture of Electromagnetic Near-Field Scanner. Dissertation, Gottfried Wilhelm Leibniz Universität Hannover, 2007.

[Ya86] Yaghjian, A.: An overview of near-field antenna measurements. IEEE Transactions on Antennas and Propagation, 34(1):30–45, January 1986.

# Security-Tests für Industriekomponenten nach IEC 62443 – Fuzzing-Testing von Kommunikationsschnittstellen

Jens Otto[1], Felix Specht[1] und Boris Waldeck[2]

[1]Fraunhofer IOSB-INA, 32657 Lemgo, {jens.otto; felix.specht}@iosb-ina.fraunhofer.de
[2]Phoenix Contact Electronics GmbH, 31812 Bad Pyrmont, bwaldeck@phoenixcontact.com

**Abstract:** Security ist eine Voraussetzung für moderne Industriekomponenten, welche zunehmend nach den Vorgaben der internationalen Normenreihe IEC 62443 entwickelt und zertifiziert werden. Diese Normenreihe definiert Anforderungen für das kontinuierliche Testen aller Kommunikationsschnittstellen einer Industriekomponente, wozu insbesondere das Fuzzing-Testing gehört. Eine Fuzzing-Bibliothek wurde vom Fraunhofer IOSB-INA entwickelt, welche für die Zertifizierung von Komponenten nach IEC 62443 eingesetzt werden kann. Die Lösung verfügt über die Möglichkeit die für das Fuzzing-Testing notwendige Datenstrukturen aus aufgezeichnetem Netzwerkverkehr zu extrahieren, wodurch sich die Lösung einfach an Protokolle oder spezifische Komponenten anpassen lässt.

**Keywords:** Security-Tests, Fuzzing-Testing, IEC 62443

## 1 Einleitung

Security ist eine Voraussetzung für moderne Industriekomponenten [Ni14]. Industriekomponenten werden daher zunehmend nach den Vorgaben der internationalen Normenreihe IEC 62443 entwickelt und zertifiziert [Bsi13]. Die Normenreihe IEC 62443 definiert Anforderungen für das kontinuierliche Testen aller Kommunikationsschnittstellen einer Industriekomponente. Insbesondere das Fuzzing-Testing ist dabei vorgeschrieben [Bsi14]. Fuzzing-Testing ist eine Technik zur automatisierten Fehlerfindung, die z.B. Kommunikationsschnittstellen systematisch mit generierten Eingaben überprüft. Dadurch können Komponentenhersteller potenzielle Sicherheitslücken aufdecken und Sicherheitsupdates zeitnah bereitstellen.

Die in dieser Veröffentlichung vorgestellte Softwarebibliothek implementiert unterschiedliche Algorithmen für das Fuzzing-Testing. Die folgenden Funktionalitäten werden unter-

[1] Fraunhofer IOSB-INA, Campusallee 1, 32657 Lemgo, jens.otto@iosb-ina.fraunhofer.de, felix.specht@iosb-ina.fraunhofer.de
[2] Phoenix Contact GmbH, Dringenauer Straße 30, 31812 Bad Pyrmont, bwaldeck@phoenixcontact.com

stützt: (1) Generieren und Mutieren von Testeingaben, (2) Extrahieren von Datenstrukturen aus aufgezeichnetem Netzwerkverkehr, (3) Datenbank mit Testeingaben, (4) Versenden von Testeingaben über unterschiedliche Netzwerkprotokolle, (5) Überwachung der zu testenden Industriekomponente und (6) automatisiertes Erzeugen von Testberichten. Unter Verwendung der vorgestellten Softwarebibliothek können Kommunikationsschnittstellen von industriellen Steuerungen kontinuierlich auf Fehler und Sicherheitslücken getestet werden. Eine Zertifizierung der Produktentwicklung nach IEC 62443-4-1 und des Produktes nach IEC 62443-4-2 werden durch die Softwarebibliothek unterstützt. Des Weiteren ist geplant, die Softwarebibliothek als Open Source bereitzustellen, sodass andere Unternehmen an den Ergebnissen partizipieren können.

Abbildung 1 zeigt die Beiträge der Veröffentlichung: (1) Vorstellung einer Softwarebibliothek für das Fuzzing-Testing von Softwarediensten für Industriekomponenten, (2) Konzept für die automatisierte Erstellung von Testfällen und (3) Konzept für die automatisierte Durchführung von Testfällen. Die Softwarebibliothek für das Fuzzing-Testing kapselt alle Funktionalitäten, um einen Testingenieur zu befähigen, Testfälle mit Fuzzing-Algorithmen zu erstellen. In der Industrie werden Testfälle häufig in der Programmiersprache Python geschrieben, daher sind die Fuzzing-Algorithmen und notwendigen Funktionalitäten in Python implementiert. Die erstellten Testfälle können verwendet werden, um Softwaredienste (Webserver, OPC UA-Server, Profinet-Stack, etc.) einer Industriekomponente, z.B. einer Industriesteuerung, zu überprüfen. Die manuelle Erstellung und die manuelle Durchführung von Testfällen sind zeitaufwändig. Um die Aufwände zu reduzieren, werden Konzepte zur automatisierten Erstellung von Testfällen und die automatisierte Durchführung der erstellten Testfälle vorgestellt.

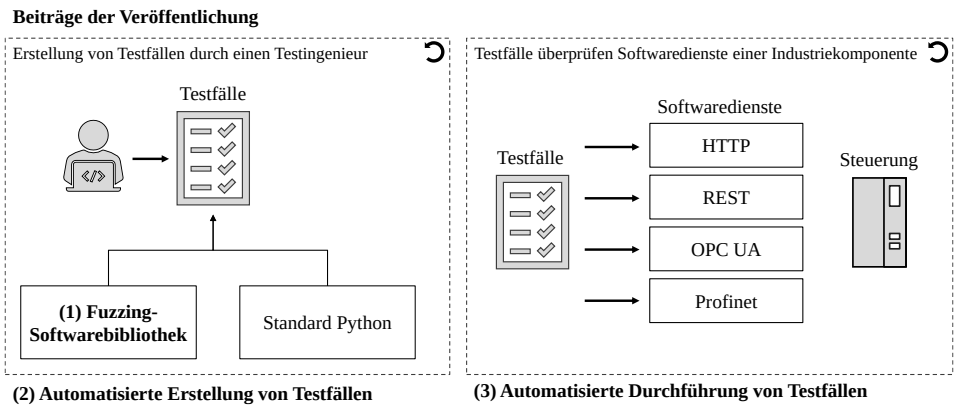



**Abbildung 1:** Überblick über die Beiträge der Veröffentlichung.

Die Veröffentlichung ist wie folgt aufgebaut. Kapitel 2 beschreibt den Stand der Technik von Fuzzing-Testing mit besonderem Fokus auf industrielle Kommunikationssysteme. Im Hauptkapitel 3 wird der Lösungsansatz im Detail vorgestellt. Kapitel 4 gibt exemplarische Ergebnisse wieder und Kapitel 5 fasst die Veröffentlichung zusammen.

## 2 Stand der Technik

Das Ziel des Fuzzing-Testing ist das Aufdecken von Fehlern in Softwarediensten, z.B. Webserver, OPC UA-Server, REST-Schnittstellen oder Profinet-Stacks. Techniken des Fuzzing-Testing werden in der Literatur auf unterschiedliche Arten klassifiziert [Ma19, Su07]. Es wird zwischen Black-, White-, oder Grey-Box-Ansätzen unterschieden.

Black-Box-Ansätze testen auf Basis der externen Kommunikationsschnittstellen und ohne Expertenwissen. Viele etablierte Softwarewerkzeuge erzeugen Testeingaben nach dem Zufallsprinzip, z.B. LibFuzzer[3], CERT Basic Fuzzing Framework[4] oder SPIKE[5]. Modernere Black-Box-Ansätze, wie Peach[6] oder funfuzz[7], berücksichtigen zusätzlich strukturelle Informationen, um Eingaben zu erzeugen. [Ch10] beschreiben einen Black-Box-Ansatz mit „Adaptive Random Testing“, eine Erweiterung des zufallsbasierten Testens. Dabei wird eine zufällig generierte Eingabe systematisch modifiziert und erneut eingespielt.

White-Box-Ansätze testen Dienste auf Basis des Quellcodes und mit Expertenwissen über den Programmablauf. Der White-Box-Ansatz von [Go08] erzeugt Testfälle basierend auf einer Analyse mit symbolischer Programmausführung. [Ga09] beschreiben einen White-Box-Ansatz mittels “Taint-Tracing”. Dabei werden potenzielle Angriffspunkte identifiziert, indem die Wechselwirkung zwischen Eingabebytes und den beeinflussten Werten im Programm analysiert wird.

Grey-Box-Ansätze testen ebenfalls auf Basis der Kommunikationsschnittstellen, die Tests werden jedoch mit Expertenwissen angereichert. Erste Grey-Box-Ansätze, z.B. Sidewinder[8], basieren auf der Erfassung und Analyse der „Code Coverage“-Metrik bei der Testausführung. Die Analyseergebnisse dienen als Eingabe für einen evolutionären Algorithmus, der neue Tests generiert [De07]. AFL[9] und VUzzer sind Beispiele für moderne Grey-Box-Ansätze, die evolutionäre Algorithmen mit anwendungsspezifischem Wissen kombinieren. Dabei werden Kontroll- und Datenflussmerkmale durch statische und dynamische Softwareanalyse erhoben [Ra17].

Unterschieden wird außerdem zwischen Ansätzen, die Testeingaben unstrukturiert oder strukturiert generieren. Unstrukturierte Ansätze setzen kein Expertenwissen über die zu

[3] „LibFuzzer“, 2015. [Online] (https://llvm.org/docs/LibFuzzer.html Stand 19.08.2022)
[4] CERT, „Basic fuzzing framework“, 2010. [Online] (https://www.cert.org/vulnerability-analysis/tools/bff.cfm Stand 19.08.2022)
[5] D. Aitel, “An introduction to SPIKE, the fuzzer creation kit,” in Proc. Black Hat USA, 2002.
[6] M. Eddington, “Peach fuzzing platform”, 2014. [Online] (https://peachtech.gitlab.io/peach-fuzzer-community Stand 19.08.2022)
[7] M. Security, “funfuzz”, 2016. [Online] (https://github. com/MozillaSecurity/funfuzz Stand 19.08.2022)
[8] S. Embleton, S. Sparks, and R. Cunningham, ““sidewinder”: An evolutionary guidance system for malicious input crafting,” in Proc. Black Hat USA, 2006.
[9] M. Zalewski, “American Fuzzy Lop,” 2014. [Online] (http://lcamtuf.coredump.cx/afl/ Stand 19.08.2022)

testenden Schnittstellen voraus, können jedoch nicht gezielt testen. Strukturierte Ansätze benötigen zunächst Informationen über die erwarteten Datenstrukturen einer Schnittstelle und können dann gezielt nach Fehlern suchen.

Es werden generierende und mutierende Ansätze unterschieden, je nachdem, ob die Eingaben von Grund auf neu generiert oder bestehende Eingaben verändert werden [Su07]. Die Softwarewerkzeuge Peach, PROTOS und Sulley sind Beispiele für generierende Ansätze [Ka01, Am12]. AFL, SymFuzz und honggfuzz[10] fallen hingegen in die Kategorie der mutierenden Ansätze [Ch15].

Ansätze können in protokoll- oder anwendungsspezifisch unterteilt werden. Protokollspezifische Ansätze testen die Implementierung eines Kommunikationsprotokolls, z.B. Profinet. Anwendungsspezifische Ansätze testen hingegen die Anwendung, die auf einem Kommunikationsprotokoll aufsetzt, z.B. eine IEC 61131-3 Laufzeitumgebung. Fuzzing-Testing ist in der klassischen IT-Security ein Thema mit einer Vielzahl von Veröffentlichungen in den o.g. Kategorien. Für IT-Systeme sind entsprechende Open Source und proprietäre Lösungen verfügbar.

Im Bereich der OT-Security existieren hingegen nur wenige Ansätze, welche insbesondere auf das Fuzzing-Testen von proprietären Industrieprotokollen abzielen. PropFuzz verwendet statistische Analyse zur Identifikation von testrelevanten Bytes in Netzwerkpaketen [Ni17]. Ein weiterer Ansatz ist eine Erweiterung für die Fuzzing-Software Peach um die Industrieprotokolle Modbus und DNP3 [Zh20]. Der Ansatz ICPFuzzer verwendet maschinelle Lernverfahren, um zunächst Muster in proprietären Industrieprotokollen zu identifizieren und anschließend zu testen [Pe21]. ISuTest ist ein Ansatz für ein modulares Test-Framework für Industriekomponenten, welches diverse Module zum Security-Testing verwendet [Pf17, Pf19, Pfb19]. ISuTest inkludiert ebenfalls bestehende Fuzzing-Ansätze als Module [Pf18].

Basierend auf den Vorarbeiten von [Pf18], gibt es folgende Anforderungen für eine Fuzzer-Implementierung im Bereich der Automatisierungstechnik, siehe Tabelle 1.

**Tabelle 1**: Anforderungen an eine Fuzzer-Implementierung nach [Pf18].

| ID | Beschreibung: |
|---|---|
| FR1 | Die Integration von Expertenwissen soll ermöglicht werden. |
| FR2 | Eingaben vor und während der Laufzeit sollen generiert werden können. |
| FR3 | Zustandsbehaftete Industrieprotokolle sollen unterstützt werden. |
| FR4 | Sequenzen von Datenpaketen sollen variiert werden können. |

[10] R. Swiecki and F. Gröbert, “honggfuzz,” 2010. [Online] (https://github.com/google/honggfuzz Stand 19.08.2022)

| | |
|---|---|
| FR5 | Die Antworten des Testgeräts sollen überwacht werden. |
| FR6 | Alle verfügbaren Netzwerkschnittstellen sollen geprüft werden. |
| FR7 | Neue Netzwerkprotokolle sollen integrierbar sein. |
| FR8 | Die Ethernet-Schicht (ISO/OSI Layer 2) muss berücksichtigt werden. |
| FR9 | Generierte Testfälle müssen reproduzierbar sein. |

# 3 Lösungsansatz

In diesem Kapitel werden die folgenden Lösungskonzepte vorgestellt: (1) Eine Softwarebibliothek für das Fuzzing-Testing von Softwarediensten für Industriekomponenten, (2) ein Konzept für die automatisierte Erstellung von Testfällen und (3) ein Konzept für die automatisierte Durchführung von Testfällen. Auf die beschriebenen Anforderungen aus dem Stand der Technik wird verwiesen, siehe Tabelle 1.

## 3.1 Vorstellung der Softwarebibliothek für das Fuzzing-Testing

Das Ziel der Softwarebibliothek für das Fuzzing-Testing ist die Kapselung aller notwendigen Funktionalitäten und Algorithmen, um einen Testingenieur zu befähigen, Testfälle mit Fuzzing-Algorithmen zu erstellen. Testfälle werden häufig in der Programmiersprache Python geschrieben, daher sind die Fuzzing-Algorithmen und notwendigen Funktionalitäten in Python implementiert. Die erstellten Testfälle können dann verwendet werden, um Softwaredienste, z.B. Webserver, OPC UA-Server, Profinet, etc., einer Industriekomponente zu überprüfen.

Abbildung 2 zeigt die Lösungselemente und den Aufbau der Softwarebibliothek für das Fuzzing-Testing. Die Softwarebibliothek besteht aus sieben Modulen: (1) Fuzzing, (2) Decoder, (3) Encoder, (4) Datenbank, (5) Reporting, (6) maschinelles Lernen und (7) einem Codegenerator.

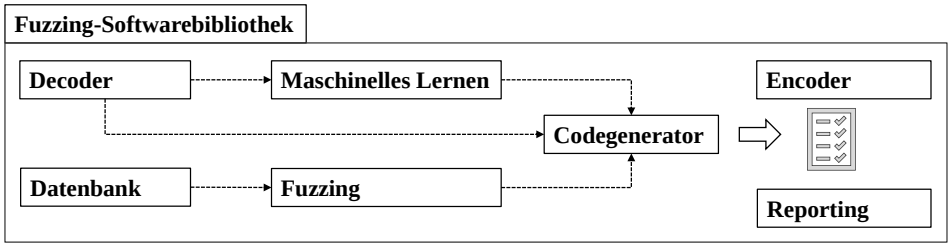



**Abbildung 2:** Aufbau der Fuzzing-Softwarebibliothek.

Das Fuzzing-Modul implementiert unterschiedliche Algorithmen für die Erzeugung von Testeingaben. Die Testeingaben werden verwendet, um unterschiedliche Industrieprotokolle auf eine fehlerhafte Verarbeitung von Eingabedaten zu überprüfen. Folgende Algorithmen sind implementiert: (1) Random-Generator: Generierung zufälliger Zeichenketten, (2) Mutation-Generator: Mutation bestehender Zeichenketten, (3) Grammar-Generator: Generierung von Zeichenketten mit Grammatiken und (4) Wordlist-Generator: Rückgabe von Zeichenketten aus Wortlisten. Das Fuzzing-Modul erfüllt damit die Anforderung FR2. Das Datenbank-Modul speichert die notwendigen Eingabedaten des Fuzzing-Moduls. Eingabedaten können z.B. Grammatiken und Wortlisten für die Generatoren sein, dadurch kann die Anforderung FR1 erfüllt werden.

Das Decoder-Modul extrahiert strukturelle Informationen aus aufgezeichneten Netzwerkdaten, z.B. HTTP GET/POST + URL. Der Decoder unterstützt HTTP, OPC UA und Profinet und kann um weitere IT- und Industrieprotokolle erweitert werden. Die strukturellen Informationen werden verwendet, um (1) den Suchraum zu reduzieren und (2) eine automatisierte Erstellung von Testfällen zu ermöglichen. Theoretisch können unendliche Testeingaben durch die Generatoren erstellt werden, daher ist es notwendig, die Suche nach Fehlern einzuschränken. Das Encoder-Modul implementiert die IT- und Industriekommunikationsprotokolle HTTP, OPC UA und Profinet. Das Modul kann um weitere Protokolle erweitert werden. Die generierten Testeingaben des Fuzzing-Moduls werden durch das Encoder-Modul an das Test- bzw. Industriegerät gesendet. Dadurch werden die Anforderungen FR2, FR6, FR7 und FR8 erfüllt.

Das Maschinelle-Lernen-Modul generiert ein Protokoll aus Ereignisdaten. Dazu werden strukturelle Informationen aus dem Decoder-Modul verwendet. Aus dem Protokoll der Ereignisdaten wird dann ein Verhaltensmodell mit z.B. Process-Mining-Algorithmen gelernt. Process Mining wird in unterschiedlichen Bereichen der Industrie verwendet, um aus Ereignisdaten Verhaltensmodelle zu lernen [Ot15, Ot18, OtV18]. Das Verhaltensmodell wird verwendet, um die Reihenfolge von zustandsbehafteten Kommunikationsprotokollen zu variieren. Dadurch werden die Anforderungen FR3 und FR4 erfüllt.

Das Codegenerator-Modul generiert Testfälle. Dazu werden die Informationen der Verhaltensmodelle und der strukturellen Informationen aus aufgezeichneten Netzwerkdaten verwendet. Dadurch wird die Anforderung FR9 erfüllt. Das Reporting-Modul überwacht das Testgerät. Dazu werden Daten, z.B. CPU- und Speicherauslastung, aufgezeichnet und ausgewertet. Ein Report kann generiert werden, um die Testdurchläufe zu dokumentieren. Dadurch wird Anforderung FR5 erfüllt.

Die Implementierung erfüllt alle Anforderung FR1 bis FR9 an eine Fuzzer-Implementierung nach [Pf18], siehe Tabelle 1.

## 3.2 Konzept für die automatisierte Erstellung von Testfällen

Die manuelle Erstellung von Testfällen durch einen Testingenieur ist zeitaufwändig. Um diese Aufwände zu reduzieren, beschreibt dieser Abschnitt ein Konzept zur automatisierten Erstellung von Testfällen. Um Testfälle automatisch generieren zu können, werden strukturelle Informationen aus aufgezeichneten Netzwerkdaten benötigt. Um Kommunikationsdaten aufzuzeichnen ist eine Interaktion mit den Softwarediensten notwendig. Eine Interaktion kann z.B. das Ausführen einer Weboberfläche oder das Auslesen eines OPC UA-Servers durch einen OPC UA-Client sein. Abbildung 3 zeigt wie Kommunikationsdaten mittels Wireshark[11] aufgezeichnet werden können.

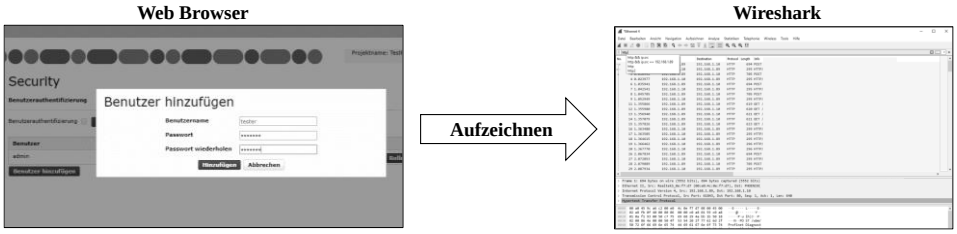



**Abbildung 3:** Aufzeichnung von Kommunikationsdaten mittels Wireshark.

Die Softwarebibliothek für das Fuzzing-Testing verfügt über die Möglichkeit die für das Generieren von Testfällen notwendigen Datenstrukturen aus den aufgezeichneten Daten zu extrahieren. Abbildung 4 zeigt die Nutzung der Software Wireshark zur Aufzeichnung und Speicherung des Netzwerkverkehrs im PCAP-Dateiformat. Dann erfolgt die Extraktion. Das Ergebnis ist eine Datenstruktur im JSON-Format.

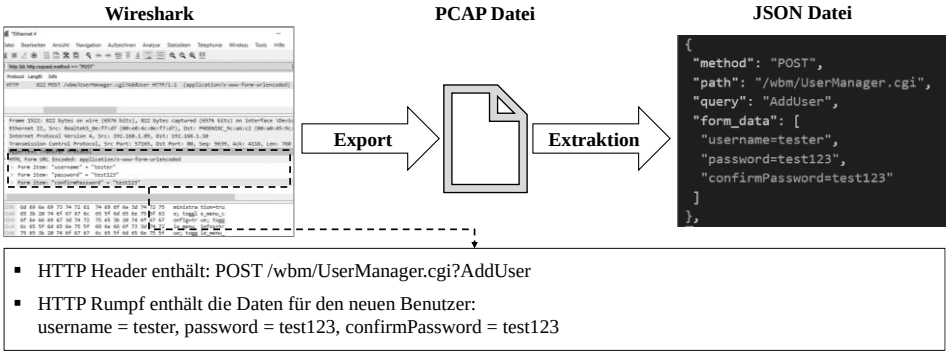



**Abbildung 4:** Extraktion von Datenstrukturen aus aufgezeichnetem Netzwerkverkehr.

[11] https://www.wireshark.org/

Die automatisierte Erstellung von Fuzzing-Testfällen ist eine Option, die durch die Softwarebibliothek ermöglicht wird. Abbildung 5 zeigt exemplarisch einen generierten Testfall: (1) Import der Softwarebibliothek, (2) Initialisierung eines Generators (z.B. Wordlist- oder Random-Generator), (3) Definition der zu testenden Funktion (hier 'AddUser'-Funktion der Weboberfläche, siehe Abbildung 3), (4) Erzeugung und Zuweisung der generierten Eingabe als Benutzername und Passwort und (5) Versandt des Netzwerkpakets und Aktualisierung des Reporting-Moduls.

**Beispiel für einen Testfall (Python Script)**

```
1  import cyberfuzz
2
3  def test_wordlist_POST_AddUser(http_client, reporting):
4      '''[Summary] Fuzzing Test for the 'AddUser' function in the web interface.'''
5
6      fuzzer = cyberfuzz.WordlistFuzzer(dataset='blns')
7
8      urlAdd      = "/wbm/UserManager.cgi" + '?' + "AddUser"
9
10     for i in range(0, 100):
11         fd = fuzzer.fuzz()
12         user = fd
13         pw = fd
14         paramsAdd = "username="+ user + "&password="+ pw +"&confirmPassword="+ pw
15
16         http_client.httpPost(urlAdd, paramsAdd)
17         reporting.updateDataStatus((urlAdd + ' | ' + paramsAdd))
```

(1) Import der Softwarebibliothek

(2) Initialisierung des Generators

(3) Definition der zu testenden URL

(4) Erzeugung und Zuweisung einer neuen Fuzzing-Eingabe

(5) Versandt des Netzwerkpakets und Aktualisierung des Reporting Moduls

**Abbildung 5:** Beispiel für einen generierten Testfall anhand der 'AddUser' Funktion in der Weboberfläche einer Industriesteuerung.

### 3.3 Konzept für die automatisierte Durchführung von Testfällen

Die manuelle Durchführung von Testfällen durch einen Testingenieur ist zeitaufwändig. Um diese Aufwände zu reduzieren, beschreibt dieser Abschnitt ein Konzept zur automatisierten Durchführung von Testfällen.

Eine Testumgebung besteht aus der zu testenden Industriekomponente und einem Test-PC, siehe Abbildung 6. Der Test-PC und die Industriekomponente kommunizieren untereinander über eine Netzwerkverbindung. Im abgebildeten Beispiel wird eine Testeingabe an den OPC UA-Softwaredienst der zu testenden Komponente geschickt. Anschließend wird der Zustand von der Industriekomponente abgefragt (Monitoring), um ausgelöste Fehler (z.B. Absturz des OPC UA-Softwaredienst, Speicherüberlauf, etc.) festzustellen.

Ein Report kann parallel zur Testausführung ausgewertet werden. Dies ist bei langen Ausführzeiten von Testfällen hilfreich. Der generierte Report enthält die folgenden Informationen: Zeitstempel des verschickten Netzwerkpakets, generierte Testeingabe, den freien Arbeitsspeicher der Industriekomponente, die CPU-Auslastung, beendete Prozesse und Alarmsignale.

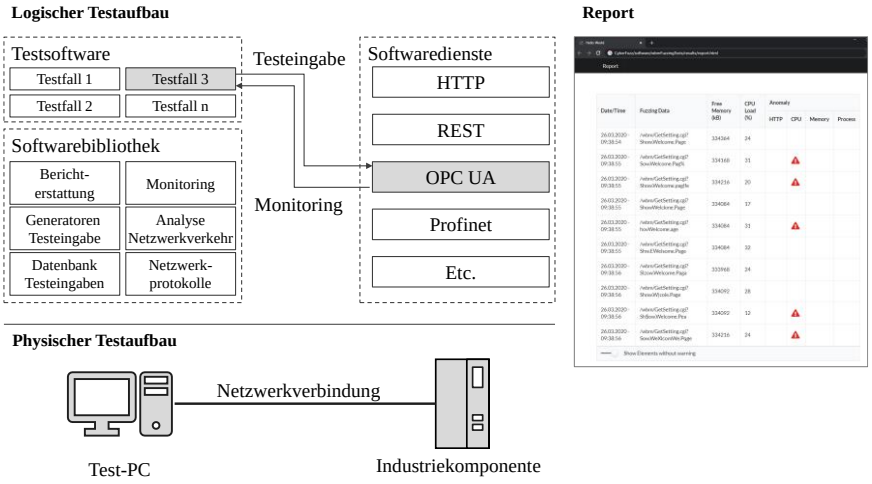



**Abbildung 6:** Konzept für die automatisierte Durchführung von Testfällen.

# 4 Einsatz der Fuzzing-Bibliothek zum Erreichen der PLCnext IEC 62443-4-1/4-2 Zertifizierung

Fuzzing-Testing von Kommunikationsprotokollen ist als Ergänzung zu den prozessualen Maßnahmen des Entwicklungsprozesses der PLCnext[12] und den funktionalen Anforderungen ein wesentlicher Bestandteil, um eine erfolgte Härtung nachzuweisen. Mit der Anwendung der IEC 62443 erweitert sich der Einsatz des Fuzzing-Testing als Bedrohungsminderung in Kombination mit dem Wissen über die Architektur und der Funktion des Produkts. Eine Analyse klassifiziert Bedrohungen in STRIDE[13]-Kategorien, z.B. Denial of Service (DoS) oder Tampering. Maßnahmen zur Mitigierung dieser Bedrohungen werden mit der Fuzzing-Bibliothek nachgewiesen. Für DoS wird eine Kommunikationsschnittstelle mit vielen Protokollen unterschiedlicher Arten belastet, z.B. gültige Protokolle, gültige Frames aber fehlerhafte Nutzdaten und vollständig fehlerhafte Protokolle. Über das Reporting wird die Auslastung, z.B. CPU, Speicher und das Dateisystem, überwacht und nach dem Bestehen des Tests, auch die Auswirkungen während und nach dem Angriff dokumentiert. Eine wichtige Erweiterung sind Protokollsequenzen, so können auch Anwendungsfälle, die sich besser an der Verwendung des Produkts orientieren, überprüft werden. Zum Nachweis der Härtung gegenüber typischen oder bekannten Tampering-Angriffen können Frames mit Angriffsvektoren verwendet werden. Wortlisten ent-

[12] „PLCnext", 2022. [Online] (https://www.phoenixcontact.com/de-de/industrien/plcnext-technology Stand 29.08.2022)
[13] „STRIDE", 2009. [Online] (https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20) Stand 29.08.2022)

halten bekannte Angriffe für z.B. Cross-Site-Scripting oder nutzen bestimmte Zeichenfolgen und deren Varianten, die Aufgrund von bekannten Schwachstellen in Third-Party-Komponenten schon zu Fehlern geführt haben.

## 5 Zusammenfassung

In dieser Veröffentlichung wurde eine Softwarebibliothek für das Fuzzing-Testing von Kommunikationsschnittstellen vorgestellt. Die Softwarebibliothek implementiert unterschiedliche Algorithmen für das Fuzzing-Testing und unterstützt das Generieren und Mutieren von Testeingaben, Extraktion von Datenstrukturen aus aufgezeichnetem Netzwerkverkehr, eine Datenbank mit Testeingaben, das Versenden von Testeingaben über unterschiedliche Netzwerkprotokolle (HTTP, OPC UA, Profinet), das Monitoring der zu testenden Industriekomponente und die automatische Erzeugung von Testberichten. Durch die Softwarebibliothek können Kommunikationsschnittstellen von industriellen Komponenten kontinuierlich auf Fehler und Sicherheitslücken getestet werden. Testingenieure werden befähigt, Fuzzing-Algorithmen zu verwenden. Eine Zertifizierung der Produktentwicklung nach IEC 62443-4-1 und des Produktes nach IEC 62443-4-2 wird unterstützt. Die Softwarebibliothek erlaubt sowohl die manuelle Erstellung und Durchführung von Testfällen als auch die automatisierte Erstellung und Durchführung dieser mit Hilfe der vorgestellten Konzepte. Dadurch werden die Aufwände für das Testen von Industriekomponenten erheblich reduziert.

## 6 Literaturverzeichnis


[Ni14] K.H. Niemann, "IT-Security-Konzepte für die Prozessindustrie.", atp edition – Automatisierungstechnische Praxis, vol. 56, no. 07-08, 2014.

[Bsi13] Bundesamt für Sicherheit in der Informationstechnik (BSI), „ICS-Security-Kompendium“, 2013.

[Bsi14] Bundesamt für Sicherheit in der Informationstechnik (BSI), „ICS-Security-Kompendium - Testempfehlungen und Anforderungen für Hersteller von Komponenten“, 2014.

[Ma19] V. Manès, et al., "The art, science, and engineering of fuzzing: A survey." in IEEE Transactions on Software Engineering, 2019.

[Su07] M. Sutton, A. Greene, and P. Amini, “Fuzzing: brute force vulnerability discovery, Pearson Education”, 2007.

[Ch10] Tsong Yueh Chen, et al, "Adaptive random testing: The art of test case diversity." In Journal of Systems and Software, vol. 83, No. 60-66, 2010.

[Ga09] V. Ganesh, T. Leek, und M. Rinard, "Taint-based directed whitebox fuzzing", in Proc. 31st IEEE International Conference on Software Engineering, 2009, pp. 474-484.

[Go08] P. Godefroid, M.Y. Levin, und D.A.Molnar, "Automated whitebox fuzz testing." Network and Distributed System Security Symposium (NDSS). vol.8, 2008.

[De07] J. D. DeMott, R. J. Enbody, und W. F. Punch, “Revolutionizing the field of grey-box attack surface testing with evolutionary fuzzing,” in Proc. BlackHat and Defcon, USA, 2007.

[Ra17] S. Rawat, et al. "VUzzer: Application-aware Evolutionary Fuzzing", Network and Distributed System Security Symposium (NDSS), vol. 17. 2017.

[Ka01] R. Kaksonen, M. Laakso, und A. Takanen, “Software security assessment through specification mutations and fault injection”, in Proc. Communications and Multimedia Security Issues of the New Century, 2001, pp. 173–183.

[Ch15] S. K. Cha, M. Woo, und D. Brumley, “Program-adaptive mutational fuzzing,” in Proc. IEEE Symposium of Security Privacy, 2015, pp. 725–741.

[Ni17] M. Niedermaier, F. Florian, und A. von Bodisco. "PropFuzz – An IT-security fuzzing framework for proprietary ICS protocols", in Proc. IEEE International conference on applied electronics (AE), 2017, pp. 1-4.

[Zh20] Z. Luo, et al. "ICS protocol fuzzing: Coverage guided packet crack and generation." In Proc. 57th ACM/IEEE Design Automation Conference (DAC), 2020.

[Pe21] P.Y. Lin, et al., "ICPFuzzer: proprietary communication protocol fuzzing by using machine learning and feedback strategies", Cybersecurity vol. 4.1, 2021.

[Pf17] S. Pfrang, D. Meier, und V. Kautz, “Towards a modular security testing framework for industrial automation and control systems: ISuTest”, in 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), 2017, pp. 1-5.

[Pfb19] S. Pfrang, und A. Borcherding, „Security Testing für industrielle Automatisierungskomponenten: Ein Framework, sein Einsatz und Ergebnisse am Beispiel von Profinet-Buskopplern, IT-Sicherheit als Voraussetzung für eine erfolgreiche Digitalisierung“, 16. Deutscher IT-Sicherheitskongress - Bundesamt für Sicherheit in der Informationstechnik (BSI), 2019.

[Pf19] S. Pfrang, A. Borcherding, D. Meier, und J. Beyerer, „Automated security testing for web applications on industrial automation and control systems”, at-Automatisierungstechnik, vol. 67, no.5, pp. 383-401, 2019.

[Pf18] S. Pfrang, D. Meier, M. Friedrich, und J. Beyerer, „Advancing Protocol Fuzzing for Industrial Automation and Control Systems”, in 4th International Conference on Information Systems Security and Privacy (ICISSP), vol. 1, 2018.

[Ot15] J. Otto und O. Niggemann, “Automatic parameterization of automation software for plug-and-produce,” in AAAI-15 Workshop on Algorithm Configuration (AlgoConf), Austin, USA, 2015.

[Ot18] J. Otto, B. Vogel-Heuser, und O. Niggemann, “Automatic parameter estimation for reusable software components of modular and reconfigurable cyber-physical production systems in the domain of discrete manufacturing,” IEEE Transactions on Industrial Informatics, vol. 14, no. 1, pp. 275–282, 2018.

[OtV18] J. Otto, B. Vogel-Heuser, und O. Niggemann, “Online parameter estimation for cyber-physical production systems based on mixed integer nonlinear programming, process mining and black-box optimization techniques,” at-Automatisierungstechnik, vol. 66, no. 4, pp. 331–343, 2018.

# Describing wireless communication systems through the Asset Administration Shell.

Gustavo Cainelli[1], Lisa Underberg[2], Lutz Rauchhaupt[3], Carlos E. Pereira[4]

**Abstract:** A crucial part of highly interconnected Industry 4.0 systems is the wireless communication system (WCS). As the WCS is one of the key elements of I4.0 and it has to cope with the level of adaptability required by Industry 4.0, it cannot be planned and installed for the worst case, which means that if changes are necessary, the WCS and/or its use of the medium should be adapted accordingly. It includes changes due to the production process requirements or changes due to external disturbances, for example, interferences. To put WCS in the context of I4.0, self-configuration is required. To enable this feature, a digital twin approach can be used, which means, the WCS can have a digital counterpart to optimize or reconfigure its parameters. For that, WCS elements must have virtual models that describe their characteristics and functionalities. This work discusses how to develop a digital model of WCS elements according to the Industry 4.0 concept using the Asset Administration Shell. Describing the communication properties in the digital domain enables the management system to monitor the WCS and take actions based on the properties' values. In this work, the authors show the relationship between the physical and digital domains pointing out the necessity to model wireless communication entities. This motivation to describe the WCS in the digital domain is shown through two use cases. Moreover, the authors explain how the wireless communication parameters can be modeled with the AAS.

**Keywords:** Industry 4.0; Wireless communication system; Digital twin; Asset administration shell; Communication resource management

## 1 Introduction

Industry 4.0 seeks to increase the efficiency of production processes both in the engineering and operational phases. In the traditional manufacturing systems (Industry 3.0) the production systems are built to produce a certain type of product and will produce it until the system is reconfigured, usually by a human. Hence, the production process sequence is determined still in the engineering phase. On the other hand, in I4.0 the sequence of the production process is decided during production based on the requirements of the product. This means that the system must be able to constantly identify what the current demands are and what

[1] ifak Institut für Automation und Kommunikation e.V., Werner-Heisenberg-Straße 1, 39106 Magdeburg, Germany, Gustavo.Cainelli@ifak.eu

[2] ifak Institut für Automation und Kommunikation e.V., Werner-Heisenberg-Straße 1, 39106 Magdeburg, Germany, lisa.underberg@ifak.eu

[3] ifak Institut für Automation und Kommunikation e.V., Werner-Heisenberg-Straße 1, 39106 Magdeburg, Germany, lutz.rauchhaupt@ifak.eu

[4] Universidade Federal do Rio Grande do Sul, Porto Alegre, Brazil cpereira@ufrgs.br

resources are available to perform the tasks accordingly [Pla]. Therefore, I4.0 compared to traditional manufacturing systems is more flexible and all details do not need to be planned in advance. As the traditional systems are not so flexible, in some cases they are planned for the worst-case which leads to a non-efficient use of resources, for example, the communication resources.

For the automation system to be flexible, the components need to provide a standardized description of their characteristics. This description contains information about the capabilities of that component or system. In this way, it is possible to determine whether or not that component can handle a specific task. In I3.0 this analysis is done during the engineering phase. However, in I4.0, flexibility requires components and systems to be able to self-adapt during runtime, and for that, model descriptions are fundamentals [JSW20].

A key element in an integrated I4.0 system is the communication system, especially the wireless communication systems (WCS) which have many advantages over the wired communication systems. In recent years, WCS has been gaining popularity in industrial applications and has been used primarily where mobility is required or wired networks are not suitable. Consequently, the WCS plays a vital role in I4.0 and has to cope with the level of adaptability required by I4.0 [Li17]. For this reason, it cannot be planned and installed for the worst case, it means that if changes are necessary, the WCS and/or its use of the medium should be adapted accordingly. It includes changes due to the production process requirements or changes due to external disturbances, for example, interferences. Therefore, for WCS to be in the context of I4.0, it should be capable of self-configuration.

One way to enable the self-configuration of WCS is through a Digital Twin (DT). In this case, the physical domain of the production process is transformed into the digital domain where the digital twins monitor and configure the production process components in real time. To develop digital twins, formal descriptions of the assets involved are necessary. Assets are entities that have value to an organization. They can be sensors, actuators, machines, documents and software, among others [Plb]. To optimize their use, they must be manageable, that is, it must be possible to monitor and configure them [JSW20]. Also, this optimization can be done automatically. For that, the assets must have a representation in the digital domain and a model description is essential for that. Consequently, the WCS also needs a formal description, which means that the communication aspects must be considered when modelling the components of a system.

To implement DTs of assets, the Platform Industry 4.0 [Plb] proposed the Asset Administration Shell (AAS). The AAS contains information that represents characteristics and behaviors of an asset. Therefore, the AAS is a tool to create fully digital versions of any kind of asset in a factory. Through the AAS an asset becomes an I4.0 component. The AAS can be used to model the main communication aspects of the production system required by the management, by compiling and exposing in an standardized way. In this paper the modelling of the communication system in the digital domain is discussed. Properties and

functions are modelled based on the AAS, this means that a communication submodel must be available for relevant automation components.

The rest of the paper is organized as follows. Section 2 shows the concepts of the physical and digital representation of the production system. Section 3 explains the asset administration shell and shows works where it is used. Section 4 describes the AAS submodels that contain communication aspects and show how to implement the AAS. Section 5 concludes and describes our planned future work.

## 2 Digital representation of production system

The digital twin is not only used for the pre-operation phase but also for the operation phase. It can monitor the physical system and changes its parameters when necessary. The development of the digital twin of the production system has two phases. In the transformation phase, the assets that compound the production system are selected and transformed into virtual descriptions through models. These models contain all necessary assets' information including their properties and functionalities. With these models, the design engineer can identify what tasks each asset can perform. In the operational phase, the virtual counterparts are not only a formal description of the asset but also have monitoring and decision-making functionalities. In this phase, the virtual counterpart reflects the current state of the physical part and can control it.

### 2.1 Transformation phase

The production system can be divided into three main parts. First, the mechanical system that is composed of elements like robotic arms, plates, rods, tables, wheels, etc. Second, the automation system composed of CLPS, sensors, actuators, controllers, among others. Finally the communication system which can be wireless or wired, including modules, routers, base station, etc. The production system containing these three layers is in the physical domain. The physical domain includes not only hardware, but also software. For representing the physical domain in the digital domain the assets go through the model building process, where their characteristics are described through models. Left hand side of Fig. 1 shows the transformation of the physical domain into the digital domain.

In the process of transforming the physical domain into the virtual domain, the assets that must have a virtual representation are selected, that is, not all assets will have a digital version. Furthermore, even if an asset is selected to have a digital version during the transformation process, not all of its characteristics will be represented in the digital domain. This means that only the relevant properties of that specific asset will be considered in the model. To determine the relevancy of a property, the use case in which the digital system will act must be analysed. The focus of this work is on configuring and optimizing the

communication system. Thus, aspects of the communication system must be considered in the models and other ones are left out. For example, aspects of the mechanical system like dimensional aspects (width, height), type of material and weight are not considered.

To model assets in the transformation phase, an standardized description must be used to allow the interoperability of components from different vendors. The AAS can be used in this case to describe the characteristics, properties, and capabilities of the asset [Pla].

### 2.2 Operation phase

In general, the digital system will act in parallel with the physical system during the production process. The digital system will receive data from the physical domain to keep up to date on the current condition of the physical part, while the physical domain will receive commands from the digital domain. In other words, the digital system monitors and acts on the physical system in order to solve problems that are not possible to solve directly in the physical domain, or even possible, but not as efficiently as in the digital one. In this way, the relevant properties of the assets are derived from the problem to be solved.

In the operational phase, the digital domain monitors and controls the physical domain. For this reason, data exchange between the domains is required. The data which will be exchanged between domains and intra-domains are distinguished into three types:

**Productive data exchange:** It is the data exchange between elements within the physical domain. It is the data traffic of the production process. For example, a sensor sends its variable process to a PLC that uses it to calculate the control action and sends a command to an actuator.

**I4.0 data exchange:** It is the data exchange within the digital domain. For example, a digital twin sends a request to another one to adapt a system parameter.

**Management data exchange:** It is the data exchange between domains. It is used for example for the digital twin to monitor and reconfigure its asset.

The three types of data exchange can be seen in the right hand side of Fig. 1.

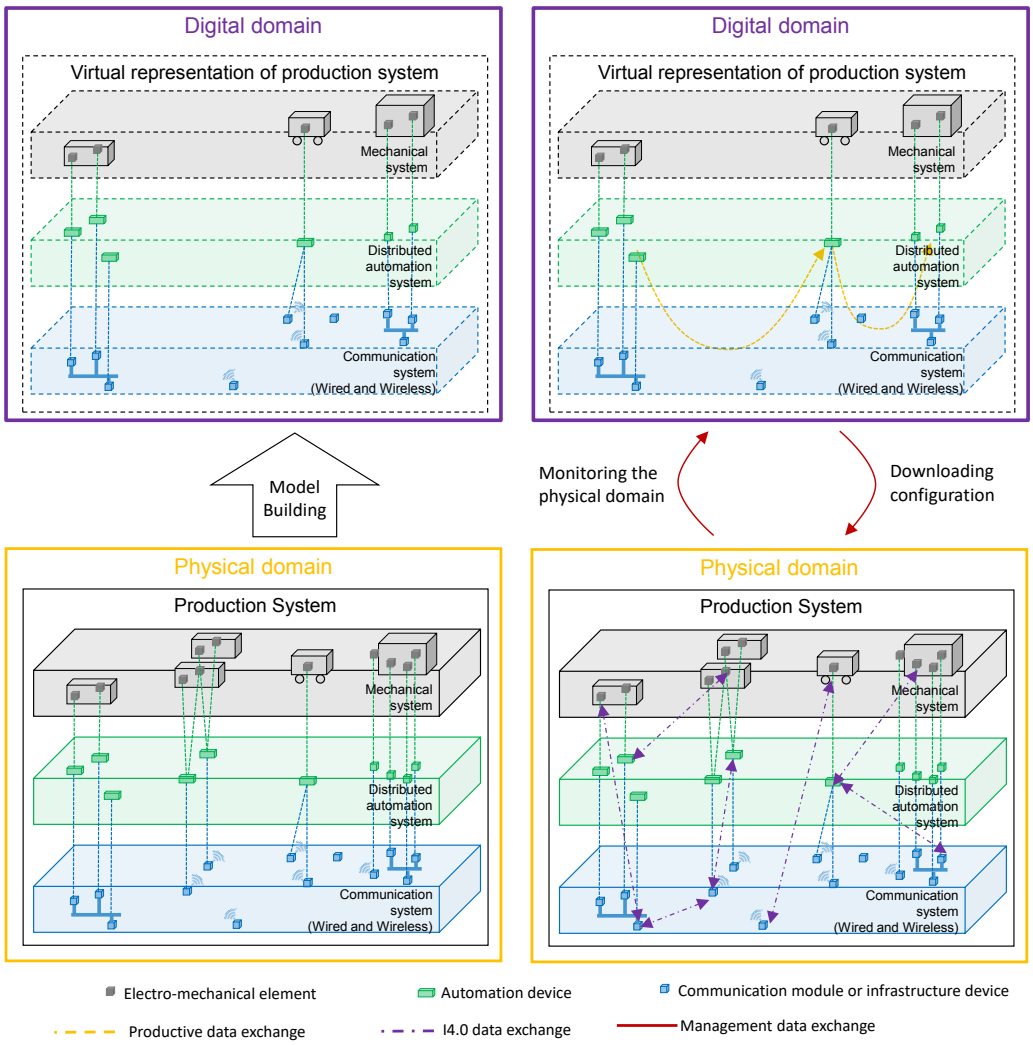



Fig. 1: Physical and digital domain: Transformation and operation phase.

In the operational phase the AAS acts as the asset's digital twin monitoring and controlling the asset. The parts of an AAS and some of its uses are described in the next section.

## 3 Asset Administration shell

The AAS is a concept to implement the Digital Twin for Industry 4.0. The AAS contains information that represents the characteristics and behaviors of assets. It can be used to create fully digital versions of any kind of asset in a factory. It does not only provide standardized and machine-interpretable data, but also provides information about the functionality offered by the asset making it an I4.0 component [Pla]. An I4.0 component is formed by asset and AAS as show in Fig. 2. The asset is in the physical domain (yellow) and the AAS in the digital domain (purple).

The AAS can be passive, reactive, and proactive regarding the interaction pattern. More details of each one can be found in [Plb]. An proactive AAS contains two main parts: the passive and the active. The passive part is the modelling of the asset based on its properties and functionalities. This information is readable and/or modifiable. Some examples are static device configuration, identification data, and current parameters' values. The active part consists of procedures and algorithms performed by the AAS. It can read and write properties' values in the AAS or the asset. Moreover, it has decision-making functionalities. Some examples are status assessment and status control, when the counterpart on the digital domain (AAS) assesses the counterpart on the physical domain (asset) and adapts it through configuration commands.

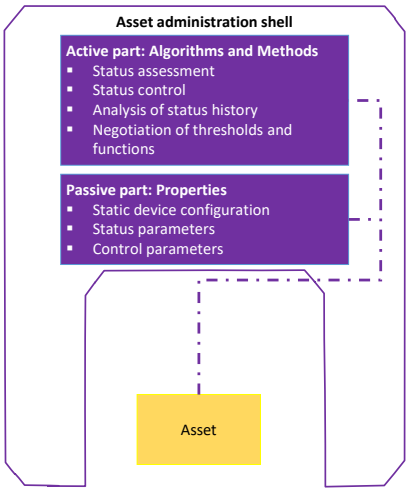



Fig. 2: I4.0 component with asset and its proactive AAS.

Currently, the use of AAS to manage the production system is the subject of several research works. The AAS is used to manage a single asset, an entire systems, and to solve interoperability issues. In order to increase interoperability in bidirectional data exchange between corporate (file-based data, e.g. Excel) and control applications (Ethernet-based,

e.g OPC UA), the authors in [Ye22] propose a conversion between AASX files and Excel spreadsheets. AASX is a package file format for representing AAS. An example scenario of data exchange for motor control is presented to validate the effectiveness of the proposal. In [YH19] a review of the current status-of-art of AAS is presented. Furthermore, the authors propose a general model for an AAS. Finally, they present an interoperability case where assets of a heterogeneous system (Wi-Fi, Powerlink, and UART) can interact with each other. In [Ye20] and [Zh21] the authors present the status quo of AAS development and propose an intuitive method for implementing AASs. The proposal includes a cluster of AASs deployed on an edge AAS host that acts as a hub to collect runtime asset data. An Industry 4.0 application scenario Plug-and-Produce, with three robots and a turntable, is presented. The AAS of the robots negotiate in the digital domain when an asset needs to jump in or jump out of the production process in the physical domain. The work of [In20] tests interoperability between a robotic arm and grinding machine by integrating them through AAS. In [PBD21] first, the authors compare the available tools to implement AAS. Then they present a methodology that significantly reduces the integration effort and computing power requirements for AAS applications. Instead of a central platform that host the AASs, each asset that is able to communicate through OPC UA, has its AAS embedded in it. In [Lv20] the authors present a framework for managing edge assets based on AAS. Each asset has its AAS and an extra AAS is created in order to manage the others. The management node simplifies the underlying device structure and performs asset management The authors in [SLl22] present a proposal for an agent-based AAS. The strategy was applied to an industrial automation system in order to analyse its applicability. The objective of the method is to improve the asset digitization process, considering agents to incorporate intelligence and collaborative functions, and increase interoperability.

In the mentioned works the submodels considered are general ones like identification and documentation and specific ones related to particular properties of the devices, like the position of the robotic arms or processing variables of sensors. Although some works like [YH19] and [Ye20] indicates that a communication submodel is required when modelling AAS, they do not provide details information about it. Furthermore, these works do not have the same point of view as our proposal, which consists of model the communication submodel not just for communication between assets, but to use this information to manage and adapt the production system.

Describing the WCS as an AAS is not simple due to the number of different components and also the complexity of each one. Taking as an example 5G technology, the Platform I4.0 proposes describing the 5G system as different AAS entities, e.g. for the 5G automation device (5G-UE AAS), for the Radio Access Network (5G-RAN AAS) and for the Core Network (5G-CN AAS). The 5G-ACIA [5G] proposes the use of two AAS called 5G-UE-AAS for 5G automation devices and 5G-NW-AAS for the entire 5G network (RAN and CN). Therefore it is necessary to carefully define which components of the WCS will be described through the AAS.

## 4 AAS of communication system

Asset modeling through AAS has different levels of detail. It can be highly complex if all asset details are included in the AAS. Thus, it is necessary to carefully evaluate which are the assets and properties of each asset that will be considered in the transformation from the physical into the digital domain. The use case will determine which assets should have a virtual representation (digital twin). This means that AAS will be developed according to the needs required by the use case. The communication aspects (focus of this article) must be considered in the model whenever the communication system has to have the ability to negotiate with the production system to adapt to new product requirements or even external interference that influence the communication channel. These two cases can be described by the use cases below:

**Use case 1: "Managing the WCS based on production requirements".** In this use case, the parameters of the wireless communication system are adjusted based on requirements that come from the production system. For example, if a product has priority, then more communication resources are allocated to the machine that produces it. In other words, the WCS is adapted based on production and automation system requirements.

**Use case 2: "Production process management based on the quality of communication".** In this use case, the parameters of a production process element (e.g. machine, device, AGV) are adapted based on the quality of communication. For example, if the wireless channel is under interference and messages are being lost, the production system is adapted to cope with that. In other words, the production system is adapted based on the WCS.

For both use cases, several parts of communication system must be considered, for example, base station, scheduling algorithms and wireless modules of automation devices. In this work, we use as an example an automation device, which is a device that performs automation functions (sensors, actuators, machines) and has communication capabilities. For the automation device, not only the communication aspects but also the general aspects of the asset are important. Therefore some submodules published by IDTA [In22] are also used here.

**Nameplate** This submodel was published by IDTA and provides process industry and factory automation equipment nameplate information to the respective AAS in an interoperable manner. More details can be found in [In22].

**Technical Data** This submodel provides technical data describing the respective AAS asset. The central element is the provision of interoperable properties through dictionaries such as ECLASS and IEC CDD (Common data dictionary). More details can be found in [In22].

**Location** Although it is not the focus of the work, a submodel called Location is suggested in this work. A wireless device will eventually change position or may still be embedded in an AGV. It must contain data such as current position (xyz), status (moving, stopped, accelerating, decelerating), speed, among others.

**Application** In the example of this work, the application part of the automation device and the communication are kept in different submodels. In the application, there are information related to the production process such as the type of the data traffic that the device transmits (periodic, aperiodic), the transmission interval (time between sending two messages) and number of bytes in the message.

**Communication** The communication submodel contains all the relevant information related to the communication. Most information is stored in the submodelcollection 'WirelessModule'. Some properties are static and represents the capabilities of the 5G module as delivered. This data generally can be found in the module's documentation (datasheets). Examples are list of frequency band that the device is compatible with, maximum transmit power, and information related to identification (e.g. manufacturer). The dynamic properties are updated during the operational phase. It includes RSSI, RSRP, and current output power.

These submodels briefly described above form the passive part of the AAS as Fig. 3 shows. The figure also shows the blocks that form the active part of the AAS. The active part is responsible for exchanging messages with other AASs and also for asset control. Some of the basic blocks need to form an proactive AAS are shown. The blocks interaction with physical and digital domain are respectively responsible for management data exchange and I4.0 data exchange. The asset's properties are monitored constantly, that means the AAS read the values of the asset and update the values on the passive part. For that, read/write block are used. The AAS can receive/send requests from/to other AAS and for that and interaction with digital domain is required. Moreover, the AAS keeps a register of its properties' values. This data can be later used by AI algorithms for optimization purposes.

The communication-related process contains the blocks responsible for tasks related to communication. In our case, it monitors properties such as SINR and RSSI. The performance parameters as transmission time and update time are handled by the application-related block. The application performance and the communication have interdependency, which means the communication system has an influence on performance parameters. More details about performance parameters can be found in [UR19]

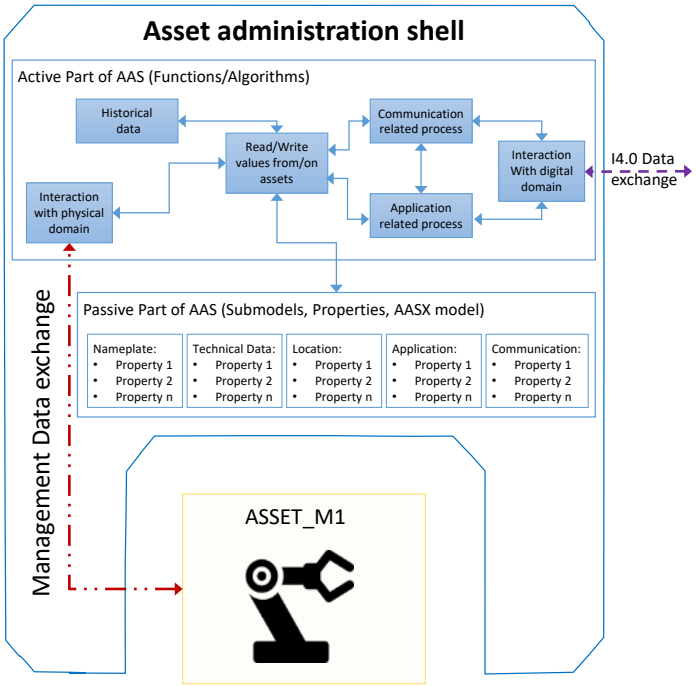



Fig. 3: Graphical representation of an asset and its AAS with active, passive and integration parts.

Some steps are necessary to model the communication system with AAS. Fig. 4 shown these steps. Initially, the use case must be analyzed in order to identify which are the main elements of the specific case that are relevant to have a virtual representation. Next, the project engineer must define for each of the asset types which are the properties of that asset that must be modeled. For this it is important to understand what role that asset plays in the use case. Once this is done, the asset information is grouped into submodels through AASX PE. This is the passive part of the AAS and contains all the asset parameters that are somehow relevant to the use case.

The file generated by AASX PE has the extension .aasx and must be converted to be used in the OPC UA server. AASX PE itself has an option to export AAS in nodeset2.xml format. The generated nodeset is the base file for generating nodes in OPC UA. The library open62541 [op] is used to create OPC UA servers and clients. During the compilation of the code, the nodeset file is used as a base to generate the node types.

In parallel with the asset modeling with AASX PE, the algorithms and functions that represent the active part of the AAS are developed. These algorithms are based on what the AAS will be used for. For example, functions are needed that allow the AAS to interact with

the physical domain and also with the virtual domain. Furthermore, in the case of this work, functions related to communication were also developed. Finally, the code is compiled generating an application (e.g. aas.exe) which is the virtual representation of the asset.

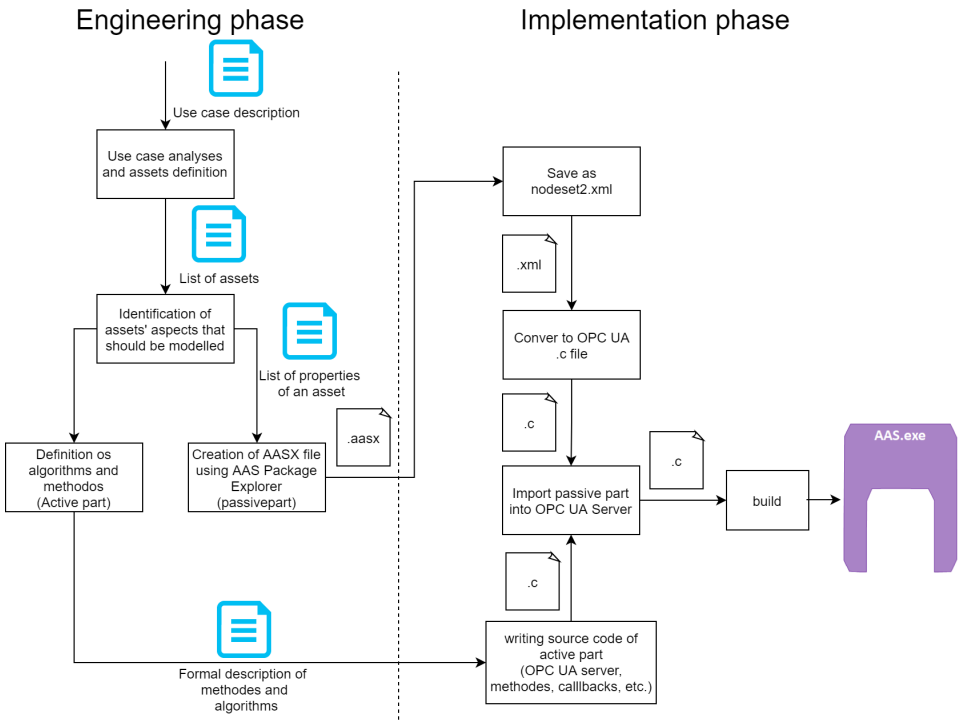



Fig. 4: Engineering and implementation phase.

After compiling the developed code, AAS is ready to be used. The developed AAS is based on an OPC UA server, and thus can be accessed by an external client. Fig. 5 shows parts of the AAS developed for an automation device. The left side of the figure shows the submodules in AASX PE. On the right side are the same submodels and properties accessed by the OPC UA Expert on the OPC UA server running. AASX PE is used during the transformation phase while OPC UA is used in the implementation and operation phase.

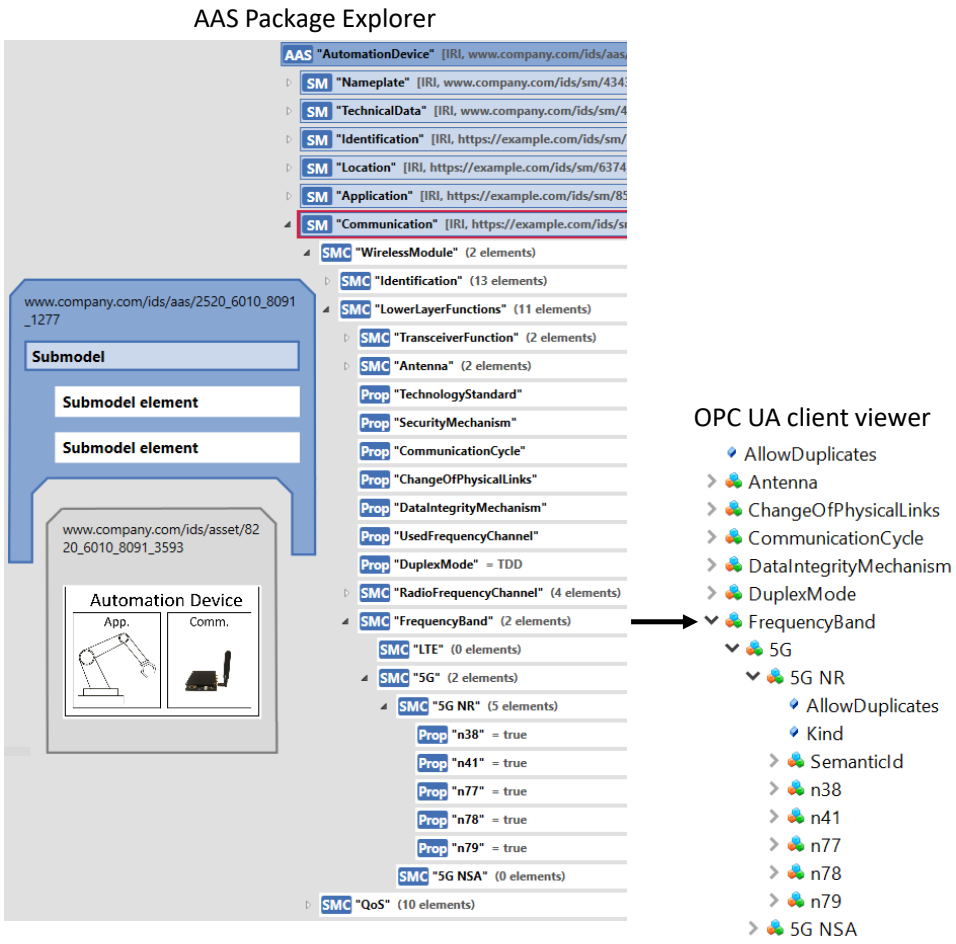



Fig. 5: AASX PE and UA Explorer.

## 5 Conclusion

This paper described the steps to develop an AAS of an automation device that communicates wirelessly. Between the proposed submodels the WirelessModule brings the information of the wireless module to the digital domain. It includes static information such as the manufacturer name and dynamic information such as RSSI. A general architecture that shows the active and passive part of an AAS is presented. Initially it was shown that the production process is transformed from the physical into the digital domain. The selected assets are modelled using AAS. This virtual, standardized model serves as the asset's digital twin during the operational phase. The AAS exchanges information with the asset to monitor and configure the asset. The focus of this work is on the communication system, and thus the main aspects considered in the proposed modules are related to device communication.

## 6 Acknowledgements

This work was funded in parts by the project *"5GIWCoW"* under contract *165GU041D*, funded by the Federal Ministry of Transport and Digital Infrastructure, Germany.

## Bibliography

[5G] 5G-ACIA: , Using Digital Twins to Integrate 5G into Production Networks (White Paper). https://5g-acia.org//. Accessed 27 Nov. 2020.

[In20] Inigo, Miguel A.; Porto, Alain; Kremer, Blanca; Perez, Alain; Larrinaga, Felix; Cuenca, Javier: Towards an Asset Administration Shell scenario: a use case for interoperability and standardization in Industry 4.0. In: NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium. pp. 1–6, 2020.

[In22] Industrial Digital Twin Association (IDTA): , AAS Submodel Templates. https://industrialdigitaltwin.org/en/content-hub/submodels, 2022. Accessed: 08 Aug. 2022.

[JSW20] Jasperneite, Juergen; Sauter, Thilo; Wollschlaeger, Martin: Why We Need Automation Models: Handling Complexity in Industry 4.0 and the Internet of Things. IEEE Industrial Electronics Magazine, 14:29–40, 03 2020.

[Li17] Li, Xiaomin; Li, Di; Wan, Jiafu; Vasilakos, Athanasios; Lai, Chin-Feng; Wang, Shiyong: A review of industrial wireless networks in the context of Industry 4.0. Wireless Networks, 23, 01 2017.

[Lv20] Lv, Bingshuo; Zhang, Yunpeng; Yang, Fan; He, Jianping: Edge Asset Management based on Administration Shell in Industrial Cyber-Physical Systems. IECON Proceedings (Industrial Electronics Conference), 2020-Octob:3817–3822, 2020.

[op] open62541: , Open Source OPC UA. http://www.open62541.org/. Accessed: 08 Aug. 2022.

[PBD21] Pribiš, Rudolf; Beňo, Lukáš; Drahoš, Peter: Asset administration shell design methodology using embedded opc unified architecture server. Electronics (Switzerland), 10(20), 2021.

[Pla] Platform Industrie 4.0: , Describing Capabilities of Industrie 4.0 Components (White Paper). https://www.plattform-i40.de/. Accessed: 08 Aug. 2022.

[Plb] Plattform Industrie 4.0: , Details of the Asset Administration Shell from idea to implementation. https://www.plattform-i40.de. Accessed 27 Nov. 2020. Accessed 27 Nov. 2020.

[SLl22] Sakurada, Lucas; Leitao, Paulo; la Prieta, Fernando De: Agent-Based Asset Administration Shell Approach for Digitizing Industrial Assets. IFAC-PapersOnLine, 55(2):193–198, 2022. 14th IFAC Workshop on Intelligent Manufacturing Systems IMS 2022.

[UR19] Underberg, L.; Rauchhaupt, L.: Performance Testing of Novel Wireless Communication Networks for Industrial Automation on the Example of 5G. In: KommA 2019. Jahreskolloquium "Kommunikation in der Automation". 2019.

[Ye20] Ye, Xun; Jiang, Junhui; Lee, Changdae; Kim, Namhyeok; Yu, Mengmeng; Hong, Seung Ho: Toward the Plug-and-Produce Capability for Industry 4.0: An Asset Administration Shell Approach. IEEE Industrial Electronics Magazine, 14(4):146–157, 2020.

[Ye22] Ye, Xun; Song, Won Seok; Hong, Seung Ho; Kim, Yu Chul; Yoo, Nam Hyun: Toward Data Interoperability of Enterprise and Control Applications via the Industry 4.0 Asset Administration Shell. IEEE Access, 10:35795–35803, 2022.

[YH19] Ye, Xun; Hong, Seung Ho: Toward Industry 4.0 Components: Insights Into and Implementation of Asset Administration Shells. IEEE Industrial Electronics Magazine, 13(1):13–25, 2019.

[Zh21] Zhang, Xiongfeng: An Industry 4 . 0 Asset Administration Shell-Enabled Digital Solution for Robot-Based Manufacturing Systems. IEEE Access, 9:154448–154459, 2021.

# Model-Based Test Case Generation for Compliance Checking of Reactive Asset Administration Shells

Björn Otto[1], Karsten Meinecke[2], Tobias Kleinert[3]

**Abstract:** The Digital Twin is a key component for Industry 4.0. For this purpose, a standard exists in the form of the Asset Administration Shell (AAS), which enables the interoperability of implementations. In this standard, a REST API is specified to access dynamic content of AASs[Pl21]. To ensure conformity of implementations of the REST API by different manufacturers, suitable methods for quality assurance have to be applied. For this purpose, this work presents an approach that generates test cases based on an OpenAPI description of the AAS using methods of model-based testing. In addition, the approach is applied to three existing implementations for evaluation.

**Keywords:** Asset Administration Shell; Test; OpenAPI

## 1 Introduction

One key concept of the Industry 4.0 is the so called *Digital Twin*. A Digital Twin is the digital representation of a physical asset. This representation can be used in various contexts and scenarios. One of these scenarios is to model the asset's state using well-defined data structures and interfaces. For this use case, the AAS has been standardized[Pl20]. It consists of a meta model to describe an asset's data in a well-defined way. Furthermore, the standard defines methods to exchange AAS data. These methods include a simple file-based method to exchange static data. For dynamic data, i.e., data which changes over time, the AAS standard defines an Hypertext Transfer Protocol (HTTP) Representational State Transfer (REST) interface. This can be implemented by server instances to provide clients with the asset's data using endpoints as described in the standard.

Since the advent of the AAS standard, the number of server implementations is constantly increasing. As proposed, one of the core claims of the AAS is interoperability. However, since the server implementations are developed by different manufacturers and for different platforms and programming languages, interoperability is difficult to maintain. For this reason, it is necessary to establish effective testing procedures[4].

[1] Institute for Automation and Communication, Werner-Heisenberg-Straße 1, 39106 Magdeburg, Germany bjoern.otto@ifak.eu

[2] Institute for Automation and Communication, Werner-Heisenberg-Straße 1, 39106 Magdeburg, Germany karsten.meinecke@ifak.eu

[3] RWTH Aachen University, Chair of Information and Automation Systems for Process and Material Technology, Turmstraße 46, 52064 Aachen, Germany kleinert@plt.rwth-aachen.de

[4] The Industrial Digital Twin Association (IDTA) already found the working group "Quality Management"to address this.

In this work, the test setup as shown in Figure 1 is used as basis.

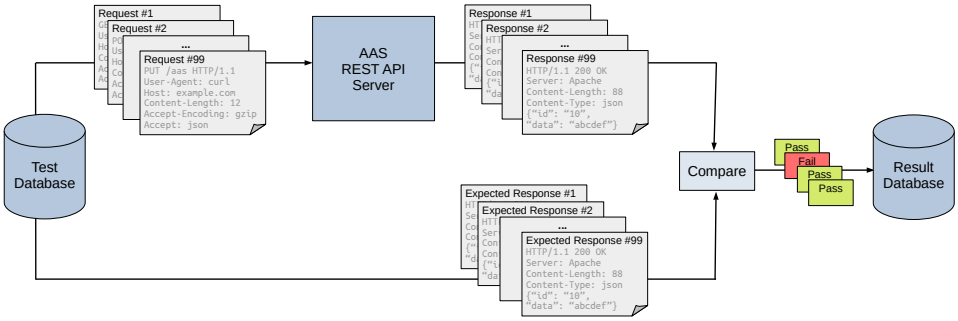



Fig. 1: Test Execution

As illustrated, testing relies on an established test database. The test database contains HTTP requests and the expected corresponding HTTP responses. During test execution, the HTTP requests are pulled from the test database and send to the AAS server instance. The server's responses are collected and compared to the ones from the test database. If a server's response matches the expected one (or the response's meta model), the associated test case is considered *passed*, otherwise *failed*. This way, the compliance of the server instance against the specification can be assessed in detail.

The last part that is necessary, is the creation of an appropriate test database, more precisely the requests and associated expected responses. One approach is to generate the database manually. However, this method has many drawbacks. First, it requires a very detailed knowledge of the specification. Furthermore, it is very time-consuming and error-prone. Even worse, it has to be performed for each new version of the AAS specification. Lastly, it is difficult to parameterize, for example towards desired specification coverage or the size of the test database.

Instead, the test database should be generated *automatically*. For this purpose, the OpenAPI description of the AAS REST API can be leveraged. An OpenAPI description is a machine-readable documentation of a REST API. It has already been used for automatic test case generation in various contexts successfully [Ar17, KČS20, HDD22].

In this work, we give an overview of existing work for OpenAPI-based test case generation. Then, we select a few methods and re-assemble them to form an algorithm tailored for the AAS. Finally, we evaluate our algorithm on an existing AAS implementation. Our goal is to reveal weaknesses and challenges of test case generation approaches given the special requirements of the AAS as a guide and a baseline for future work.

## 2 Background

In this section we would like to introduce a few topics which are needed for understanding the context of this work.

### 2.1 Hypertext Transfer Protocol (HTTP)

The HTTP [Ni99, FR22a, FR22b] is the fundamental protocol for data exchange in the World Wide Web. It features a simple request-response flow between a client and a server, as shown in Figure 2.

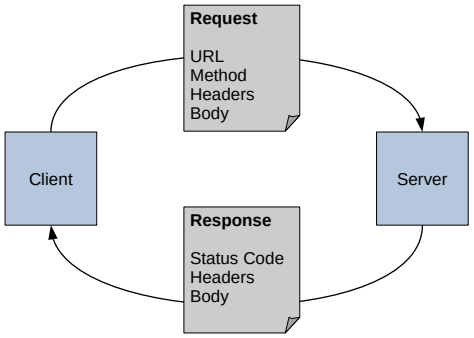



Fig. 2: HTTP request and response flow

The request sent by the client, consists of four parts:

- the requested **URL**, e.g., `/index.html`
- the **method**, which basically determines the action to perform, e.g., `GET` or `DELETE`
- the request **headers** to specify additional meta information, e.g., the client's supported languages
- the request **body**, if any, which contains the actual data

The server answers with exactly one response, which consists of three parts:

- a **status code** which determines if the operation was successful (values 200-299) or failed (values 400-599).
- response **headers** containing additional meta information, e.g., the language of the response body
- the response **body** containing the actual data

In common applications, the client will send various requests. From the HTTP view, all these request-response pairs are considered as self-contained, independent transactions.

## 2.2 Representational State Transfer (REST)

REST [FR14] is an architecture for HTTP-based server interfaces. It is centered around the abstract concept of *resources*, which can be retrieved and modified. The actual resources provided by a REST interface highly depend on the use case. They can range from users over blog posts to more abstract resources like permissions.

Especially, REST defines the format of HTTP requests to access resources. This includes the exact definition of HTTP methods and their semantics as shown in Table 1.

Tab. 1: HTTP methods and their semantic as defined by REST

| Method | Semantic |
|---|---|
| GET | Retrieve resources |
| POST | Create a resource |
| PUT | (Fully) update a resource |
| PATCH | (Partially) update a resource |
| DELETE | Delete a resource |

## 2.3 OpenAPI

The OpenAPI specification[Op21] defines a schema to describe REST APIs. To be machine-readable, this schema can be either serialized as JSON or YAML files. Instead of explaining every detail of the standard, we describe the basic concepts using an example. It can be found in Listing 1 and describes a simple REST API which allows fetching of video information.

```yaml
openapi: 3.0.3
paths:
  "/videos":
    get:
      parameters:
        - name: id
          in: query
          required: true
      responses:
        '200':
          content:
            application/json:
              schema:
                properties:
                  title:
                    type: string
                  duration:
                    type: number
```

List. 1: Example OpenAPI description of a video API

After the version information in line 1, the paths of the API are listed. In our case, the API only exposes one endpoint with the path `/videos` (line 3). This endpoint can only be invoked with the `GET` method (see Table 1). The request expects one parameter, the `id` of the video in question, which is a required parameter (line 5-8). On success, the server will respond with a status code 200 (line 10). The response body will contain the requested video information then, which are its `title` and `duration` (line 14-18).

While the example above only exposes one endpoint with one parameter, real-world REST APIs have various endpoints, where each of them can have various parameters and responses.

### 2.4 Asset Administration Shell

The AAS is a standard for the data structures of Digital Twins. For this, the AAS meta model is defined in [Pl20]. An excerpt of this meta model is shown in Figure 3. It states that an AAS consists of various *Submodels*. Submodels correlate to different aspects of the asset across the value chain like installation, configuration or operation.

Each Submodel, in turn, consists of various *SubmodelElements*. SubmodelElements describe the actual properties of the Submodel. This can be done, e.g., in form of a simple value-property, a value range, or a file. Especially, a SubmodelElement can be a collection of SubmodelElements, which renders the meta model recursive.

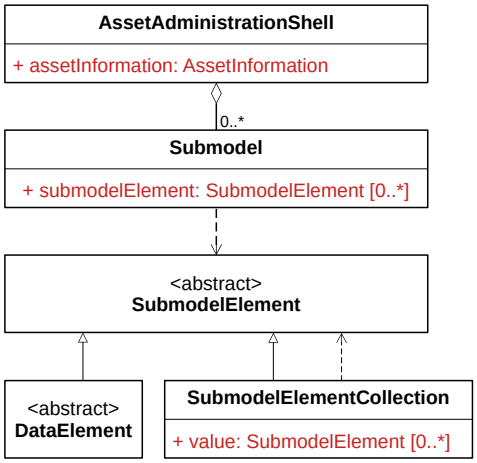



Fig. 3: AAS meta model (excerpt from [Pl20])

The data of an AAS can be exchanged in two ways:

- By serializing the AAS to file. This appearance is referred to as *passive AAS*. The standard[Pl20] defines exchange formats, including JSON and XML.

- By serving the data of the AAS by a server instance. This is called *reactive AAS*. The standard[Pl21] defines the REST API to access the AAS. The JSON serialization format for request and response data is inherited from the *passive AAS*, above.

This work focuses on the reactive AAS. It offers various endpoints, which can be categorized as follows (list not conclusive):

- **Asset Administration Shell**: gives access to complete, distinct AASs.
- **Submodel Interface**: gives access to complete, distinct Submodels.
- **AASX File Server Interface**: returns an AAS serialized as *passive* AAS.
- **Registry Interface**: allows to register and query target server for AAS identifiers.

An AAS server can (and probably will) only serve a subset of the above endpoints.

## 3 Related Work

Since the release of version 3.0 of the OpenAPI specification[Op21] in 2017, the number of publications leveraging OpenAPI descriptions to assess REST implementations is constantly increasing.

In [Ar17] the authors present *EvoMaster*. This tool aims to reveal bugs using a white-box testing approach. It utilizes coverage in addition to the OpenAPI specification to generate new test cases using an genetic algorithm. This approach is for the most part not applicable in our test setting, which requires independency from platform and programming language (black-box).

In [AGP18] the authors propose a fuzzing approach based on OpenAPI descriptions. Their tool, *REST-ler*, aims to infer as much information about the API as possible to reveal bugs. They do so by guessing dependencies between requests from the OpenAPI description and by reusing response-data from subsequent requests. Since the approach is search-based, the resulting test database is not fixed but depends on the implementation to test, which is not applicable in our test setting.

The authors of *RestTestGen*[VDC20] try to create the endpoint dependency graph automatically, too. They also highlight the importance of negative testing, i.e., creating erroneous requests, intentionally. We re-assembled the concept of the dependency graph in our approach. However, we aim to infer dependencies from OpenAPI's links feature instead of guessing them.

The tool *QuickRest*[KČS20] is based on *TestCheck*, a property-based testing tool. The authors convert OpenAPI specifications into TestCheck compatible constraints and test oracles. The

actual test generation is then done by TestCheck. A similar approach is implemented by *Schemathesis*[HDD22]. It leverages Hypothesis [MHC19] for test case generation. These tools aim to find samples for the JSON schemas of the OpenAPI specification automatically. We derived this idea for our approach, but only for simple schemas. For more complex schemas, like the `AssetAdministrationShell` object, we fall back to providing samples manually.

While the approaches target functional testing, OpenAPI specifications can be used for non-functional testing as well. For example, in [Bu20] an approach for monitoring performance and availability of a REST service is proposed. Non-functional testing is out of the scope of this work.

## 4 Proposed Approach

In this section we present our approach to generate test cases for an AAS server based on the AAS OpenAPI description.

### 4.1 Requirements

The purpose of the generated tests is to compare different AAS implementations in terms of their compliance to the specification. This means, that for all implementations, the same test database must be used, i.e., the test database is fixed. Consequently, search based methods are not applicable.

Furthermore, we want to decouple the test generation from the test execution, to e.g. feature test-driven development. This requires the test generation to be independent of the actual platform or programming language of the implementation to test. Therefore, the resulting generation must be a black-box approach.

Finally, our test case generation must cope with the complex meta model of the AAS, which is deeply nested and recursive.

### 4.2 Architecture

The overall architecture of our approach is shown in Figure 4.

First, the OpenAPI specification is parsed to extract the JSON schemas as well as the API's endpoints. The schemas are utilized to generate a sample database. The sample database is augmented with manually created samples. Next, the dependencies between endpoint parameters are analyzed to order them. Finally, the test case generation utilizes the sample database to generate appropriate request parameters and bodies.

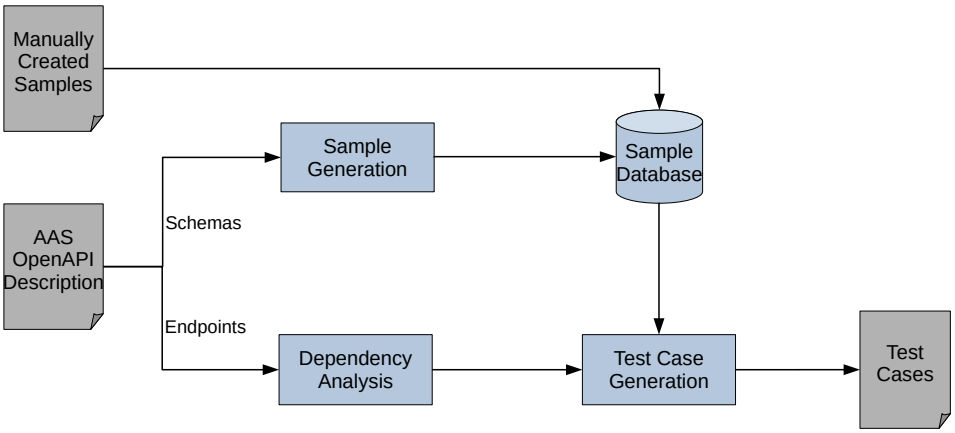



Fig. 4: Test Generation Pipeline

### 4.3 Sample Generation

The later test case generation needs JSON data which fulfills certain JSON schemas. These samples are needed to generate correct (and incorrect) request bodies and parameters. For this, the sample database is built, from which samples are drawn. It contains either manually selected samples and auto generated ones.

Auto generated samples are derived from a schema using some simple approaches:

- For **primitive types**, like strings and numbers, random values are generated.
- For **enums**, a sample for every possible enum value is selected.
- For **arrays**, the empty array is selected as sample.
- For **objects**, required attributes are iterated and generated as described above. Optional attributes are omitted.

This method is sufficient for simple JSON schemas, but fails to generate complex samples. An important example for this is the `AssetAdministrationShell` class. To fully describe a valid AAS, additional constrains have to be fulfilled, which cannot be expressed using JSON schema. For this reason, the sample database is augmented with manually created samples.

### 4.4 Dependency Analysis

The endpoints of the AAS REST interface are not independent from each other. Consider the following flow as an example: A client application first retrieves all Submodel Elements, to get the `idShortPath` of a certain instance. Then, the client uses that `idShortPath` to invoke a operation on that Submodel Element which returns the `handleId` of the ongoing operation. By using the `idShortPath` and the `handleId`, the client can poll for the operation's result. Finally, the client could delete the Submodel Element. Obviously, these calls are not independent from each other, but must be carried out in that order to use results in later requests.

To take this into account, OpenAPI's links are leveraged. A link defines the dependency between two endpoints. So, by parsing all links, a dependency graph is built. The dependency graph of the above example is shown in Figure 5.

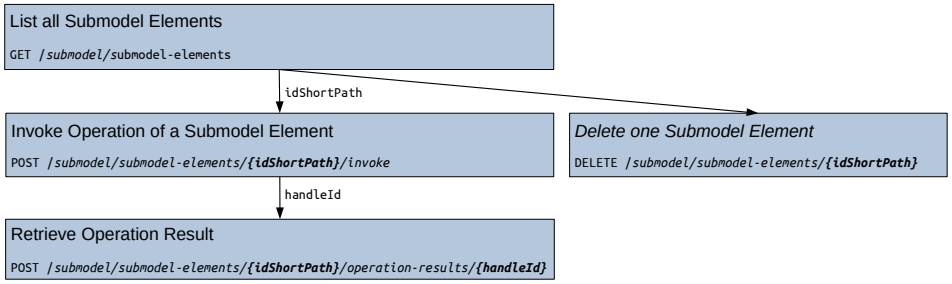



Fig. 5: AAS Endpoint Dependencies (Excerpt)

By analyzing the whole dependency graph, the endpoints are ordered. This order is passed to the test case generation.

### 4.5 Test Case Generation

To generate test cases, our algorithm iterates the endpoints as ordered by the subsequent dependency analysis. For each endpoint, i.e., each path-method-pair, it generates *positive* and *negative* test cases. Here, a positive test case is a test case with correct parameters, while a negative test case involves wrong parameter values and should consequently trigger an error response.

For positive test cases, the following variables can be chosen:

- The **request body**, if any, must follow the scheme. The algorithm must try out all valid samples from the sample database.
- All **required parameters** must be present and valid. The algorithm must try out all valid parameters.

- Every **optional parameter** can be either present and one of the valid examples or missing.

The response must match the schema defined in the OpenAPI specification for a successful HTTP response (i.e. with status code 200-299, [FR22b]). For negative test cases, our algorithm starts with an arbitrary positive test case and introduces exactly one of the following errors:

- The **request body**, if any, is invalid. The algorithm must try out all invalid samples from the sample database.
- A **required parameter** is either left out or set to an invalid value.
- An **optional parameter** is set to an invalid value.

Please note, that negative test cases are incorrect regarding the *format*. There are also requests, which are *semantically* incorrect, e.g., if the user is not authorized to perform an operation. Generating these types of test cases is left for future work.

The response must match the schema defined in the OpenAPI specification for an erroneous HTTP response (i.e. with status code 400-499).

## 5 Evaluation

We applied our test case generation to a few real-world AAS server implementations to assess its feasibility. To generate test cases, we used the official AAS OpenAPI description[5]. However, this description was missing information we needed for generation, including the links between operations and the responses in case of errors. Consequently, we added these accordingly. Furthermore we needed to provide manual samples for in total 11 schemas. Using the enhanced description and our manual samples, our algorithm generated a test database consisting of 183 test cases, of which 54 where positive test cases.

We then used the test database to test three real-world implementations as depicted in Figure 1. The results are shown in Table 2.

Tab. 2: Test results

| Implementation | Source | Tests Passed |
|---|---|---|
| Eclipse BaSyx | https://github.com/eclipse-basyx/basyx-java-examples | 55% |
| FA$^3$ST Service | https://github.com/FraunhoferIOSB/FAAAST-Service | 47% |
| AASX Server | https://github.com/admin-shell-io/aasx-server | 0% |

The *AASX Server* implementation does not follow the AAS REST API but implements a proprietary interface. This is the reason, that all tests fail.

[5] https://app.swaggerhub.com/apis/Plattform_i40/Entire-Interface-Collection

Most of the failing tests of *Eclipse Basyx* and *FA³ST* implementations are caused by mismatching error codes in negative test cases. As we had to add these to the OpenAPI description manually, most of them do not match with the actual return code.

Additionally, some tests are not independent as explained in Section 4.4. For example, when fetching the list of SubmodelElements fails, all subsequent requests using the `idShort` will fail, too.

The test results show, that current implementations are still in early stages. Our approach is able to generate test cases which reliably reveals incompliances in those implementations.

## 6 Conclusion and Outlook

In this work, we presented an approach for the automated generation of a test database for the verification of AAS server implementations. As a basis we used the AAS OpenAPI description.

As related work already indicated, proper modeling of dependencies between REST operations is crucial for meaningful test case generation. So we integrated this into our algorithm. However, the official AAS OpenAPI description is currently missing this information, so that we had to add it in form of links. Additionally, the AAS OpenAPI description is missing responses in case of errors. Since negative test cases are as important as positive ones, we needed to add these responses, too.

Another important aspect is the generation of samples for JSON schemas. Usually, REST APIs only use rather flat JSON objects, so that test case generation methods do not focus on this part. For our use case, the AAS, this is more difficult because of the complex meta model of the AAS. This is why, we had to add manually created samples, especially for the `AssetAdmininistrationShell` object itself. Related work like [OK22] could be leveraged to generate these automatically as well.

Finally, the AAS REST API allows for parameters modifying the response format. These can currently not be expressed using the OpenAPI description. Other aspects of REST APIs like authorization are left for future work, too.

## Bibliography


[AGP18] Atlidakis, Vaggelis; Godefroid, Patrice; Polishchuk, Marina: Rest-Ler: Automatic Intelligent Rest Api Fuzzing. arXiv preprint arXiv:1806.09739, 2018.

[Ar17] Arcuri, Andrea: RESTful API Automated Test Case Generation. In: 2017 IEEE International Conference on Software Quality, Reliability and Security (QRS). IEEE, pp. 9–20, 2017.

[Bu20] Bucaille, Steven; Cánovas Izquierdo, Javier Luis; Ed-Douibi, Hamza; Cabot, Jordi: An Openapi-Based Testing Framework to Monitor Non-Functional Properties of Rest Apis. In: International Conference on Web Engineering. Springer, pp. 533–537, 2020.

[FR14] Fielding, Roy T.; Reschke, Julian: Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. RFC 7231, June 2014.

[FR22a] Fielding, Roy T.; Reschke, Julian: HTTP Semantics. RFC 9110, June 2022.

[FR22b] Fielding, Roy T.; Reschke, Julian: HTTP/1.1. RFC 9112, June 2022.

[HDD22] Hatfield-Dodds, Zac; Dygalo, Dmitry: Deriving Semantics-Aware Fuzzers from Web API Schemas. In: 2022 IEEE/ACM 44th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion). IEEE, pp. 345–346, 2022.

[KČS20] Karlsson, Stefan; Čaušević, Adnan; Sundmark, Daniel: QuickREST: Property-based test generation of OpenAPI-described RESTful APIs. In: 2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST). IEEE, pp. 131–141, 2020.

[MHC19] MacIver, David; Hatfield-Dodds, Zac; Contributors, Many: Hypothesis: A New Approach to Property-Based Testing. Journal of Open Source Software, 4(43):1891, November 2019.

[Ni99] Nielsen, Henrik; Mogul, Jeffrey; Masinter, Larry M; Fielding, Roy T.; Gettys, Jim; Leach, Paul J.; Berners-Lee, Tim: Hypertext Transfer Protocol – HTTP/1.1. RFC 2616, June 1999.

[OK22] Otto, Björn; Kleinert, Tobias: A Flow Graph Based Approach for Controlled Generation of AAS Digital Twin Instances for the Verification of Compliance Check Tools. Submitted for publication, 48th Annual Conference of the IEEE Industrial Electronics Society, 2022.

[Op21] OpenAPI Specification v3.1.0. https://spec.openapis.org/oas/v3.1.0.html.

[Pl20] Plattform Industrie 4.0: Details of the Asset Administration Shell - Part 1 - The exchange of informationen between partners in the value chain of Industrie 4.0 (Version 3.0RC01). Bundesministerium für Wirtschaft und Energie, 2020.

[Pl21] Plattform Industrie 4.0: Details of the Asset Administration Shell - Part 2 - Interoperability at Runtime – Exchanging Information via Application Programming Interfaces 1.0RC02). Bundesministerium für Wirtschaft und Energie, 2021.

[VDC20] Viglianisi, Emanuele; Dallago, Michael; Ceccato, Mariano: RESTTESTGEN: Automated Black-Box Testing of RESTful APIs. In: 2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST). IEEE, pp. 142–152, 2020.

# An Asset Administration Shell Version Control to Enforce Integrity Protection

Andre Bröring[1], Marco Ehrlich[2], Lukasz Wisniewski[3], Henning Trsek[4], Stefan Heiss[5]

**Abstract:** Integrity as one security objective is an important aspect for the adoption of the Asset Administration Shell (AAS) as a data exchange format and data source for data-driven services. Until now, the AAS offers an access control solution as a preventive integrity measure. Nevertheless, preventive methods lack detecting integrity violations due to random errors, accidental modifications, or malicious attacks from access permitted endpoints. This work proposes a concept to complement the access control with a detective integrity measure by creating a traceable version history for the AAS data. After the analysis of the state of the art of AAS security measures and state of the art version control systems, a typical use case for an AAS version control is described. In a preliminary work, an integrity verification scheme for individual submodels using a Certificates submodel and a Signature submodel is proposed. This work complements that integrity verification scheme with the version history in a Versioning submodel as an extension of the Signature submodel. In the evaluation, the data growth of the version history meta data created by the proposed concept is compared with the data overhead created by the versioning in a typical Git repository as an existing version control system.

**Keywords:** Asset Administration Shell; Digital Twin; Version Control; Security; Integrity

## 1 Introduction

Industrie 4.0 is offering a great potential to increase digitalization in the industries and support new technologies in the area of industrial automation. Upcoming data-driven systems and innovative services are fundamental concepts of Industrie 4.0 and considered as the basis to link virtual and physical processes [WSJ17]. As more data is collected at different stages of an asset life cycle, digital twins are one approach to enable an interoperable data exchange. The Asset Administration Shell (AAS) is a promising implementation for a digital twin in the industrial domain. The standardized structure of the AAS enables industrial organizations to store and exchange asset data in an interoperable way. For example, modifications to an asset can be logged and documented in the AAS. As a result, several companies in the value chain can add data to the AAS and access data from the AAS during the asset life cycle. This data can be used for new use cases, such as an automatic safety and security

[1] Institute Industrial IT - inIT, Campusallee 6, 32657 Lemgo, Germany, andre.broering@th-owl.de
[2] Institute Industrial IT - inIT, Campusallee 6, 32657 Lemgo, Germany, marco.ehrlich@th-owl.de
[3] Institute Industrial IT - inIT, Campusallee 6, 32657 Lemgo, Germany, lukasz.wisniewski@th-owl.de
[4] Institute Industrial IT - inIT, Campusallee 6, 32657 Lemgo, Germany, henning.trsek@th-owl.de
[5] Technische Hochschule Ostwestfalen-Lippe, Campusallee 12, 32657 Lemgo, Germany, stefan.heiss@th-owl.de

assessment [Eh20], the training of new Artificial Intelligence (AI) models, or to ensure the quality of products [Pl18]. In order to ensure reliable results of these use cases, the security of the AAS data is of upmost importance.

The three general security objectives are availability, integrity, and confidentiality. For the AAS up to the current development status, the confidentiality is partly realized by limited read permissions in a fine grained access control solution and encryption of whole AASX packages [Pl20a]. The availability depends on the availability of the infrastructure providing the AAS. Integrity is the remaining security objective, that is essential as it impacts the availability as well, for example by outages due to manipulated data [Pl18]. Until now, the integrity of the AAS is considered by adding digital signatures to AASX packages [Pl20a], by a limited write access via the AAS API [Pl20a], or by using Distributed Ledger Technology (DLT) as a tamper-proof storage [Br21]. For the AAS version control, there is a concept that uses a Git-based version control system [Re20]. This Git-based version control system as well as the DLT-based storage do not store all the data inside the AAS itself. This results in dependencies to data outside the AAS, contradicting the concept of the AAS as the central data source for all asset data.

As a result, AAS-based measures to guarantee the integrity of the AAS data at rest from the threats of random errors, accidental modifications, and targeted manipulations [Pl18] should be implemented. In a preceding work, an integrity verification scheme is proposed to enable an integrity verification of individual submodel versions using a Signature submodel [Br22]. However, a reconstruction of old versions is not possible with the proposed concept. Therefore, this work complements the integrity verification scheme with an extension of the Signature submodel with a version history in order to restore and verify past versions of the AAS. Similar to the integrity verification scheme, the meta data for versioning of single submodels is stored inside the AAS to keep the AAS as the central source for the asset data, including the version history.

The remaining structure of this paper is as follows: Section 2 gives a short overview of the AAS and its current security features, as well as a summary of general version control systems and the AAS integrity verification scheme as the basis for this concept. Based on an example use case in Section 3, the concept for the AAS version control is proposed in Section 4. The evaluation in Section 5 analyses the data overhead of the concept in comparison with a creation of a version history using Git. Section 6 summarizes the results and identifies the future work in this domain.

## 2 State of the Art

### 2.1 The Asset Administration Shell

The Asset Administration Shell (AAS) as the industrial implementation of a digital twin will be the future repository containing all life cycle data about industrial assets, such as a

hardware component, machine, or software, in a standardized structure. An AAS contains submodels with the specific asset data. A submodel consists of submodel elements, such as property elements to store single values, reference elements to enable links to other objects, or Submodel Element Collections (SMCs) as a list with more submodel elements to enable a hierarchical tree structure [Pl20a]. These submodels can be based on standardized submodel templates that support the interoperability and a common understanding of the submodel content.

The submodels for a single asset can all be stored in one AAS, or divided into multiple AASs referring to each other. These multiple AASs for one asset can be stored in different locations, for example in embedded storage on the asset itself, in edge storage, or in cloud storages to make it accessible from the inside and outside a company network. All AASs can also have different communication capabilities according to a classification into three types. The first type, the passive AAS, respectively AAS as a file, stores the asset data in a file, such as JSON- or XML-file, that can optionally be embedded in an AASX-file, an AAS package file format similar to ZIP files. Passive AASs can be created, for example, with the AASX Package Explorer[6], an open-source implementation with a graphic user interface, and exchanged, for example, with a Secure Download Service [Pl20b]. The re-active AAS as the second type can be accessed via an API that is described in a technology-neutral way [Pl21b]. The AASX Server[7] is one open-source implementation to provide the AAS API. Via the API, the whole AAS, individual submodels, individual submodel elements, or only the values of the submodel elements can be created, read, updated, and deleted individually. The HTTP/REST interface already includes an access control with settings for read and write restrictions on an AAS, submodel, and submodel element level. The third type, the proactive AAS, autonomously and pro-actively interacts with other AASs and Industrie 4.0 components using Industrie 4.0 language with a defined grammar for an interoperable communication and understanding [Pl21a]. Due to the limited amount of concrete concepts and implementations available for this third type of AASs, it is out of the scope of this work and can be added later as soon as the characteristics are described in more detail.

## 2.2 The Asset Administration Shell Integrity

Until now, the integrity of AASs was considered with digital signatures appended to whole AAS packages and a limited write access for the AAS API [Pl20a]. Nevertheless, integrity can be reached by prevention and detection [HC21]. Access control is a preventive action that reduces the chance of manipulations via the API. It does not provide a possibility to check the data for random errors, as well as accidental manipulations and targeted attacks via permitted endpoints. Furthermore, it does not enable a version control to trace the potential errors and manipulations.

[6] https://github.com/admin-shell-io/aasx-package-explorer
[7] https://github.com/admin-shell-io/aasx-server

Another concept for an AAS versioning including security features for the integrity, confidentiality, and availability of AASs is proposed in [Re20]. It uses a distributed file system, a Git-based version control, and a blockchain-based logging of modifications of the AAS. The distributed storage and Git-based version control enables modifications and enrichments of AASs in collaboration with other organizations, as well as the creation of a version history of the AAS [Re20]. In that concept, the versioning and integrity meta data is not stored inside the AAS itself. Instead, this data is stored in the Git repository meta data creating dependencies on external data sources outside the AAS.

### 2.3 Version Control Systems

Many version control systems are optimized for evolving objects, such as software code during the development. Those systems enable a synchronization and collaboration between the developers, a management of versions with changes, additions, and deletion of features, as well as a restoration of past well-defined states [ZND18]. Software code usually consists of text-based files, similar to the JSON- and XML-representation of the AAS. In addition, other features of the version control systems, such as the collaboration along the AAS life cycle, are useful for the AAS. Therefore, software version control systems are analysed as the basis for an AAS versioning.

In Centralized Version Control Systems (CVCSs), such as Subversion (SVN), there is a central server that saves the full version history. The clients are connected to that server and only have the latest version of a file in their local working copy. After making changes, a client can store the file again on the server to share the changes with other developers updating their working copies [ZND18]. For Distributed Version Control Systems (DVCSs), such as Git, there is no central server with a repository needed. All local repositories contain the entire history of files. As a result, the clients can work offline and store changes in a local version history. The exchange of the versions is then possible in peer-to-peer communications between the local repositories [ZND18]. However, without a central server, there is no single source of truth and central backup, so that every local repository has to create an own backup. In addition, DVCSs are the less intuitive for the users, as, for example, hashes are used for versions instead of version numbers [KU11]. However, the usage of hashes for versions enables a unique identification of these versions and provides integrity, as a manipulation of the version would lead to a new hash.

In the background of the version control systems, snapshots with whole copies of the changed files can be stored, or change sets only storing the differences between two versions independently of other changes. For the snapshots, two versions can be compared to identify the differences. With change sets, the user has to generate a version from a baseline and the following changes [KU11]. Another distinguishing feature is the handling of parallel changes. The newer applications of version control systems mainly use the approach to merge two versions into one. Another option is to lock files that are currently edited in order to prevent other user from changing the same file [KU11].

The concept developed in this work uses the advantages of existing version control systems. The AAS version control should enable a distributed versioning of AASs along the life cycle without central servers. The resulting versions should be able to be merged.

### 2.4 Underlying Integrity Verification Scheme

An integrity verification scheme for single submodels is presented in [Br22] using a **Signature** submodel to store the integrity verification meta data. This complements the preventive access control with a detective integrity measure. The current state of the Signature submodel contains hashes and digital signatures for the integrity verifications of single submodel versions. The Signature submodel equals the submodel shown in Fig. 1, but without the submodel elements marked with **NEW**. In short, it contains one SMC for each submodel that is signed, such as **SecurityAssessment**. In these SMCs, there is a so called Hash-SMC for each version that can be identified with the **idShort** that is assembled from the algorithm (e.g. "SHA_256") and the hash created from the corresponding canonical JSON representation of the submodel. In addition, there is a descriptive commit message and a timestamp. To ensure the integrity, a hash (**ChainedHash**) generated from the submodel-hash, **Timestamp**, **CommitMessage**, and hash of the previous version (**ChainedHashPrevious**) are stored for each version. The chaining of hashes generates an integrity protection for past versions as well, and creates a unique identification for each version. Optionally, one or more signatures (e.g. **Signature 1**) can be attached to each version in order to create a link between the submodel version and a user identity provided by a public key certificate. The public key certificates are stored in the **Certificates** submodel as the second proposed submodel in [Br22].

The core features of the concept are the unique identification of versions by their hashes, the chaining of hashes to ensure the integrity of past versions as well, and the possible integrity verification on a submodel level. However, a reconstruction of old versions for the verification of the integrity of the past version is not possible with current meta data stored in the Signature submodel. Therefore, this work extends that submodel with version history meta data.

## 3 Use Case

As an example use case, the storage of Safety & Security relevant data for an automatic security assessment of modular production systems [Eh20] is described. Therefore, an asset operator wants to store safety and security relevant data about an asset inside the AAS. The data should be accessed via the API of a re-active AAS and be updated with every modification of the safety and security characteristics and conditions of the asset. The integrity and authenticity of the data is important to enable correct assessment results based on the data. Modifications should be version-controlled to reconstruct past states and to

trace the changes. Malicious modifications from unauthorized users affecting the safety and security of a machine should be detected.

The described use case addresses the security assessment of a production system based on data stored in a SecurityAssessment submodel inside an AAS [Eh22]. The AAS version control proposed in this work addresses the integrity of the data inside the SecurityAssessment submodel of the AAS as the basis for a reliable security assessment of the production system.

## 4 Concept

In general, the structure of the Signature submodel from [Br22] is adopted and extended with the version history data. As the version history is the new focus of the proposed submodel, the Signature submodel is renamed into **Versioning**. The signatures are additional features attached to the version history, as these are used to enable an integrity verification of the versions. In order to visualize the concept, the AASX Package Explorer is used to create a prototypical draft of the proposed Versioning submodel and fill it manually with example values, as shown in Fig. 1. As an example, a version history for the **SecurityAssessment** submodel is created containing the Safety & Security data as described in Section 3.

For the creation of the version history, the structure of the Signature submodel is completely adopted and extended with the **DiffAdd**- and **DiffRemove**-SMC as highlighted with **NEW** in Fig. 1. These two SMCs contain the change set between two versions. Storing the change set avoids complete copies of a submodel for each version in order to save data space. Therefore, the reverse delta between two versions is used. The versioned submodel, such as the SecurityAssessment submodel in this example, always contains the newest version of the submodel data. The Versioning submodel contains a copy of all submodel elements removed in this version in the DiffRemove-SMC and a copy of all submodel elements added in this version in the DiffAdd-SMC. If a submodel element changed, there is a copy of that submodel element with the old value in the DiffRemove-SMC and with the new value in the DiffAdd-SMC. A change set always contains the whole tree to the lowest changed submodel element. For example, in Fig. 1, in Version 2 ("**SHA_256_7401...**") **Laser02** containing eight elements is added to **Zones**, as it is stored in the DiffAdd. As another example, in Version 3 ("**SHA_256_76d8...**"), the **Telephone** property changed from "0000" (shown in DiffRemove) to "1111" (shown in DiffAdd). For the Telephone property, all parent SMCs are stored as well.

The change set can be used to restore old versions. In order to reconstruct an old version, a user or software tool can follow the chain of versions containing the change sets. For example, if the previous version should be reconstructed, all submodel elements from the DiffAdd-SMC are removed again and the submodel elements from the DiffRemove-SMC are added. In addition, versions can be merged using the proposed concept. Therefore, two versions with the same ChainedHashPrevious have a similar parent version. As a result, all

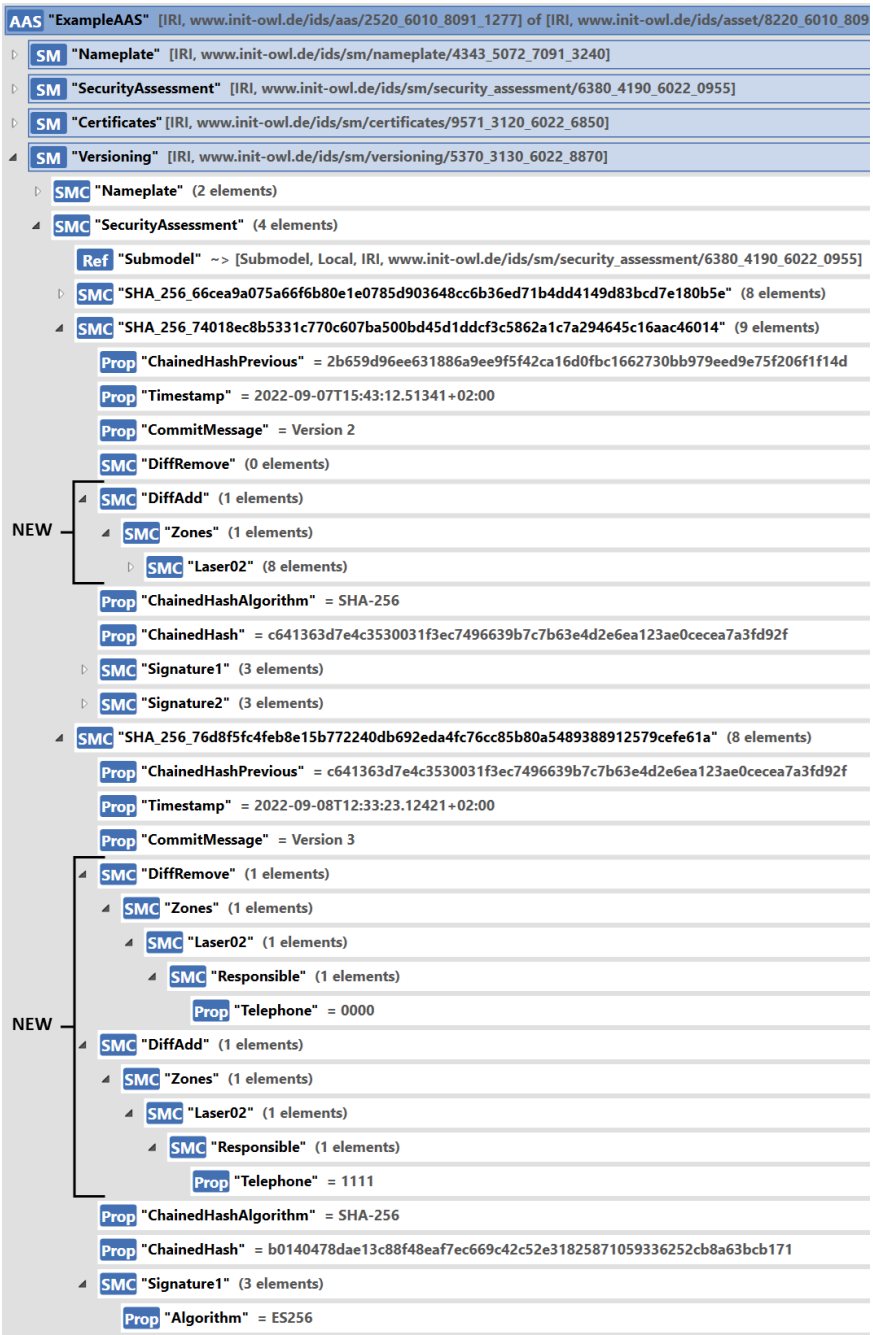



Fig. 1: Versioning Submodel Proposal as an Extension of the Signature Submodel from [Br22]

following change sets can be compared. If there are no changes in the similar submodel elements, the changes can be applied to the target version. In case of conflicting changes in the same submodel element, a conflict resolution is needed.

Besides the two SMCs containing the change set, the properties from the Signature submodel, namely the descriptive CommitMessage, the Timestamp, the ChainedHash, and the ChainedHashPrevious, are stored for each version. This data is similar to the data stored for a version in the Git version control system. The signatures for the versions are still independent of the version history, as Signature-SMCs, such as **Signature1**, can be flexibly added to each Hash-SMC according to the integrity verification scheme in [Br22].

## 5 Evaluation

In the evaluation, the needed memory for the version meta data as a data overhead is analysed and compared to a versioning in a Git repository. In this evaluation, the JSON representation of the AAS is used. All submodel elements refer to an example **ConceptDescription** in the **semanticId**. An example of the attributes of such a property is shown in List. 1:

```
1  { "value": "2
       b659d96ee631886a9ee9f5f42ca16d0fbc1662730bb979eed9e75f206f1f14d",
2    "semanticId": {
3      "keys": [ {
4          "type": "ConceptDescription", "local": true,
5          "value": "0173-1#01-xxx000#001", "index": 0, "idType": "IRDI" }
6      ] },
7    "constraints": [],
8    "idShort": "ChainedHash",
9    "category": "PARAMETER",
10   "modelType": { "name": "Property" },
11   "valueType": { "dataObjectType": { "name": "string" } },
12   "kind": "Instance" }
```

List. 1: JSON Representation of the ChainedHash Property

The example property in List. 1 consists of 363 ASCII characters in the JSON format without spaces and line breaks, leading to a size of 363 bytes. In the proposed concept, for every version, a new Hash-SMC is created containing the submodel hash in the idShort and the property elements ChainedHashPrevious, Timestamp, CommitMessage, ChainedHashAlgorithm, and ChainedHash as shown in Fig. 1. The resulting size for such a Hash-SMC and property elements created per version is about 2830 bytes without the change set and signatures. In addition, the size of a Signature-SMC containing one signature is about 1600 bytes. The size of the change set inside the DiffAdd-SMC and DiffRemove-SMC depends on the changed content, as a copy of the changed content is stored in those SMCs.

As an example, in the following evaluation, ten versions for different random changes of an AAS are created in order to determine the evolving data size that is summarized in Tab. 1. The first column shows the version and a short description of the executed modification for the corresponding version. The second column shows the size of the AAS with only the SecurityAssessment submodel, whereas the third column includes the Versioning submodel with the corresponding accumulated versioning meta data. The last column shows the size of a Git repository containing the corresponding JSON representation of the AAS. All AASs are without signatures and the Certificates submodel.

Tab. 1: Sizes of Basic AAS and the Versioning Overhead for Two Concepts

| Version (Modification) | AAS without Versioning | AAS with Versioning | Git Repository |
|---|---|---|---|
| Version 1 (Initial Version) | 14.229 Bytes | 30.482 Bytes | 56.247 Bytes |
| Version 2 (SMC Laser02 added) | 21.610 Bytes | 48.332 Bytes | 75.975 Bytes |
| Version 3 (Prop. Telephone changed) | 21.610 Bytes | 52.980 Bytes | 79.891 Bytes |
| Version 4 (SMC description changed) | 21.662 Bytes | 56.388 Bytes | 83.913 Bytes |
| Version 5 (SMC Laser01 removed) | 14.281 Bytes | 59.520 Bytes | 71.319 Bytes |
| Version 6 (Ref. AccessPoint changed) | 14.281 Bytes | 64.496 Bytes | 74.570 Bytes |
| Version 7 (4 Prop. (Name) changed) | 14.293 Bytes | 73.916 Bytes | 77.882 Bytes |
| Version 8 (Prop. Telephone changed back) | 14.293 Bytes | 78.796 Bytes | 81.142 Bytes |
| Version 9 (SMC in Conduit removed) | 9.900 Bytes | 81.880 Bytes | 74.513 Bytes |
| Version 10 (5 New SMCs in Conduit) | 31.787 Bytes | 128.739 Bytes | 125.661 Bytes |

**AAS Size without Versioning** The basic size for the AAS with only the SecurityAssessment submodel (second column in Tab. 1) in the Initial Version is 14.229 bytes. After adding the Laser02 in Version 2, it has 21.610 bytes. The change of the Telephone property value from “0000” to “1111” in Version 3 does not change the length and, hence, not change the AAS size. If submodel elements are deleted, the AAS gets smaller, such as for Version 9. The changes of Version 2 and Version 3 are also visible in Fig. 1.

**Versioning with Proposed Concept** In the next step, the size of the AAS with the SecurityAssessment and additional Versioning submodel is determined (third column in Tab. 1). As the versioning meta data for the Initial Version contains a copy of all SecurityAssessment submodel elements, it is already more than the double size (30.482 bytes) of the AAS without the Versioning submodel. Then with every version, a new Hash-SMC with about 2.830 bytes and the additional change set is created. The resulting growth of the AAS containing the Versioning submodel is shown in Fig. 2.

**Versioning with Git** As a comparison, the same changes are made and versioned in a Git repository. In order to enable an efficient versioning with Git, the JSON representation is not compressed into one line, but contains line breaks to enable a line-by-line comparison.

Therefore, the basic size of the AAS is bigger. For the determination of the complete Git repository size, the changes are manually executed in the JSON file and a Git commit created for each change. The repository size is determined from the folder containing the AAS and Git repository meta data. The initialization of an empty repository on the used computer system with *Git 2.31.1.windows.1* creates a folder containing 23.918 bytes of Git meta data. After adding the initial version of the AAS JSON, the repository grew by the size of the file to 56.247 Bytes (fourth column in Tab. 1). The continuing growth of the repository is visualized in Fig. 2.

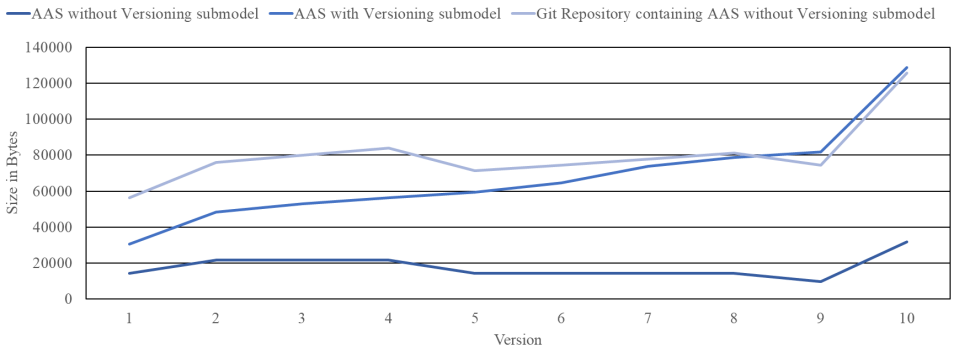



Fig. 2: Data Size for the Three Scenarios From Tab. 1

**Results** The comparison of the two version control concepts shows that the size of the Git repository containing the Initial Version is almost double of the AAS with the proposed Versioning submodel. After that, Fig. 2 shows that the behaviour of the two concepts in terms of the data size is similar if data is added. If data is changed, the Git repository grows slower. If data is deleted (e.g. SMC Laser01 in Version 5), the AAS with only the SecurityAssessment submodel is getting smaller, but the AAS including the proposed Versioning submodel is growing. This is because a copy of the deleted content is stored in the DiffRemove-SMC. In contrast, the size of the Git repository is getting smaller as well, even though a reconstruction of a past bigger version of the AAS is possible. This is due to efficient compression methods for the version history used in Git repositories [CS14]. In addition, Git only compares the lines of the JSON file, whereas the proposed concept stores complete copies of changed submodel elements including all parent SMCs, as shown for the changed Telephone property value in Fig. 1.

In summary, the data overhead for ten versions is similar for the proposed concept and a Git repository as an existing version control system. The initial repository size of the Git repository has the biggest impact on the data size. After that, the size of the Git repository can get smaller if content is deleted from the original submodel. In contrast, the AAS with the proposed Versioning submodel is always growing, even if content is deleted from the original submodel. As a result, the overhead of the Git repository is growing slower than the AAS Versioning submodel proposed in this concept.

## 6 Conclusions and Outlook

In this work, an AAS version control to enforce integrity protection is developed as one component to create future secure AASs. An example Use Case is the storage of Safety & Security data about an asset, such as a production system. A Signature submodel as the basis for the concept is extended with the version change set. This enables the reconstruction of previous submodel versions by the reverse delta between two versions. All the meta data for the version history is stored inside the AAS, keeping the AAS as the central data source for all asset data, including the version history. In addition, a merging of two versions and the signing of single versions is possible to increase the integrity. The evaluation results compare the proposed concept with an existing Git version control system, indicating a slower growth of the Git repository compared to the proposed concept. However, the proposed AAS version control system enables the built-in attachment of signatures to versions and creates a human-readable history of submodel elements.

In future work, long term measurements of the data overhead for more than ten versions will be executed in order to compare the long term growth of the version history in realistic scenarios. Therefore, the current state of the concept can be implemented in existing open source tools, such as the AASX Package Explorer and AASX Server. In future, compression methods for the versioning meta data and the AAS will be developed to reduce the overhead. Afterwards, an adoption for encrypted AAS submodels will be executed to enhance the confidentiality of the AAS content. This may also have an impact on the Git repository size [Sh15]. In addition, the attachment of signatures will be considered in the further evaluation. For different merging scenarios, an automatic conflict resolution will be developed to enable an easy merging of multiple versions of AASs ran by different organizations during the AAS life cycle.

## 7 Acknowledgement

This work was funded by the Ministry of Economic Affairs, Innovation, Digitalization and Energy of the State of North Rhine-Westphalia in the research project “Automatic Evaluation and Monitoring of Safety & Security Properties for Intelligent Technical Systems” (AutoS$^2$).

## Bibliography

[Br21] Bröring, Andre; Belyaev, Alexander; Trsek, Henning; Wisniewski, Lukasz; Diedrich, Christian: Secure Asset Administration Shell exchange with Distributed Ledger Technology. In: Shaping a globally secure Industrie 4.0 Ecosystem. Plattform Industrie 4.0, Berlin, 2021.

[Br22] Bröring, Andre; Ehrlich, Marco; Wisniewski, Lukasz; Trsek, Henning; Heiss, Stefan: Towards an Asset Administration Shell Integrity Verification Scheme. In: 2022 27th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). 2022.

[CS14] Chacon, Scott; Straub, Ben: Pro Git: Version 2.1.338-2-g8a81047, 2022-04-05. Apress, 2014.

[Eh20] Ehrlich, Marco; Benk, Stefan; Harder, Dimitri; Kleen, Philip; Trsek, Henning; Schriegel, Sebastian; Jasperneite, Jürgen: Automatische Bewertung und Überwachung von Safety & Security Eigenschaften – Strukturierung und Ausblick. Jahreskolloquium Kommunikation in der Automation (KommA), 2020.

[Eh22] Ehrlich, Marco; Bröring, Andre; Diedrich, Christian; Jasperneite, Jürgen: Towards Automated Risk Assessments for Modular Industrial Automation and Control Systems: State of the Art Survey and Information Model Proposal. In: 17. Fachtagung EKA – Entwurf komplexer Automatisierungssysteme, Magdeburg, Germany. 2022.

[HC21] Harley, Kelsey; Cooper, Rodney: Information Integrity: Are We There Yet? ACM Computing Surveys, 54(2):1–35, 2021.

[KU11] Koc, Ali; Uz Tansel, Abdullah: A Survey of Version Control Systems. ICEME 2011, 2011.

[Pl18] Plattform Industrie 4.0: Integrity of Data, Systems and Processes as the Core Element of Networking and Digitalization: Discussion Paper. Federal Ministry for Economic Affairs and Energy, Berlin, April 2018.

[Pl20a] Plattform Industrie 4.0: Details of the Asset Administration Shell: Part 1 - The exchange of information between partners in the value chain of Industrie 4.0 (Version 3.0RC01): Specification. Federal Ministry for Economic Affairs and Energy, Berlin, November 2020.

[Pl20b] Plattform Industrie 4.0: Secure Download Service: Discussion Paper. Federal Ministry for Economic Affairs and Energy, Berlin, October 2020.

[Pl21a] Plattform Industrie 4.0: Functional View of the Asset Administration Shell in an Industrie 4.0 System Environment: Discussion Paper. Federal Ministry for Economic Affairs and Energy, Berlin, April 2021.

[Pl21b] Plattform Industrie 4.0: Details of the Asset Administration Shell: Part 2 - Interoperability at Runtime – Exchanging Information via Application Programming Interfaces (Version 1.0RC02): Specification. Federal Ministry for Economic Affairs and Energy, Berlin, November 2021.

[Re20] Redeker, Magnus; Volgmann, Sören; Pethig, Florian; Kalhoff, Johannes: Towards Data Sovereignty of Asset Administration Shells across Value Added Chains. In: 2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). IEEE, Piscataway, NJ, 2020.

[Sh15] Shirey, Russell G.; Hopkinson, Kenneth M.; Stewart, Kyle E.; Hodson, Douglas D.; Borghetti, Brett J.: Analysis of Implementations to Secure Git for Use as an Encrypted Distributed Version Control System. In: 2015 48th Hawaii International Conference on System Sciences. IEEE, pp. 5310–5319, 2015.

[WSJ17] Wollschlaeger, Martin; Sauter, Thilo; Jasperneite, Juergen: The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. IEEE Industrial Electronics Magazine, 11(1):17–27, 2017.

[ZND18] Zolkifli, Nazatul Nurlisa; Ngah, Amir; Deraman, Aziz: Version Control System: A Review. 3rd International Conference on Computer Science and Computational Intelligence 2018, 2018.

# Impressum

13. Jahreskolloquium
**Kommunikation in der Automation**
(KommA 2022)

03. November 2022 • Lemgo

OPEN-Book

ISBN: 978-3-9818463-3-1
DOI: https://doi.org/10.25644/a4ws-9a49

Herausgeber:
Jürgen Jasperneite, Lemgo
Ulrich Jumar, Magdeburg

**Institut für industrielle Informationstechnik**
Technische Hochschule Ostwestfalen-Lippe

Campusallee 6
D-32657 Lemgo
Telefon: +49 5261 702-2400
Internet: www.init-owl.de | www.jk-komma.de