Anwendungen

Jingyun Zhao, Birgit Vogel-Heuser, Fandi Hartl*, Josua Höfgen, Felix Ocker, Bernd Vojanec, Timo Markert und André Kraft

Erstellung und Integration von Verwaltungsschalen aus Heterogenen Datenquellen

Creation and Integration of Asset Administration Shells from Heterogeneous Data Sources

https://doi.org/10.1515/auto-2023-0172
Empfangen September 15, 2023; angenommen Januar 15, 2024

Zusammenfassung: Heterogene Datenquellen und inkonsistente Schnittstellen zwischen verschiedenen Ingenieursdisziplinen stellen eine enorme Herausforderung bei der Erstellung einer virtuellen Repräsentation eines realen Produkts, Anlage oder Prozesses dar. Zu dieser digitalen Darstellung wird die Verwaltungsschale (VWS) als ein standardisiertes Konzept vorgeschlagen. Jedoch erfordert die VWS-Erstellung oft erheblichen manuellen Aufwand, um Informationen aus technischen Engineering- und Betriebsdaten zu extrahieren und zu strukturieren. Der folgende Beitrag schlägt einen semi-automatisierten Ansatz zur Extraktion von Engineering-Informationen aus Dokumenten der Formate PDF, STP, XML, AML sowie URDF vor, um anhand dieser Informationen mehrere VWS zu erstellen, die zusammengeführt werden können. Anhand dreier Anwendungsfälle werden die Möglichkeiten und Grenzen des Ansatzes diskutiert.

Schlagwörter: Verwaltungsschale; automatische Informationsextraktion; heterogene Engineering-Daten; Dateiformate; VWS-Erstellung; VWS-Zusammenführung

Abstract: Heterogeneous data sources and inconsistent interfaces between different engineering disciplines pose a tremendous challenge in creating a virtual representation of a real-world product, plant or process. For this digital representation, the Asset Administration Shell (AAS) is proposed as a standardized concept. Unfortunately, the creation of AAS often requires a huge manual effort in extracting and structuring information pieces from diverse engineering and operation data by hand. This work proposes a semi-automatic approach to extract engineering information from documents of the data types of PDF, STP, XML, AML, URDF and a method to execute AAS creation and merging. The possibilities and limitations of the proposed approach are discussed by using the three use cases.

Keywords: Asset Administration Shell; automatic information extraction; heterogeneous engineering data; file format; AAS creation; AAS merging

***Corresponding author: Fandi Hartl**, Lehrstuhl für Automatisierung und Informationssysteme, TUM School of Engineering and Design, Technische Universität München, Garching, Deutschland,
E-mail: fandi.bi@tum.de
Jingyun Zhao, **Josua Höfgen und Felix Ocker**, Lehrstuhl für Automatisierung und Informationssysteme, TUM School of Engineering and Design, Technische Universität München, Garching, Deutschland,
E-mail: jingyun.zhao@tum.de (J. Zhao), josua.hoefgen@tum.de (J. Höfgen), felix.ocker@tum.de (F. Ocker)
Birgit Vogel-Heuser, Lehrstuhl für Automatisierung und Informationssysteme, TUM School of Engineering and Design, Core Member der MDSI und Leiterin der Sector Work der MIRMI, Technische Universität München, Garching, Deutschland, E-mail: vogel-heuser@tum.de
Bernd Vojanec, WITTENSTEIN SE, Corporate Research & Development, Igersheim, Deuschland, E-mail: bernd.vojanec@wittenstein.de
Timo Markert, Resense GmbH, Geschäftsführung, Klingenberg, Deutschland, E-mail: timo.markert@resense.io
André Kraft, BMW AG, Digitalisierung Shopfloor, München, Deutschland, E-mail: andre.kraft@bmw.de

1 Einleitung

Digitale Zwillinge (DZ) bestehen sowohl aus Engineering-Informationen als auch aus Betriebsdaten [1]. Hierbei können die Informationen in verschiedenen Dateiformaten vorliegen und gegebenenfalls inkonsistente Daten enthalten. Die Verwaltung dieser Informationsvielfalt stellt eine Herausforderung bei der Implementierung von DZ dar. Zur standardisierten Beschreibung eines physischen Objekts stützen wir uns auf die Verwaltungsschale (Abk. VWS, engl. Asset Administration Shell, Abk. AAS). Als integrales Element von Industrie 4.0 (I4.0) dient sie der

Open Access. © 2024 the author(s), published by De Gruyter. This work is licensed under the Creative Commons Attribution 4.0 International License.

digitalen Abbildung und Verwaltung von Informationen über Assets (z. B. Bestandteile einer I4.0-Komponente) [2]. Der Zweck der VWS-Struktur besteht darin, die Interoperabilität von Produkt- und Prozessinformationen in der gesamten Wertschöpfungskette zu ermöglichen [2]. Die VWS wird üblicherweise vom Hersteller oder Produzenten erstellt. Kunden können jedoch auch zur Entwicklung und Nutzung von Inhalten der VWS beitragen, insbesondere bei individuellen Produktkonfigurationen. In einem Anwendungsfall könnte beispielsweise ein Getriebehersteller Informationen über einzubauende Schrauben, Stifte, Normen und Toleranzen von seinen Zulieferern in verschiedenen Dokumenten in unterschiedlichen Formaten erhalten. Der Hersteller formalisiert anschließend alle Getriebeinformationen mithilfe des vorgestellten Ansatzes in einer VWS und liefert diese an den Kunden, d. h. an den Maschinen- und Anlagenbauer. Dieser ist anschließend in der Lage, durch einen Import der Getriebe-VWS in die VWS des Hauptprodukts (z. B. einer Maschine) einen umfangreichen DZ zu erstellen. Da der Austausch von Dateien bestimmte Anforderungen erfüllen muss, z. B. in Bezug auf Benutzerfreundlichkeit und Sicherheit, bietet die VWS eine Grundlage für eine bilaterale Vereinbarung über Sicherheitsanforderungen, die für die Übertragung und Wiederverwendung der Dateien erfüllt werden müssen.

Neben den Vorteilen der Anwendung von VWS für die Verwaltung von Assets gibt es jedoch auch einige Herausforderungen, die die Anwendung von VWS in der Praxis erschweren. Nach unserem besten Wissen erstellen führende Unternehmen für mechatronische Komponenten automatisch eine VWS aus ihren digitalen Produktkatalogen [3]. Der Ausgangspunkt für die Nutzung solcher Produktkataloge ist in der Regel die Berücksichtigung der Datenstruktur in unternehmenseigenen Anwendungsfällen und somit in der Regel nicht generalisierbar. Bei der automatischen VWS-Erstellung stellen Aspekte der Generalisierbarkeit eines Ansatzes eine Herausforderung dar (Herausforderung 1, Abk. H1). Die verschiedenen technischen Daten können aus der Planung, dem Entwurf oder der Qualitätskontrolle stammen, und können ferner unterschiedliche Formate haben, z. B. Planungsdokumente in MS Word/PDF, Konstruktionsdaten in einem MCAD/ECAD-Format und Auditdaten in XML. Darüber hinaus werden Betriebsdaten in Echtzeit während der Produktion erstellt, dokumentiert und gemeldet. In der Praxis werden Betriebsdaten häufig in XML-Formaten strukturiert. Die Extraktion von Informationen aus heterogenen Dokumentenformaten sollte in Abhängigkeit vom konkreten Dateityp durchgeführt werden, um daraus VWS zu erstellen. Die verwendeten Mapping-Algorithmen zur Transformation von Dateiformaten in VWS können hierbei in Abhängigkeit vom spezifischen Dateiformat variieren (H2). Die Heterogenität der Datenquellen sowie inkonsistente Schnittstellen zwischen unterschiedlichen Ingenieursdisziplinen stellen die Anforderungen an die Integration dieser vielfältigen Perspektiven auf die Assets dar (H3). Unser Ansatz zur semi-automatisierten VWS-Generierung und -Zusammenführung basiert auf der Nutzung von Engineering-Dokumenten unterschiedlicher Formate wie PDF, STP, XML, AML sowie URDF. Ebenfalls erfolgt die Extraktion von anlagenbezogenen Informationen aus diesen Dokumentenformaten. Die extrahierten Informationen werden anschließend in ECLASS abgebildet, um semantische Interoperabilität zu ermöglichen. Die Inkonsistenzen zwischen den Informationen werden bereinigt, bevor sie in die VWS integriert werden. Das Ziel dieses Ansatzes besteht darin, die Erstellung von VWS zu erleichtern und gleichzeitig eine präzise sowie konsistente Darstellung dieser VWS sicherzustellen. Im Gegensatz zur herkömmlichen Verarbeitung von Daten auf Produktebene liegt der Fokus unseres Ansatzes auf der Ebene der Engineering-Dokumente. In diesem Beitrag wird der Ansatz anhand von drei verschiedenen Anwendungsfällen präsentiert. Diese Demonstration ermöglicht eine detaillierte Vorstellung der Effektivität und Anwendbarkeit des semi-automatisierten Ansatzes in realen industriellen Szenarien.

Der Mehrwert dieser Arbeit liegt neben der Informationsextraktion aus verschiedenen Dateiformaten in VWS mit eindeutigem Mapping in ECLASS und SemanticID insbesondere in der Bewältigung von Inkonsistenzen zwischen den extrahierten Informationen. Unterschiedliche Inkonsistenzen, darunter sowohl syntaktische als auch semantische Inkonsistenzen in den semi-automatisch generierten VWS, werden durch eine Reihe vorgeschlagener Regeln überprüft und bereinigt. Das automatisierte Inkonsistenzmanagement trägt zur effektiven Erstellung von integrierten VWS bei, wodurch potenzielle Fehler im Vergleich zur manuellen VWS-Erstellung minimiert werden.

Der Beitrag ist wie folgt gegliedert. Abschnitt 2 gibt einen Überblick über den aktuellen Forschungsstand zu DZ, VWS und Informationsextraktionsmethoden zur VWS-Erstellung. In Abschnitt 3 wird die Methodik der semi-automatisierten Erstellung von VWS auf Grundlage ausgewählter Dateiformate vorgestellt. Drei Anwendungsfälle und Implementierungsdetails werden in Abschnitt 4 beschrieben, während die Ergebnisse in Abschnitt 5 kritisch diskutiert werden. Der Artikel schließt mit einer Zusammenfassung und einem Ausblick in Abschnitt 6.

2 Der aktuelle Stand der Technik zum Digitalen Zwilling, der Verwaltungsschale und den Informationsextraktionsmethoden

Dieser Abschnitt gibt einen Überblick über die grundlegenden Konzepte des DZ, der VWS und der Informationsextraktionsmethoden, welche für die VWS-Erstellung relevant sind.

2.1 Digitaler Zwilling und Verwaltungsschale

Im Kontext der I4.0 ist der DZ als „dynamische virtuelle Repräsentation eines physischen Objekts oder Systems über seinen Lebenszyklus hinweg“ definiert [4]. Der DZ sollte sämtliche digital verfügbaren Daten, Modelle und Dokumente des umfassenden Produktentwicklungszyklus einschließen. Dies umfasst technische Informationen aus heterogenen Quellen, etwa Entwürfe über Fertigungsdaten oder Wartungsinformationen, die während des ganzen Lebenszyklus eines realen Gegenstückes gesammelt werden. Dadurch können Nutzer des DZs ihre physischen Objekte den gesamten Produktentwicklungszyklus hinweg effizienter und zuverlässiger steuern, manipulieren und optimieren. Es existieren einige DZ-Plattformen, die von verschiedenen Dienstleistern angeboten werden, wie z. B. Eclipse Ditto [5] und Microsoft Azure [6]. Diese Plattformen bieten umfassende Unterstützung durch Tools und kombinierbare Software-Dienste, die auf einer einheitlichen Definition physischer Objekte und ihrer Daten beruhen. Allerdings sind diese Definitionen derzeit proprietär, was die plattformübergreifende Wiederverwendbarkeit und die Integration von Diensten verschiedener Anbieter erschwert. Zusätzlich fehlt es diesen plattformabhängigen Lösungen an Unterstützung für die Wiederverwendung vorhandener Informationen in technischen Modellen. Diese technischen Modelle erfassen nicht nur Daten aus der Mechanik, wie zum Beispiel das CAD-Modell für ein Asset, sondern auch aus der Mechatronik und Softwaretechnik. In der Mechanik sind Bereiche wie Elektrik (z. B. elektrische Komponenten in Elektronikplänen), Pneumatik (z. B. pneumatische Schaltpläne zur Steuerung pneumatischer Systeme) und Prozesstechnik (wie zum Beispiel R&I-Fließbilder für Rohrleitungen sowie weitere Komponenten eines Prozessleitsystems) eingeschlossen. In der Softwaretechnik wird beispielsweise nach IEC 61131 für speicherprogrammierbare Steuerungen (SPS) integriert. Sie bieten eine ganzheitliche Darstellung des Lebenszyklus eines Systems im Automatisierungsdomäne [7]. Der Mangel an Unterstützung für die Wiederverwendung von Informationen kann zu doppelter Arbeit bei der Erstellung und Aktualisierung sowohl der technischen Modelle als auch der Definition der DZ führen. Dies bedeutet, dass Informationen möglicherweise mehrmals manuell eingegeben oder aktualisiert werden müssen, was ineffizient ist und das Risiko von Inkonsistenzen erhöht [8]. Es gibt auch Ansätze zur Erstellung von DZ mithilfe von AutomationML [9]. Dabei liegt der Vorteil darin, Informationen aus unterschiedlichen Domänen zu kombinieren. Allerdings wird AutomationML hauptsächlich in den Entwurfsphasen eines Produktes eingesetzt. Die Herausforderung liegt in der Entwicklung wiederverwendbarer Module und im Umgang mit verschiedenen Varianten und Versionen von Industrieanlagen in AutomationML. Dies wird insbesondere durch semantische Unterschiede zwischen dem Computer Aided Engineering Exchange (CAEX) und anderen Standards wie beispielsweise dem PLCopen XML erschwert. Obwohl AutomationML auf dem strukturierten und generischen Modell CAEX basiert, können Inkompatibilitäten aufgrund unterschiedlicher Interpretationen von Konzepten auf der Modulebene auftreten [4]. Angenommen, ein Ingenieur erstellt ein wiederverwendbares Modul für eine spezifische Steuerungsfunktion in AutomationML mit spezifischen Attributen, Parametern und Verhaltensweisen nach den CAEX-Standards. Bei dem Versuch, dieses Modul in ein System zu integrieren, das den PLCopen XML-Standard verwendet, können semantische Unterschiede in Bezug auf die Steuerungslogik und die Bedeutung bestimmter Parameter auftreten. Diese Unterschiede können zu Unklarheit führen, etwa bei logischen Bedingungen oder der Interpretation von Steuerungsaktionen. Diese erschweren durch die reibungslose Wiederverwendung und Integration wiederverwendbarer Module zwischen verschiedenen Standards in der Automatisierungstechnik [9].

Die VWS ist eine der DZ-Lösungen, mit denen standardisierte digitale Asset Repräsentationen erzeugt und gesteuert werden können. Das VWS-Framework schafft Schnittstellen zwischen der physischen und virtuellen Wertschöpfungskette. Jede physische Komponente der Wertschöpfungskette sollte ihre eigene VWS haben. Die VWS definiert sich als standardisierte digitale Repräsentation eines Assets. Die Struktur der VWS der Version 3.0RC02 ist in Abbildung 1 dargestellt [2], [10]. Diese spezifiziert beispielsweise die globale eindeutige ID (Identifikation), Versionsinformationen (Administration) sowie eine obligatorische Referenz zu den Informationen des Asset Identifiers (Asset-Information). Die Details über das jeweilige Asset sind in Submodels enthalten. Jedes Submodel stellt einen bestimmten Aspekt

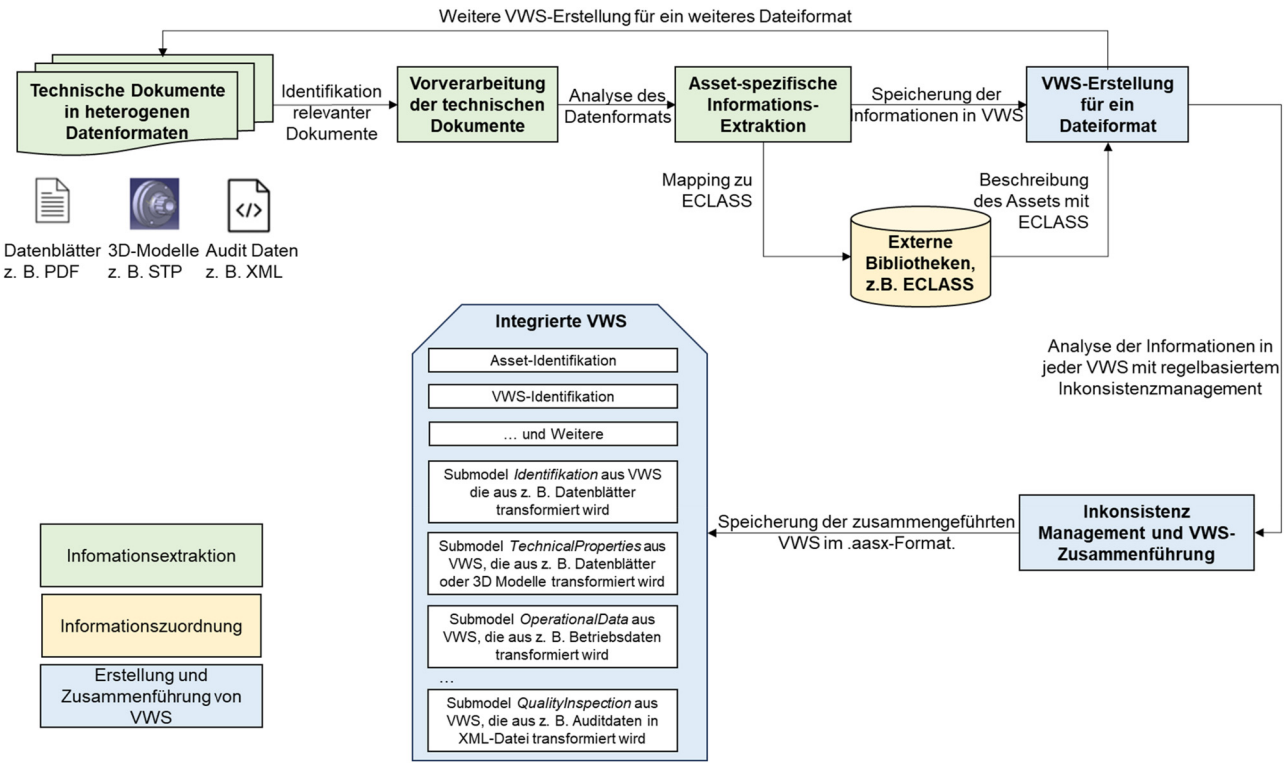

Abbildung 1: Übersicht des Konzepts zur semi-automatisierten VWS-Erstellung aus heterogenen technischen Daten.

von einem Asset in Form hierarchisch organisierter Objekte vom Typ SubmodelElement dar. Jedes SubmodelElement modelliert die Asset-spezifischen Charakteristika, wie Merkmale, Eigenschaften und Fähigkeiten der betrachteten physischen Komponente. Durch die Zuweisung einer semantischen ID zu jedem *Submodel* und *SubmodelElement* wird die semantische Beschreibung der enthaltenen Informationen eindeutig festgelegt [2]. Semantische IDs können hierfür, u. a., aus dem ECLASS Standard [11] oder dem IEC CDD (Common Data Dictionary) [12] entnommen werden. ECLASS verwendet, wie von der Plattform I4.0 empfohlen, für jedes Element global eindeutige Identifikatoren, sogenannte *International Registration Data Identifier* (IRDI) oder *Uniform Resource Identifier, Locator, Name* (URI, URL, URN), um sicherzustellen, dass die Semantik „eindeutig, sprachneutral, maschinenlesbar und branchenunabhängig“ ist [13]. Auf Grundlage von standardisierten *Submodels* gewährleistet die VWS den interoperablen Informationsaustausch von Asset-Informationen und Asset-Diensten zwischen Anwendungen und allen Stakeholdern [14]. Im Kontext der VWS werden die ENUM-Werte des VWS-Attributs „AssetKind“, wie „Template“ und „Kind“, verwendet. Diese Attribute spielen eine entscheidende Rolle bei der Strukturierung und Definition von Assets innerhalb des VWS-Frameworks. Das „Template“ kennzeichnet einen Prototyp zur Erstellung von Instanzen von Assets mit gemeinsamen Merkmalen und erleichtert somit die Schaffung konsistenter und wiederverwendbarer Komponenten. Das „Kind“-Asset ist eine konkrete, eindeutig identifizierbare Komponente eines bestimmten Templates [2].

Die Auswahl der in VWS-Anwendungen integrierten Daten variiert in der Praxis je nach Art der Anwendung. Die selektive Datenintegration sollte eine effiziente Nutzung der VWS-Ressourcen ermöglichen [1]. Das bedeutet, dass die erforderliche Datenmenge, -granularität und -typ für den Austausch anhand der Art der Anwendung definiert werden. Ein Beispiel hierfür ist der Austausch von Engineering-Informationen zwischen Partnern in der Wertschöpfungskette, also Bedienern in einer Produktionslinie, einem Maschinenintegrator und Lieferanten, für „Design-to-Order“-Produkte [15]. Der Bediener definiert die Anforderungen einer Anlage, also eine Teilsystemspezifikation, und teilt sie dem Integrator mit, um die tatsächliche Anlage auszuwählen. Es ist nicht zwingend erforderlich, alle Daten in die VWS zu übertragen. Beispielsweise enthält diese Teilsystemspezifikation nur eine grobe Struktur der Anlage,

wie Funktion und Ort der Anlage. Nicht notwendige Datenquellen wie der gesamte Programmierungscode können dann lediglich referenziert werden. Integratoren erstellen dann zusammengesetzte Systeme auf der Grundlage von Marktforschung. Anforderungen können teilweise durch bestehende O&G (Oil and Gas)-Standards formalisiert werden, beispielsweise kann die grobe Anlagenstruktur mit einer ISA-95/ISA-88-Hierarchie abgebildet werden. Die Daten können auch mithilfe von IDTA standardisierte Submodel-Templates für den Austausch zwischen Partnern bereitstellen [16]. In diesem Beispiel können die Submodel-Templates „Technical Data“ und „DEXPI“ von IDTA genutzt werden. Die Submodel-Templates „Technical Data“ enthält Assets, die von einem Hersteller von Industrieanlagen mit technischen Daten (Eigenschaften) auf den Markt gebracht werden. Die Submodel-Templates „DEXPI“ enthält eine Vielzahl von (teil-)standardisierten Artefakten wie Diagrammen, Zeichnungen, Tabellen, Zertifikaten und anderen Dokumenten aus verschiedenen Disziplinen. Die Verantwortung für die Erstellung der VWS sollte bei den beteiligten Partnern liegen [15]. Wenn beispielsweise der Hersteller VWS für Kontaktinformationen und CAD-Informationen seines Produkts bereitstellt, sollte er für die Auswahl der auszutauschenden Daten, die Aktualisierung dieser Daten und die Bereinigung der Daten verantwortlich sein.

2.2 Informationsextraktion für die Erstellung von Verwaltungsschalen

Im Bereich der Produktionssysteme, der sich u. a. aus Mechanik, Elektronik und Softwaretechnik zusammensetzt, liegen die Informationen über den gesamten Lebenszyklus hinweg in heterogenen Dokumenten vor [7]. In der Regel werden diese Dokumente von Ingenieuren verschiedener Disziplinen erstellt, die unterschiedliche Modellierungssprachen verwenden. Die semi-automatisierte Extraktion von Daten aus heterogenen Datenquellen sowie die strukturierte Speicherung in einer VWS ermöglicht einen einfacheren und schnelleren Zugriff auf diese Informationen und die Entwicklung von disziplinübergreifenden DZ. Ein Zusammenhang zwischen diesen disziplinspezifischen Sprachen und der VWS kann auf der Grundlage der Gemeinsamkeiten und Unterschiede ihrer Metamodellen erfolgen. Die daraus resultierende, semi-automatisierte Generierung der DZ aus den vorhandenen heterogenen Daten ermöglicht somit erhebliche Zeit- und Kosteneinsparungen [8].

Abhängig vom Grad der maschinellen Interpretierbarkeit, d. h., wie gut die Daten von Maschinen verstanden werden können, werden sich technische Daten in verschiedene Kategorien unterteilen. Hierzu gehören unstrukturierte Daten wie ausformulierte Texte in PDF-Dateien, strukturierte Daten wie die Daten gemäß einem strengen Schema in XML-basierten Formaten, und formale Daten wie die Daten in Ontologien [17]. Unstrukturierte XML-Daten mit Einbettung von unstrukturiertem Text in Elementen werden in diesem Beitrag nicht behandelt. Verschiedene Informationsextraktionstechniken, z. B. regelbasierte oder auf Maschinellem Lernen (ML) basierende Ansätze [18], können verwendet werden, um technische Daten von verschiedenen Dokumentenformaten in einer VWS abzubilden. In [19] wird ein automatischer Mapping-Algorithmus von AML, einer XML-basierten, objektorientierten Datenmodellierungssprache, auf eine VWS vorgeschlagen. Allerdings werden nur die AML-Elemente *InternalElement* der AML *InstanceHierarchy* in der VWS abgebildet. Die Ergänzung von ECLASS ermöglicht durch standardisierte Beschreibungen einen anwendungsübergreifenden Austausch technischer Daten für ein Produkt [20]. Die Interaktion mit semantisch heterogenen Daten wird durch die Abbildung von unbekanntem oder unklarem Vokabular in einer Zielbibliothek realisiert. Die Zielbibliothek ist ein standardisiertes Vokabularcorpus. Sie kann zum Beispiel das ISO-DistilBERT-SE-Modell sein, das verschiedene Definitionen und Bezeichnungen für *SubmodelElements* in der Identifikations- und Entwurfsdomäne von Pumpen mit ISO-Standards enthält [21]. Das Hinzufügen von Semantischen IDs zu *Submodels* und *SubmodelElements* in der VWS, die mit der Zielbibliothek verknüpft sind, unterstützt die eindeutige Beschreibung von Parametern und Funktionen der Pumpen. Diese Zuordnungen können mit Hilfe von *Natural Language Processing* (NLP, dt. natürliche Sprachverarbeitung) [22] und anderen Techniken des MLs, wie z. B. der Bilderkennung [23] und dem Transferlernen [24], durchgeführt werden. Darüber hinaus stellt [25] einen Ansatz vor, um Eigenschaftsinformationen eines Assets aus Tabellen in PDF-Dateien zu extrahieren und in vordefinierten VWS-Vorlagen zur Instanziierung zu speichern. Die verwendete, vordefinierte VWS-Vorlage wird jedoch manuell erstellt. In [26] werden Informationen aus mehrspaltigen Excel-Tabellen automatisch extrahiert und in VWS gespeichert. Beide Methoden zur Extraktion von Tabelleninformationen [25], [26] berücksichtigen die semantische Beschreibung des Assets nicht, welche beispielsweise durch das Mapping von Eigenschaftsnamen in ECLASS gelöst werden kann. Außerdem erfordert die VWS-Erstellung von PDF-Tabellen im Vergleich zu Excel-Tabellen einen zusätzlichen Schritt in der Tabellenerkennung: die Reihen der Tabelle sind vor der Informationsextraktion zu identifizieren. Formale Ontologien können ebenfalls als Grundlage für die Darstellung von Anlageninformationen im technischen Prozess der

VWS-Erstellung dienen. Jedoch kommen Ontologien nicht so häufig zum Einsatz, da sie auf der Prädikatenlogik basieren und die existierenden Modelle mit erheblichem Aufwand entsprechend angepasst werden müssen, um sie in die Ontologie zur Wissensrepräsentation in einem bestimmten Bereich zu integrieren. Hierzu gibt es verschiedene Ansätze und Arbeiten, wie beispielsweise das Verwenden von Industriestandards als Entwurfsmuster für Ontologien [27] oder der Versuch, OPC UA in eine Ontologie zu integrieren [28]. Diese Ansätze sind jedoch bisher noch nicht weitreichend in der industriellen Welt etabliert. Es ist oft schwierig für Ingenieure, die mit den zugrunde liegenden Technologien wie beispielsweise *semantischer Webtechnologie (SWT)* nicht vertraut sind, Ontologie zu erstellen und zu verwenden [17]. Dies schränkt die Abbildung von Ontologien in die VWS ein.

Die bisherige Forschung hat sich hauptsächlich auf die Überführung von Daten in VWS für bestimmte Formate konzentriert. Beispielsweise wurden Tabellen in PDF/Excel sowie InternalElements im AML-Format in die VWS umgewandelt. Die Berücksichtigung der Datenüberführung in VWS unter Einbeziehung anderer Dateiformate wurde bisher nicht in Betracht gezogen. Die systematische Erforschung des Inkonsistenzmanagements zwischen den extrahierten Informationen und den erstellten VWS steht noch aus. In Bezug auf die Datenformate können sich diese Inkonsistenzen als sowohl syntaktische als auch semantische Unstimmigkeiten unterscheiden [29]. Syntaktische Inkonsistenzen repräsentieren Verstöße gegen die für ein Modell definierten Anforderungen und Randbedingungen, während semantische Inkonsistenzen Widersprüche bei der Übertragung von Daten in ein Modell bedeuten.

3 Ansatz für die Semi-automatisierte Erstellung von Verwaltungsschalen

In diesem Abschnitt wird unser Ansatz zur semi-automatisierten Erstellung von VWS im Kontext der I4.0 beschrieben. Zunächst wird ein Überblick über den Ansatz gegeben. Anschließend werden die einzelnen Komponenten, d. h. Informationsextraktion, Informationszuordnung, Erstellung und Zusammenführung der VWS erläutert.

3.1 Überblick über den Ansatz

Abbildung 1 illustriert den entwickelten semi-automatisierten Ansatz zur Erstellung von VWS aus heterogenen Dokumenten. Unterschiedliche Dokumentenformate enthalten unterschiedliche Informationen. PDF-Dateien bestehen beispielsweise aus Tabellen mit Informationen zur Identifikation des Herstellers, Kontaktdaten und den technischen Eigenschaften eines Produkts. STP-Dateien dienen der Speicherung von 3D-Modellen. Im Fall von XML-Formaten werden z. B. Audit-Daten von Produkten protokolliert. Nach der Vorverarbeitung der Dokumente werden die assetbezogenen Informationen mit den entsprechenden verschiedenen Techniken extrahiert (siehe Abschnitt 3.2). Anschließend werden die extrahierten Informationen auf ECLASS abgebildet, um die semantische Interoperabilität zu ermöglichen (siehe Abschnitt 3.2). Aus jedem Dokument wird eine VWS generiert. Bevor diese VWS zu einer integrierten VWS zusammengeführt werden, werden zunächst Inkonsistenzen zwischen ihnen beseitigt (siehe Abschnitt 3.3).

3.2 Informationsextraktion und Information-Mapping

Nicht alle Informationen aus den zahlreichen technischen Dokumenten werden für die Erstellung der VWS verwendet. Die falsch erfassten Daten sollten gelöscht werden, während sich wiederholende Daten wie technische Eigenschaften für ein Asset oder andere Merkmale eines Objektes, die in verschiedenen Dokumenten erfasst wurden, aussortiert werden. Darüber hinaus sollten für die DZ-Erstellung irrelevante Dateien, wie z. B. leere Dokumente ausgeschlossen werden. In der Informationsvorverarbeitung werden Dokumente mithilfe der *Term Frequency-Inverse Document Frequency (TF-IDF)*-Methode [30] analysiert. Um relevante Dokumente für ein bestimmtes Asset zu identifizieren, werden innerhalb eines Dokumentenarchivs, das alle Dokumente aller Komponenten einer Anlage enthält, Schlüsselbegriffe wie beispielsweise der Asset-Name anhand der Berechnung des *TF-IDF*-Scores bewertet. Der höchste *TF-IDF*-Score repräsentiert die relevanteste Dokumentation für das Asset.

Um relevante Informationen aus verschiedenen Dateiformaten zu extrahieren, werden unterschiedliche Ansätze verwendet. Für die Extraktion von Informationen aus einem PDF-Datenblatt, das Tabellen enthält, wird eine

Zeilen-Spalten-Strukturanalyse verwendet, um Daten aus Tabellen zu erhalten. Bei PDF-Dateien werden Tabellen mit komplexerer Struktur, wie in Tabellen eingebettete Tabellen, bisher nicht berücksichtigt. Die technischen Eigenschaftsnamen eines Assets auf den Klassifikationsstandard von ECLASS abzubilden, ist eine der Herausforderungen der Informationsextraktion bei zweispaltigen Tabellen. Dabei sollen die Werte und Wertebereiche automatisiert erkannt und anhand von ECLASS vorgeschlagenen Einheiten umgerechnet werden. Bei STP-Datenformaten werden die Informationen geparst und musterbasierte Methoden zur Informationsextraktion verwendet. XML-basierte Formate wie URDF und AML sind im Allgemeinen ebenfalls strukturiert, haben aber andere Modellierungsprinzipien. Daher wird eine Modelltransformation zwischen diesen Formaten und der VWS durchgeführt. Im Rahmen dieser Arbeit haben wir gezielte Untersuchungen der Dateiformate durchgeführt, darunter PDF, STP, XML, AML und URDF, unter Verwendung der in Tabelle 1 aufgeführten Techniken zur Informationsextraktion.

Für jedes extrahierte, eindeutig identifizierbare und semantisch voneinander unterscheidbare Element, die sogenannten *Submodels* und *SubmodelElements,* wird ECLASS verwendet. Die Zuordnung der aus den Dokumenten extrahierten Elemente zum ECLASS erfolgt durch Ermittlung der geringsten Levenshtein-Distanz zwischen der *idShort* der Elemente und dem *PreferredName* im ECLASS [25]. Die Einheiten der Elementwerte werden ebenfalls mithilfe einer Nachschlagtabelle in standardisierte Einheiten umgerechnet. Dabei gibt die erste Spalte die Einheit an (z. B. km), die zweite Spalte die entsprechende SI-Einheit (z. B. m) und die dritte Spalte enthält die Berechnung (z. B. [/, 1000]).

3.3 Erstellung und Zusammenführung von Verwaltungsschalen

Für jede Datei wird zunächst eine eigene VWS erstellt. Jede dieser VWS repräsentiert somit nur einen bestimmten Aspekt eines Assets. Technische Eigenschaften aus Datenblättern, Geometrieinformationen aus 3D-Modellen und die kinematische Struktur von URDF-Assets werden jeweils auf ein *Submodel* der VWS abgebildet.

Da die VWS aus unterschiedlichen Dokumenten mit verschiedenen Dateiformaten erstellt werden, die die Perspektive verschiedener Disziplinen repräsentieren, können Inkonsistenzen zwischen diesen VWS vorliegen. In diesem Beitrag werden sowohl syntaktische als auch semantische Inkonsistenzen betrachtet. Syntaktische Inkonsistenzen beziehen sich auf Unstimmigkeiten oder Fehler in der Struktur oder Grammatik der Erstellungsregeln einer VWS. Solche Inkonsistenzen werden mittels vordefinierter Regeln überprüft und durch die Anwendung der korrekten Syntax korrigiert. Semantische Inkonsistenzen hingegen weisen auf Unstimmigkeiten in der Bedeutung oder in der Interpretation von Informationen hin. Die Erkennung und Behebung dieser Inkonsistenzen erfolgen durch die Anwendung einer Reihe von Regeln und semantischen IDs. Die unterschiedlichen Bezeichnungen, wie beispielsweise Einheiten für technische Eigenschaftswerte, werden mit den in ECLASS definierten Bezeichnungen verglichen und korrigiert. Für Inkonsistenzen, die durch Benutzer verursacht werden, werden entweder Benutzer oder Experten konsultiert und manuell korrigiert. Nach der Auflösung der Inkonsistenz wird eine integrierte VWS mit einer vordefinierten Struktur erstellt, indem alle initial erstellten VWS zusammengeführt werden. Diese vordefinierten Strukturen enthalten *Submodels* wie *Submodel*-Identification [31], [32], *Submodel*-TechnicalProperty [32], *Submodel*-MechanicalBreakdown [33], *Submodel*-Controller [34] und *Submodel*-TransformationConfiguration [34]. Je nach Anwendung werden die jeweiligen *Submodels* entsprechend erfüllt.

Tabelle 1: Übersicht über verwendete Techniken zur Informationsextraktion für verschiedene Dateiformate.

Format	Extraktionsmethode
PDF	Zeilen-Spalten-Strukturanalyse für Tabellen
STP	Parser und Interpretation
XML-basierte Formate (AML, URDF)	Parser und Transformation

4 Implementierung und Ergebnisse

In diesem Abschnitt werden drei ausgewählte Anwendungsfälle beschrieben: der Extended Pick&Place Unit (xPPU) Demonstrator, die Montage von Planeteetrieben und die physische Simulation von Robotern. Die Einzelheiten der Implementierung und die Ergebnisse des vorgeschlagenen Ansatzes werden im Folgenden erläutert.

4.1 Anwendungsfälle

Die Auswahl dieser drei Anwendungsfälle erfolgte, um die Anwendbarkeit und Effizienz des Ansatzes in verschiedenen industriellen Kontexten zu demonstrieren, wobei unterschiedliche Branchen, Dokumentenformate und Komplexitätsgrade berücksichtigt werden. Während der erste

Anwendungsfall eine Laboranlage xPPU [35] ist, beziehen sich die Anwendungsfälle 2 und 3 auf eine Maschine und ein Bauteil, die als Komponenten in industriellen Anlagen verwendet werden. Trotz der Unterschiede in den enthaltenen Informationen weisen alle drei Anwendungsfälle Dokumentendateien mit Dateiformaten, wie PDF, XML und STP-Dateien, auf. Die Untersuchung dieser unterschiedlichen Dateiformate dient der Übertragbarkeit des Ansatzes. Im Folgenden erfolgt eine detaillierte Beschreibung der drei Anwendungsfälle:

Anwendungsfall 1: Die xPPU ist eine Laboranlage, die aus einem Lager, einem Kran, einer Rampe, und weiteren Komponenten, wie z. B. Förderbändern, Weichen und Sicherheitsmodulen besteht. Mittels dieser Komponenten kann die xPPU Werkstücke unterschiedlicher Materialien handhaben [36].Eine Auswahl von Dokumenten und Modellen für den xPPU-Anwendungsfall ist in Abbildung 2, links, gezeigt.

Anwendungsfall 2: Ein häufig verwendeter Roboter ist ein Roboterarm mit sieben Freiheitsgraden und Drehmomentsensoren in jeder Achse [37]. Der Roboter bietet in seiner Forschungskonfiguration eine große Menge an Open-Source-Dokumenten, z. B. Handbücher, URDF- und STP-Dateien (siehe Abbildung 2).

Anwendungsfall 3: Die Planetengetriebemontage stützt sich auf die von den eigenen Abteilungen ermittelten und von den Zulieferern bereitgestellten Informationen, über z. B. Schrauben, Stifte, Normen und Toleranzen [38]. Das montierte Planetengetriebe besteht aus mehreren Teilen, z. B. Zahnrädern, Ritzeln und Bolzen. Nach der Montage wird eine Reihe von Tests mit einem Prüfbericht für die Qualitätskontrolle durchgeführt, darunter auch der End-of-Line-Test. Im Bericht werden Echtzeit-Prüfdaten des Planetengetriebes bereitgestellt. Darüber hinaus werden die eingebetteten Informationen des Getriebes in einer VWS gespeichert, die dem Kunden zur Verfügung gestellt werden kann (siehe Abbildung 2, rechts).

4.2 Implementierungsdetails

Für jedes Dokumentenformat werden im Folgenden die Implementierungsdetails und die Ergebnisse der VWS-Erstellung beschrieben. Für jeden Anwendungsfall sind die verarbeiteten Dateiformate in Tabelle 2 gelistet.

4.2.1 Vorverarbeitung

Zur Identifikation anlagenbezogener Dokumente und zur Extraktion der darin enthaltenen anlagenbezogenen Informationen wurden die nachfolgenden Vorverarbeitungsschritte implementiert. Zuerst wurden sämtliche Dokumente, die in verschiedenen digitalen Formaten vorlagen und im Zusammenhang mit einem Asset standen, gesammelt. Im nächsten Schritt erfolgte eine sorgfältige Prüfung der ermittelten Dokumente, um solche auszuschließen, die

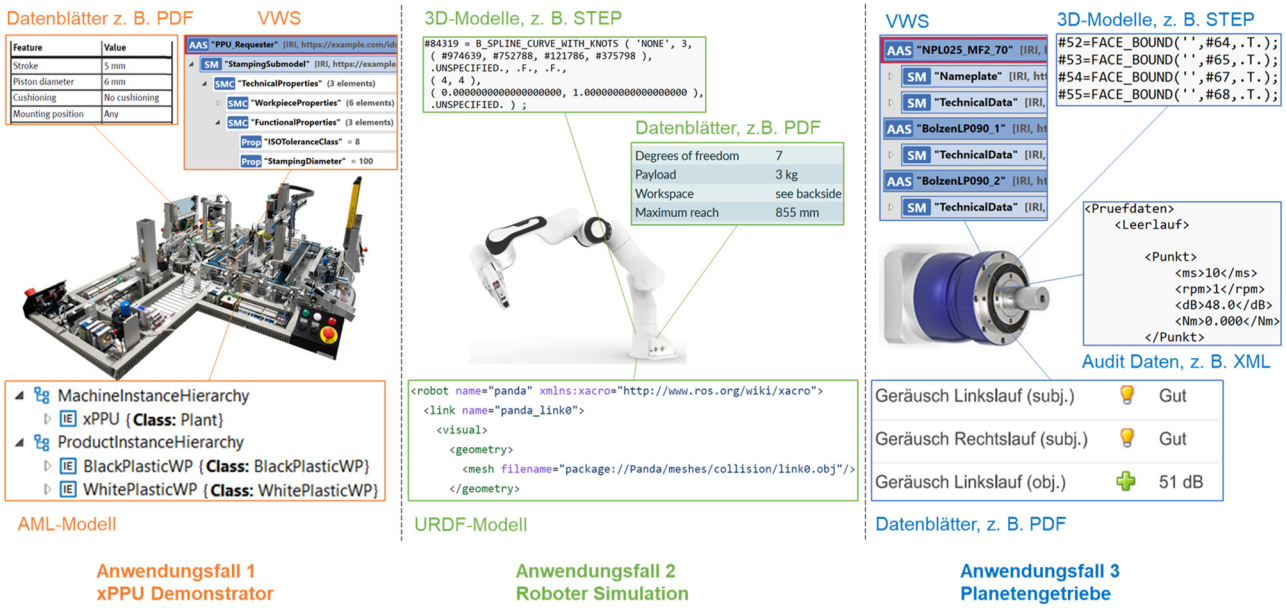

Abbildung 2: Übersicht der Anwendungsfälle für die semi-automatisierte VWS-Erstellung aus den verschiedenen heterogenen Engineering-Dokumenten.

Tabelle 2: Verarbeitete Dateiformate per Anwendungsfall.

Verarbeitetes format	xPPU	Roboter simulation	Planetengetriebe
PDF	Vorhanden	Vorhanden	Vorhanden
STP	Nicht vorhanden	Vorhanden	Vorhanden
VWS	Vorhanden	Nicht vorhanden	Vorhanden
XML-basierte Formate (AML, URDF)	AML	URDF	XML

fehlerhafte Daten enthielten, einen nicht funktionsfähigen Programmcode aufwiesen oder beschädigt waren. Fehlerhafte Daten können auf Unregelmäßigkeiten oder Beschädigungen während verschiedener Phasen des Dokumentenprozesses zurückzuführen sein. Dies kann durch verschiedene Faktoren verursacht werden, wie zum Beispiel Übertragungsfehler, Konvertierungsprobleme oder Dokumente, die über einen längeren Zeitraum nicht aktualisiert wurden. Zum Beispiel fehlt in der manuell erstellten VWS-Datei eines Getriebes die *idShort* einiger Bauteile. Danach wurden die verbleibenden Dokumente gemäß ihren jeweiligen Dateitypen kategorisiert. In jedem Anwendungsfall existierte ein technisches Dokument, das eine umfassende Übersicht über das jeweilige Asset bot. Basierend auf diesem technischen Dokument wurden alle relevanten Dokumente identifiziert, zu denen Datenblätter, 3D-Modelle, Konfigurationsdokumentation und Programmcode gehören. Diese Identifikation erfolgte mittels einer TF-IDF-Analyse. Allerdings gibt es gewisse Einschränkungen bei der Anwendung der TF-IDF-Analyse, insbesondere im Fall von STP-Dateien, die hauptsächlich 3D-Modelldaten enthalten. In diesem Kontext sind die Ergebnisse der TF-IDF-Analyse weniger aussagekräftig. Aus diesem Grund wurde eine manuelle Auswahl der relevanten Dokumente vorgenommen.

4.2.2 Mapping von PDF-Tabellen zu VWS

PDF-Dateien können Titelseite, Inhaltsverzeichnis, Tabellen, Abbildungen, Literaturverzeichnis und andere Elemente enthalten. In dieser Arbeit werden allgemeine zweispaltige Tabellen in Datenblättern sowie Tabellen mit dem erwarteten Layout betrachtet. Die zweispaltige Tabelle wird identifiziert und extrahiert, wobei die linke Spalte die Eigenschaftsnamen und die rechte Spalte die Eigenschaftswerte enthält. Tabellen mit erwartetem Layout könnten zum Beispiel eine 5-spaltige Tabelle für die Definition eines generischen Submodells mit den Spalten PropertyID/SubmodelElementCollectionID, PropertyID, ValueType/DataType, Value und SemanticID umfassen. Nach einer manuellen Bereinigung der Datenblätter durch Entfernen von PDF-Dateien, die nicht mit den Assets in Zusammenhang stehen, erhielten wir PDF-Dateien, die zweispaltige Tabellen umfassen. Die linke Spalte enthält die Eigenschaftsnamen und die rechte Spalte Eigenschaftswerte. Der Mapping-Algorithmus ist in Algorithmus 1 dargestellt. Die Informationsextraktion erfolgte automatisch mit Hilfe des Python-Pakets *Camelot*.[1] Vor der Erstellung der VWS mit den extrahierten technischen Eigenschaften wurde jeder Eigenschaftsname auf ECLASS abgebildet, um die zugehörige semantische ID zu erhalten. Eigenschaftswerte wurden entsprechend auf standardisierte Einheiten umgerechnet. Das Python-Paket *pyenchant*[2] wurde für die Berechnung der Levenshtein-Distanz und Paket *quantulum3*[3] für die Extraktion von Werten und Einheiten aus unstrukturiertem Text verwendet. Die Muster in den *Regular Expressions* (RegEx) sind so gestaltet, dass sie die Formen von „numerischen Werten und Einheiten“ erkennen können. Hierbei sind nicht alle Repräsentationen von Werte und Einheiten definiert, jedoch sind mehr als 290 Einheiten abgedeckt. Für Eigenschaften, die in ECLASS nicht gefunden wurden, wird zunächst eine semantische ID von „0000“ verwendet, die später von Ingenieuren manuell zu korrigieren ist. Zusätzlich wurde mit Hilfe des Python-Pakets *basyx-python-sdk*,[4] einer Python-Implementierung des VWS-Metamodells für I4.0-Systeme nach der Schnittstellenspezifikation in [2], eine VWS im Format.aasx, welche die technischen Eigenschaften des Assets abbildet, erstellt. Allerdings treten dabei Fehler auf. Im Anwendungsfall 1 sollte beispielsweise der Wert für die „Mounting position“ des Kompaktzylinders „Vertikal +/−5.0“ betragen, anstatt „−5.0“. Ebenso sollte der Wert für die „Grade of Filtration“ des Kompaktzylinders „40“ sein, anstatt „5.0“. Dies liegt daran, dass die „Mounting position“ separat als +5 und −5 extrahiert wird, was darauf hinweist, dass der Extraktionsalgorithmus noch verbessert werden muss. Diese Fehler und das Endergebnis des Mappings ist in Abbildung 3, oben, dargestellt.

1 https://camelot-py.readthedocs.io/en/master/.
2 https://github.com/pyenchant/pyenchant.
3 https://github.com/nielstron/quantulum3.
4 https://github.com/eclipse-basyx/basyx-python-sdk.

Algorithmus 1. Mapping-Algorithmus von PDF-Tabelle zu VWS mit Verbindung zu ECLASS.

```
Eingabe: PDF-Dokument mit Tabelle
Ausgabe: VWS-Datei mit Informationen aus Tabellen und Verbindung zu
         ECLASS
Identifiziere die Tabelle im PDF;
Extrahiere die Tabelle aus dem PDF;
Strukturiere die Tabelle in Zeilen und Spalten;
für jede Zeile tue
    Extrahiere Eigenschaftsnamen aus der linken Zelle;
    Extrahiere Eigenschaftswerte (bzw. Wertebereiche mit Zahlen und
     Einheiten) aus der rechten mithilfe Regular Expressions;
    Suche den Eigenschaftsnamen in ECLASS;
    wenn Eigenschaftsnamen in ECLASS gefunden dann
        Erhalte die entsprechende semantische ID und weise die semantische
         ID dem Eigenschaftsnamen zu;
        Erhalte standardisierte Einheiten für die Eigenschaftswerte;
        Konvertiere Eigenschaftswerte in standardisierte Einheiten;
    sonst
        Weise dem Eigenschaftsnamen die semantische ID 0000 zu;
    Ende
Ende
Erstelle für jede Eigenschaft mit semantischer ID ein SubmodelElement
 Property;
Erstelle ein Submodel TechnicalProperty (beinhaltet alle Eigenschaften);
Erstelle ein Asset mit URI und Type Instance für das Submodel;
Generiere die VWS für das Asset im Format .aasx;
```

4.2.3 Mapping von STP zu VWS

Die Daten in einer STP-Datei werden durch die Verwendung von Punkten, Kanten und Flächen sowie topologischen Beziehungen beschrieben, um die 3D-Formen geometrischer Entitäten zu repräsentieren. Die geometrischen Eigenschaften, wie beispielsweise das Volumen der Entität, können nur durch Berechnungen extrahiert werden. Das Paket *pythonocc*[5] wurde als STP-Parser verwendet, der auf Basis einer übergebenen STP-Datei die Geometrieinformationen des 3D-Modells des Assets, d. h. Volumen, Größe der Oberfläche, Schwerpunkt und Bereich (Länge, Breite und Höhe), zurückgibt. Nach dem Mapping auf ECLASS wurde das gleiche Verfahren wie beim Mapping von PDF auf VWS durchgeführt. Es wurde eine VWS mit zwei *Submodels* erstellt. Ein *Submodel* bildet die Identifikation im Kopfteil der STP-Datei ab, während das andere alle verfügbaren Geometrieinformationen im Datenteil präzise abbildet (siehe Abbildung 3, links in der Mitte, der Kopfteil ist rot markiert und der Datenteil gelb).

4.2.4 Mapping von URDF zu VWS

Eine URDF-Datei enthält Informationen über die kinematische Struktur und das Aussehen eines Roboters, die in *Robot Operating System* (ROS) verwendet werden. Diese können weiter unterteilt werden in die mechanischen Komponenten, die Steuerung und die maximalen Drehmomente für Gelenke, die Transformationskonfiguration und technische Eigenschaften wie das Material der Verbindungen. Da die URDF-Datei auf XML basiert, wurde das Python-Paket *ElementTree*[6] verwendet, um diese zu parsen. Ähnlich wie beim Mapping der aus STP extrahierten Elemente auf ECLASS und im Mapping von STP auf VWS wurden fünf *Submodels* in der VWS erstellt. Diese sind: *Submodel* für die Darstellung der Steuerung, *Submodel* für maximale Drehmomente für Gelenke, *Submodel* für mechanische Komponenten, *Submodel* für Transformationskonfiguration und *Submodel* für technische Eigenschaften wie das Material der Verbindungen. Diese *Submodels* dienen dazu, alle Kinetik-Informationen eines

5 https://github.com/tpaviot/pythonocc-core.

6 https://docs.python.org/3/library/xml.etree.elementtree.html.

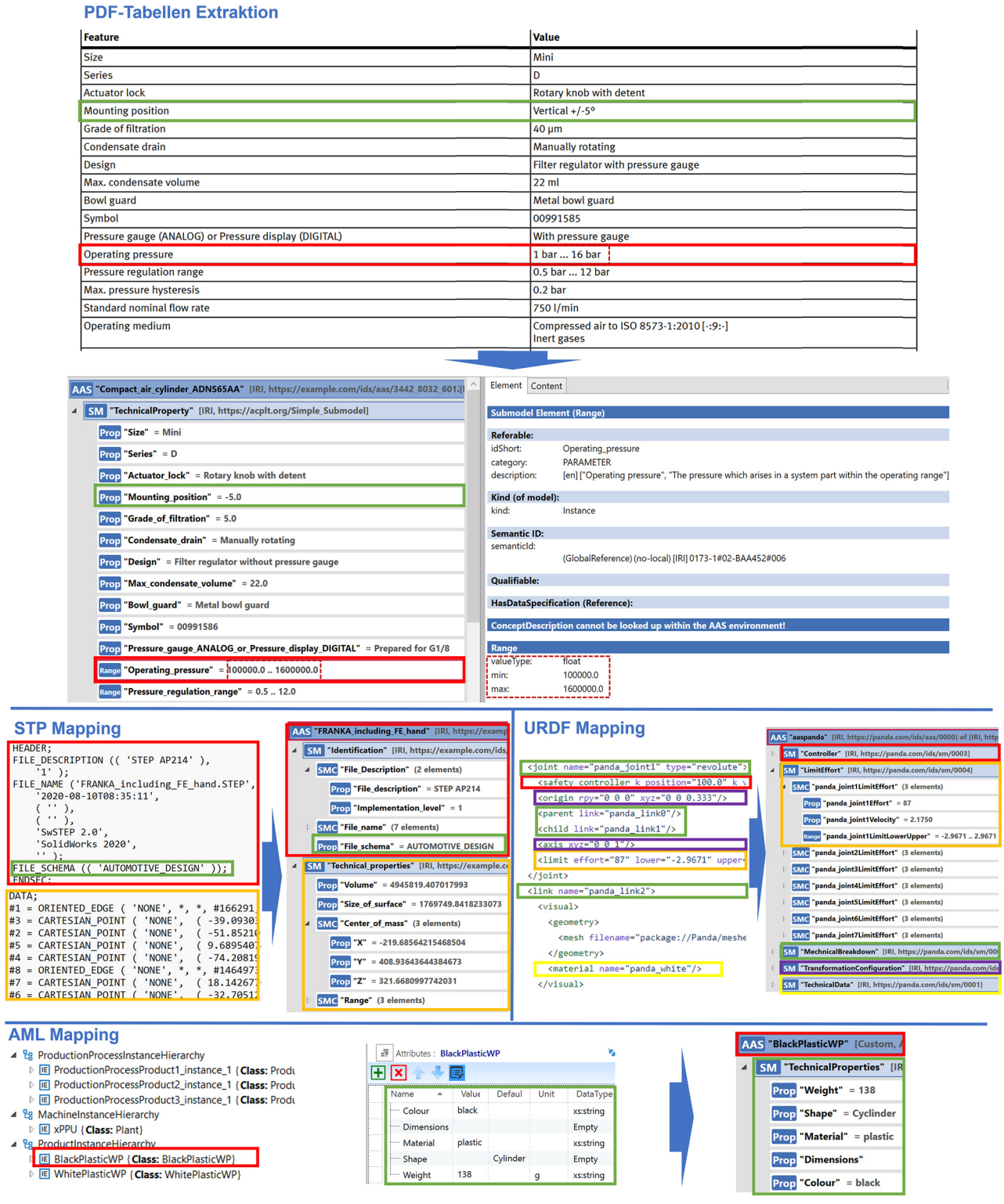

Abbildung 3: Übersicht über die implementierten Informationsextraktionsmethoden. Äquivalente Informationen in technische Daten und der VWS sind farblich hervorgehoben.

Roboters entsprechend darzustellen (siehe Abbildung 3, Mitte rechts).

4.2.5 Mapping von AML zu VWS

AML ist ebenfalls ein XML-basiertes Austauschformat, das jedoch komplexere Beziehungen enthalten kann. Im Anwendungsfall der xPPU werden die Beziehungen zwischen Prozessen, Produkten und Ressourcen (PPR) für die xPPU modelliert. Die VWS-Erstellung erfolgt durch das Mapping jeder Bibliothek von *InterfaceClassLib*, *RoleClassLib* und *SystemUnitClassLib* in AML auf eine VWS, wobei jede Klasse als *Submodel* repräsentiert wird. Zusätzlich wird der Typ dieser VWS als ein „Template“ festgelegt, da jedes „Template“ einer Bibliothek ähnelt und somit wiederverwendbar für die Erstellung von Asset-Instanzen ist. Jedes *InternalElement* mit Ausnahme derjenigen, welche einen Prozess in der InstanceHierarchy darstellen, wurde auf eine VWS abgebildet und „Instance“ als *AssetKind* der *AssetInformation* in der VWS festgelegt, da dies konkret auf Asset-Instanzen verweist. Die Serialisierung von AML zu VWS ist in Abbildung 3, unten, dargestellt. Die Abbildung des Prozesses und der Beziehungen zwischen Produktion, Prozess und Ressource für den Anwendungsfall 1 befindet sich noch in der Entwicklung. Der zukünftige Plan besteht darin, durch die Integration von VWS, AML und Open Platform Communications Unified Architecture (OPC UA) „eine umfassende vereinheitlichte digitale Interoperabilitätslösung für Geräte, Software und allgemein zwischen Assets während ihres gesamten Lebenszyklus“ zu schaffen [39]. Ein Beispiel hierfür ist die Asset-Integration durch VWS, die die Kommunikation mit dem entsprechenden Asset sowie erforderliche Daten in die VWS einbindet, und zwar über etablierte Interoperabilitätsstandards wie OPC UA. IDTA bietet die Submodel-Templates „Asset Interface Description“ (AID) an, die den Zugang zum Asset ermöglicht, um mit einem Asset oder Service zu interagieren [16]. Dies umfasst beispielsweise den Zugriff auf erforderliche Daten, die Registrierung dieser Daten und deren Integration in die VWS über Kommunikationsprotokolle wie Modbus, OPC UA, HTTP und MQTT. Spezifische Detailinformationen über ein Asset werden nicht dupliziert, sondern stattdessen in VWS-*Submodel* referenziert und können in AutomationML oder Simulationsmodellen eingebettet sein.

4.2.6 Inkonsistenzmanagement und VWS-Zusammenführung

Nach dem Mapping-Prozess erhielten wir mehrere VWS, die unterschiedlichen Aspekte des gewünschten Assets interpretieren. Der nächste Schritt ist die Zusammenführung dieser VWS zu einer integrierten VWS. Um eine integrierte VWS für z. B. die Planetengetriebebaugruppe zu erstellen, wurde eine vordefinierte integrierte VWS ohne Betriebsdaten mit dem AASXPackage-Explorer[7] bereitgestellt. Die Betriebsdaten wurden extrahiert, indem die XML-Datei geparst wurde und alle Betriebsdaten der gleichen Kategorie in einem einzigen Vektor zusammengefasst wurden. Mit Hilfe des Python-Pakets *basyx-python-sdk* wurden die Betriebsdaten der integrierten VWS hinzugefügt, indem für jede Datenkategorie ein *Submodel* erstellt wurde. Das Mapping auf ECLASS wurde nicht implementiert, da die Sprache des Anwendungsfalls Deutsch ist. Die Umsetzung des Mappings deutscher Begriffe in ECLASS ist von verschiedenen Faktoren abhängig, wie der Anzahl der zu mappenden Begriffe, der Konsistenz der Terminologie und der Verfügbarkeit von Tools für das Mapping. Je umfangreicher die Begriffsmenge, die Terminologie und je fortschrittlicher die Tools, desto aufwändiger könnte die Umsetzung sein. Da es mehr Tools und Forschung für englische Begriffe gibt, ist es sinnvoll, die Umsetzbarkeit und Genauigkeit des Mappings vorerst nur für Englisch zu betrachten. In diesem Beitrag wird das Mapping auf ECLASS ausschließlich in englischer Sprache durchgeführt. Bei unterschiedlichen Sprachen können mehrsprachige Eigenschaften der VWS mittels sprachneutraler Identifikatoren wie ECLASS, genutzt werden. Das ermöglicht beispielsweise eine Produktbeschreibung in verschiedenen Sprachen von internationalen Herstellern. Sprachtags können auch dabei helfen, die Sprache jeder Information eindeutig zu kennzeichnen und den Sprachkontext zu klären. Vor der Integration der VWS wurden Konsistenzprüfungen mit einer Reihe von vordefinierten Regeln durchgeführt. Eine Übersicht über diese Inkonsistenzen und die Regeln zur Lösung der Inkonsistenzen finden sich in Tabelle 3. Einige Regeln sind in [2] spezifiziert, z. B. sollte die *idShort*, welche ein Element innerhalb eines *Submodel* identifiziert, nur Buchstaben, Ziffern und Unterstriche enthalten und zwingend mit einem Buchstaben beginnen. Diese Regeln wurden als Grundlage für die Syntaxprüfung der erstellten VWS betrachtet. Andere Regeln wurden entsprechend der semantischen Prüfung der erstellten VWS definiert, z. B. sollten die Echtzeit-Betriebsdaten innerhalb des Bereichs der von den Betreibern aufgezeichneten Audit-Werte liegen. Solche Inkonsistenzen wurden durch den Vergleich der Eigenschaftswerte aller Elemente mit der gleichen semantischen ID, gelöst. Zwei VWS wurden in die vordefinierte VWS integriert. Die erste VWS umfasst die technischen Eigenschaften des Getriebes und das standardisierte *Nameplate* gemäß ZVEI [32]. Die eingefügten Informationen basieren

7 https://github.com/admin-shell-io/aasx-package-explorer.

Tabelle 3: Übersicht über die betrachteten Inkonsistenzen und die Regeln zur Behebung der Inkonsistenzen.

Inkonsistenzen	Regeln zur Lösung der Inkonsistenzen	Beispiel
Syntaktische Inkonsistenzen		
Die idShort für referenzierbare Elemente folgen nicht den erlaubten Einschränkungen. (Die *idShort* für ein referenzierbares Element soll nur Buchstaben, Ziffern und Unterstriche enthalten und zwingend mit einem Buchstaben beginnen. Die maximale Länge der *idShort* soll 128 Zeichen betragen.)	1. Die *idShort* wird mithilfe einer Lookup-Tabelle angepasst, in der Sonderzeichen auf die entsprechenden angepassten Zeichen abgebildet werden. 2. Falls die *idShort* nicht mit einem Buchstaben beginnt, wird der Name des *Submodels* am Anfang der *idShort* hinzugefügt. Der Name des Submodels wird ebenfalls überprüft, ob er mit einem Buchstaben beginnt. 3. Die Länge der *idShort* wird auf maximal 128 Zeichen begrenzt. Danach wird abgeschnitten.	Inkonsistenz: *idShort* für eine Eingenschaft = Temperature (°C) Korrigiert: *idShort* für die Eigenschaft = Temperature_grad_celsius (Siehe Beispiel in Abbildung 4.)
Die *valueTypes* des *SubmodelElements* z. B. *Property/valueType* befinden sich nicht innerhalb der vordefinierten Typen.	Die *valueTypes* von S*ubmodelElements*, die sich nicht innerhalb der vordefinierten Typen befinden, werden standardmäßig als *xs:string* betrachtet[a].	Inkonsistenz: *valueType* für einen Eigenschaftswert = *xs:token* Korrigiert*: valueType* für den Eigenschaftswert = *xs:string*
Die *URI* für eine VWS ist nicht vorhanden.	Eine *URI* für die VWS wird wie folgt zugewiesen: http://www.<AssetName>.de/aas/example/0000.	Falsch: keine *URI* für die VWS für die xPPU vorhanden Korrigiert: die URI für die VWS = http://www.xppu.de/aas/example/0000
	Semantische Inkonsistenzen	
Echtzeit-Betriebsdaten liegen nicht innerhalb des Bereichs der von den Betreibern aufgezeichneten Auditwerte.	Inkonsistenzen werden durch den Vergleich der Eigenschaftswerte aller *SubmodelElements* mit derselben semantischen ID identifiziert, dokumentiert und später werden die korrekten Informationen von den Bedienern eingeholt.	Inkonsistenz: Die Betriebsdaten der Umgebungs-temperatur werden von einem Temperatursensor erfasst [18 °C, 18,1 °C, …, 20 °C], während ein Bediener aufgezeichnete Daten für die minimale und maximale Umgebungstemperatur in diesem Zeitraum angibt [17 °C, 20 °C], was aufgrund eines Fehlers vom Bediener falsch aufgezeichnet wurde. Korrigiert: Die Betriebsdaten bleiben unverändert, während die Audit-Daten laut Bedienern [18 °C, 20 °C] sein sollten.
Unterschiedliche *idShort*, *valueType* und Einheiten werden für unterschiedliche *SubmodelElements* genutzt, jedoch mit derselben Bedeutung.	Die semantische ID wird möglichst jedem *SubmodelElement* zugewiesen.	Inkonsistenz: Unterschiedliche *idShort* wie z. B. „Temperatur“ und „Umgebungstemperatur“ werden für die Eigenschaftsnamen in unterschiedlichen VWS verwendet, jedoch mit gleicher Bedeutung „Temperatur im äußeren Umfeld des Betriebsmittels“. Korrigiert: Die semantische ID 0173-1#02-BAE540#004 wird beiden Eigenschaften zugewiesen.
Falsche *valueTypes* und Einheiten werden für *SubmodelElements* genutzt.	Inkonsistenzen werden durch Vergleichen des genutzten und standardisierten *valueType* sowie der Einheit in ECLASS identifiziert und dokumentiert. Später werden die korrekten Informationen bei Ingenieuren angefragt.	Inkonsistenz: Einheit für die Technische Eigenschaft „Temperatur“ = m Korrigiert: Einheit für die Technische Eigenschaft „Temperatur“ = grad celsius

[a]Für die Bedeutung dieser Typen siehe Liste von Resource Description Framework (RDF)-kompatiblen XML Schema Definition (XSD)-Typen mit kurzer Beschreibung von das World Wide Web Consortium (W3C). https://www.w3.org/TR/rdf11-concepts/#xsddatatypes.

auf den vom Hersteller bereitgestellten Datenblättern bezüglich des Getriebes und des Herstellers. Die andere VWS erfolgt über das Prüfprotokoll am Ende der Produktionslinie im Anwendungsfall 3. Diese VWS behandelt zeitliche Reihen von Daten und wurde nicht mithilfe eines Submodel-Templates erstellt. Es besteht jedoch die Möglichkeit, in der Zukunft das IDTA Submodel-Template „Time Series Data“ für die Erstellung der VWS zu verwenden. Die integrierte VWS wird in Abbildung 4 dargestellt. Im Vergleich zu den bereits vorhandenen VWS von xPPU und Planetengetriebe bieten die generierten VWS Vorteile. Sie sind umfassender und berücksichtigen die Gesamtaspekte eines Assets über seinen Lebenszyklus hinweg. Zudem besteht das Potenzial, dass die mit dem semi-automatisierten Ansatz generierten VWS möglicherweise fehlerfreiere Informationen enthalten, im Vergleich zur manuellen Erstellung von VWS, die

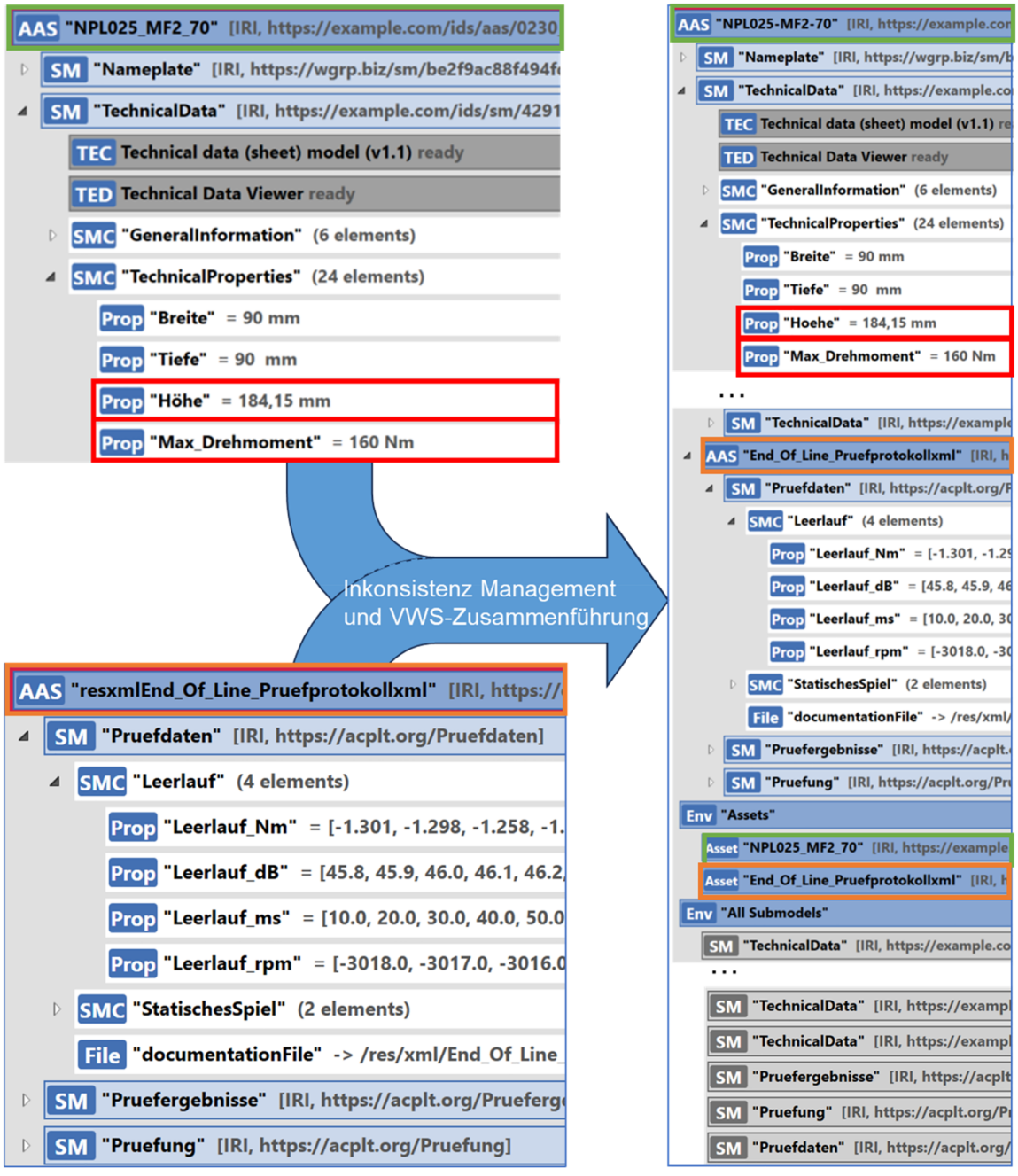

Abbildung 4: Zusammenführung der aus den jeweiligen Datenformaten resultierenden Verwaltungsschalen zu einer integrierten VWS unter Berücksichtigung von möglichen Informationsdopplungen und Inkonsistenzen.

zeitaufwendig und fehleranfällig ist. Die generierten VWS profitieren außerdem davon, dass sie weniger Inkonsistenzen aufweisen, da diese vor der Integration der separaten VWS aufgelöst werden.

Die Inkonsistenzmethode wird im Unternehmen verwendet, um die manuell erstellte VWS zu bereinigen. Die *idShort*, die nicht mit den Einschränkungen der VWS übereinstimmen, werden identifiziert und gelöscht (siehe rote Hervorhebungen in Abbildung 4).

5 Diskussion der Ergebnisse

Der vorgeschlagene Ansatz zur Verwaltung heterogener Datenquellen bei der Erstellung eines VWS erweitert den Stand der Technik, indem Regeln für das Mapping von STP zu VWS und URDF zu VWS erstellt werden. Es wurde ein Konzept für die Zusammenführung von VWS aus verschiedenen Sichten in eine integrierte VWS vorgeschlagen. Darüber hinaus wird das Konzept der Inkonsistenzprüfung im Rahmen der Integration mehrerer VWS erweitert. Im Vergleich zu einer manuellen Implementierung verbessert dieser Ansatz die Effizienz bei der Erstellung einer VWS. Am Beispiel ist durch die Erstellung einer VWS aus einer PDF-Tabelle der manuelle Aufwand erheblich reduziert worden, da die Erstellung einer VWS in Sekunden verringert wird. Dieser Ansatz dient auch als Ausgangspunkt für die Erkundung weiterer Dokumentenformate und ermöglicht die Verwaltung heterogener Datenquellen durch die Erstellung von VWS.

Allerdings gibt es noch einige Einschränkungen und Verbesserungsmöglichkeiten. Zum einen werden in dieser Arbeit nur ausgewählte Formate von Dokumenten mit strukturierten Daten untersucht. Unstrukturierte Daten, wie z. B. ausformulierter Text, können mit LLMs (Large Language Models) effizienter verarbeitet werden [40]. Für die Extraktion von Informationen aus Abbildungen in einer PDF-Datei soll *Optical Character Recognition* (OCR, dt. optische Zeichenerkennung) verwendet werden. Zum anderen sind formale Daten wie Daten in Ontologien nicht weit verbreitet, aber ihre Integration in eine VWS wäre mittels SWT vergleichsweise unkompliziert. Auch die bisher betrachteten Methoden zur Informationsextraktion können in Zukunft erweitert werden. Die Analyse komplexer Tabellenstrukturen, z. B. von Tabellen über mehrere Seiten, mehrspaltige Tabellen oder Tabellen mit verschachtelten Strukturen, basiert auf der Grundlage zweispaltiger Tabellen. Zum Beispiel können Klassen-Instanz-Beziehungen und Instanz-Instanz-Beziehungen in einer Tabelle identifiziert und extrahiert werden. Beim Mapping von STP zu VWS werden die geometrischen Eigenschaften durch Berechnungen extrahiert. Diese Methode bietet die Möglichkeit, bestimmte Zusammenhänge oder Abhängigkeiten in den Daten zu identifizieren, die später bei anderen Mapping-Methoden genutzt werden können. Bei AML wird jedes AML-Element einer VWS zugeordnet, aber die Beziehungen zwischen diesen AML-Elementen werden nicht weiter berücksichtigt, sodass die in die VWS übertragenen Informationen nicht vollständig sind. Zum Beispiel können die Instanz-Instanz-Beziehungen zwischen AML-Elementen *InternalElement* durch das *RelationshipElement* dargestellt werden. Die Klassen-Instanz-Beziehung zwischen beispielsweise *InternalElement* und *SystemUnitClasses* kann in VWS durch derivedFrom zwischen assetKind und assetTyp dargestellt werden. Außerdem verursachen die Informationsextraktion und Information-Mapping einige Fehler, die manuell korrigiert werden müssen. Fehler bei der Informationsextraktion und Information-Mapping können durch die Berücksichtigung aller möglichen Bedingungen oder durch die Kombination verschiedener leistungsfähiger Techniken wie NLP, Data-Mining-Techniken und SWT verbessert werden.

Gleichzeitig hat die Verwendung von ECLASS als Wörterbuch der standardisierten Semantik einige Nachteile. Einer der Nachteile ist, dass es für ein und dasselbe Wort unterschiedliche Bezeichner geben kann, wenn es in verschiedenen Bereichen unterschiedlich verwendet wird. Dies betrifft zwei Fälle. Im ersten Fall geht es um Synonyme, d. h. unterschiedliche Bezeichnungen für Objekte mit gleicher Bedeutung. IEC CDD und ECLASS regeln den Umgang mit Synonymen. Der sogenannte „Preferred Name“ in ECLASS ist eine eindeutige Bezeichnung im Klassifikationssystem, und die dazugehörige Beschreibung erklärt die Bedeutung. In diesem Beitrag werden nur ähnliche Bezeichnungen in der Datei und in ECLASS herausgesucht. Eine andere Bezeichnung der Datei, die zwar die gleiche Bedeutung hat, aber keine ähnliche Bezeichnung in ECLASS aufweist, wird nicht herausgesucht und sollte vom Ingenieur manuell hinzugefügt werden. Dies könnte in der Zukunft mit NLP behandelt werden. Im zweiten Fall betrifft es die semantische Vielfalt eines Wortes, d. h. dasselbe Wort kann in verschiedenen Kontexten eine unterschiedliche Bedeutung haben. Dies kann zu möglichen Missverständnissen führen. Die semantische Bedeutung des Wortes sollte innerhalb spezifischer Kontexte verstanden werden. Dies wird nicht in diesem Beitrag behandelt und sollte in der Zukunft mit Ansätze, wie beispielsweise dem NLP oder SWT angegangen werden. Die Verwendung der Levenshtein-Distanz zur Zuordnung eines Elements zum ECLASS ist noch nicht genau genug, da der bloße Vergleich des Elementnamens Mehrdeutigkeiten nicht ausschließt. Wenn ein

bestimmtes Wort in ECLASS nicht gefunden werden kann, müssen die neuen Einträge in ECLASS manuell angelegt werden. Die Umrechnung der Einheiten erfolgt durch eine Beispiel-Nachschlagtabelle. Existierende Bibliotheken, wie beispielsweise die Python-Bibliothek „QuantiPhy“,[8] können ebenfalls für die Umrechnung der Einheiten eingesetzt werden. Die Originaldokumente, die Informationen über das Asset enthalten, sollten ebenfalls im VWS dokumentiert werden. Dies ermöglicht die Rückverfolgung der Originaldokumente und unterstützt die Versionierung des Assets.

Der Mapping-Prozess zu ECLASS wird in unserer Implementierung außerdem ausschließlich in englischer Sprache durchgeführt. Wenn der Mappingalgorithmus für deutsche Begriffe in ECLASS identisch mit dem für englische Begriffe ist und die Levenshtein-Distanz verwendet wird, sollte die Mappinggenauigkeit für deutsche Begriffe gleich hoch sein wie für englische Begriffe. In Fällen, in denen der Ausgangstext auf Deutsch ist und ECLASS in Englisch vorliegt, wäre eine zusätzliche Übersetzung erforderlich. Hierbei könnte die Genauigkeit des Mappings niedriger ausfallen, da Übersetzungsprozesse zusätzliche Herausforderungen und potenzielle Bedeutungsverschiebungen mit sich bringen können. Sofern Tools für das Mapping auf NLP basieren, könnte die Genauigkeit besser sein, da die semantische Bedeutung berücksichtigt wird. Da die Genauigkeit für englische Begriffe in diesem Beitrag noch nicht zufriedenstellend ist, wird das Mapping für deutsche Begriffe derzeit nicht in Betracht gezogen. Dies stellt eine offene Forschungsfrage dar.

Zusätzlich wird das Inkonsistenzmanagement mithilfe regelbasierter Ansätze durchgeführt, die durch bestehende Regeln nur begrenzt abgedeckt sind. Die Regeln in Tabelle 3 können weiterentwickelt werden. Zum Beispiel können lange *idShorts* mithilfe einer Lookup-Tabelle auf 128 Zeichen gekürzt werden. Beim Zurücklesen der Namen kann die Lookup-Tabelle erneut verwendet werden, um den Namen zu vervollständigen. Für die Umwandlung von Werten vom Typ Integer zu String sind zusätzliche Schritte erforderlich. Dies könnte beispielsweise die vorherige Umwandlung ungültiger Ganzzahlen in Gleitkommazahlen einschließen. Weitere Regeln können hinzugefügt werden, um Inkonsistenzen heterogener Datenquellen im Rahmen der VWS-Erstellung zu verwalten [29]. Andere Ansätze, wie z. B. die Modellsynchronisation, können in der zukünftigen Arbeit verwendet werden.

Schließlich wurde in dieser Arbeit nur die statische VWS erforscht. Die Erstellung von dynamischen VWS aus dynamischen Datenquellen während des Lebenszyklus eines Assets ist noch in der Entwicklung.

6 Schlussfolgerungen und Ausblick

Im Zusammenhang mit der Erstellung von VWS für I4.0-Komponenten ist die manuelle Extraktion von Asset-Informationen aus einer Vielzahl heterogener Dokumente und deren Speicherung in einer VWS sehr zeitaufwändig. Ein automatischer oder zumindest semi-automatisierter Ansatz könnte diesen Arbeitsaufwand erheblich reduzieren und zudem eine erhöhte Konsistenz gewährleisten, erfordert aber die Kombination verschiedener Techniken und Werkzeuge. In diesem Beitrag wird deshalb ein Ansatz zur semi-automatisierten Erstellung von VWS aus heterogenen Dokumenten in der Domäne der Produktionssysteme vorgestellt. Anhand von drei Anwendungsfällen wird dieser Ansatz für die Untersuchung von Dokumentenformaten wie PDF und STP sowie XML-basierten Formaten wie AML und URDF aufgezeigt. Das Information-Mapping aus den einzelnen Formaten zu einer VWS wird im Detail erläutert. Darüber hinaus ermöglicht das Mapping von extrahierten Elementen zu ECLASS unter Verwendung der Levenshtein-Distanz die Interoperabilität von VWS in verschiedenen Anwendungen und Branchen. Im Rahmen der Zusammenführung verschiedener Modelle erstellte VWS werden Inkonsistenzen durch eine Reihe von vordefinierten Regeln behoben. Für die Datenintegration wird das Konzept gewählt, verschiedene VWS aus unterschiedlichen Dokumenten zu erstellen, um eine in sich konsistente und gültige VWS zu erschaffen. Obwohl während der Verarbeitung einige Fehler auftraten, können diese mit geringem Arbeitsaufwand manuell korrigiert werden. Die Ergebnisse der Implementierung zeigen, dass dieser Ansatz den Prozess der VWS-Erstellung erleichtert und für die Untersuchung anderer Dokumentenformate verallgemeinert werden kann.

In zukünftiger Forschung wird der Ansatz für die Informationsextraktion aus bisher unberücksichtigten Dokumentenformaten mit Hilfe fortgeschrittener Techniken wie NLP und SWT erweitert. Die dynamische Erstellung von VWS ist ebenfalls vielversprechend, wobei hierfür eine umfangreiche Menge an Echtzeitdaten während des Betriebs berücksichtigt werden muss. Weitere Methoden zur Inkonsistenzverwaltung während der VWS-Erstellung können dem Ansatz ebenfalls hinzugefügt werden. Darüber hinaus kann eine GUI für Ingenieure entwickelt werden, um technische Dokumente zu importieren und Fehler bei der Zuordnung von Elementen zu ECLASS zu korrigieren.

8 https://quantiphy.readthedocs.io/en/stable/.

Danksagung: Dieser Beitrag wurde im Rahmen der vom Bayerischen Staatsministerium für Wirtschaft, Landesentwicklung und Energie (StMWi) geförderten Leuchtturm-Initiative KI.FABRIK (Phase 1: Infrastruktur- und F&E-Programm, Fördernummer DIK0249) unterstützt.
Research ethics: Not applicable.
Author contributions: The authors have accepted responsibility for the entire content of this manuscript and approved its submission.
Competing interests: The authors state no conflict of interest.
Research funding: Bavarian State Ministry for Economic Affairs, Regional Development and Energy (StMWi) Lighthouse Initiative KI.FABRIK, (Phase 1: Infrastructure and R&D program, grant no. DIK0249).
Data availability: Not applicable.

Literatur

[1] B. Vogel-Heuser and K. Bengler, "Von Industrie 4.0 zu Industrie 5.0 — Idee, Konzept und Wahrnehmung," *HMD Praxis der Wirtschaftsinformatik*, vol. 60, pp. 1124—1142, 2023.
[2] S. Bader, E. Barnstedt, H. Bedenbender, B. Berres, M. Billmann, and M. Ristin, *Details of the Asset Administration Shell — Part 1 (The Exchange of Information between Partners in the Value Chain of Industrie 4.0 (Version 3.0RC02))*, Berlin, Federal Ministry for Economic Affairs and Climate Action (BMWK), 2022.
[3] Festo, "Creating Digital Twins is quicker and error-free with Festo's award-winning online product catalogue," [Online]. Available at: https://press.festo.com/de/node/3770 Accessed: Nov. 27, 2023.
[4] R. Bolton, *et al*., "Customer experience challenges: bringing together digital, physical and social realms," *J. Serv. Manag.*, vol. 29, no. 5, pp. 776—808, 2018.
[5] Eclipse foundation, "Eclipse Ditto," [Online]. Available at: https://projects.eclipse.org/projects/iot.ditto Accessed: Nov. 28, 2023.
[6] Microsoft, "Azure digital twins," [Online]. Available at: https://azure.microsoft.com/en-us/products/digital-twins Accessed: Nov. 28, 2023.
[7] B. Vogel-Heuser, F. Ocker, I. Weiß, R. Mieth, and F. Mann, "Potential for combining semantics and data analysis in the context of digital twins," *Philos. Trans. R. Soc., A*, vol. 379, no. 2207, p. 20200368, 2021.
[8] D. Lehner, S. Sint, M. Vierhauser, W. Narzt, and M. Wimmer, "AML4DT: a model-driven Framework for developing and maintaining digital twins with AutomationML," in *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2021, pp. 1—8.
[9] R. Drath, A. Luder, J. Peschke, and L. Hundt, "AutomationML — the glue for seamless automation engineering," in *2008 IEEE International Conference on Emerging Technologies and Factory Automation*, 2008, pp. 616—623.
[10] Z. Bradac, P. Marcon, F. Zezulka, J. Arm, and T. Benesl, "Digital twin and AAS in the industry 4.0 Framework," in *IOP Conference Series: Materials Science and Engineering*, vol. 618, 2019, p. 12001.
[11] Eclass, "ECLASS standard," [Online]. Available at: https://eclass.eu/eclass-standard Accessed: Nov. 27, 2023.
[12] International Electrotechnical Commission, "IEC 61360-4 — IEC/SC 3D — common data dictionary (CDD — V2.0018.0001)," [Online]. Available at: https://cdd.iec.ch/cdd/iec61360/iec61360.nsf/TreeFrameset?OpenFrameSet Accessed: Nov. 27, 2023.
[13] A. Belyaev, *et al*., "Modelling the semantics of data of an asset administration shell with Elements of ECLASS," in *Joint White Paper from Plattform Industrie 4.0 and ECLASS*, 2021.
[14] K. Wei, J. Z. Sun, and R. J. Liu, "A review of asset administration shell," in *2019 IEEE International Conference on IEEM*, 2019, pp. 1460—1465.
[15] S. Grüner, M. Hoernicke, K. Stark, N. Schoch, N. Eskandani, and J. Pretlove, ""Towards asset administration shell-based continuous engineering in process industries"," *at — Automatisierungstechnik*, vol. 71, no. 8, pp. 689—708, 2023.
[16] IDTA, "AAS submodel templates," [Online]. Available at: https://industrialdigitaltwin.org/content-hub/teilmodelle Accessed: Nov. 21, 2023.
[17] F. Ocker, Knowledge representation and integration in the context of production systems," Ph.D. thesis, Technical university munich, 2022.
[18] S. Sarawagi, "Information extraction," *Foundations and Trends in Databases*, vol. 1, no. 3, pp. 261—377, 2007.
[19] A. Lüder, A.-K. Behnert, F. Rinker, and S. Biffl, "Generating industry 4.0 asset administration shells with data from engineering data logistics," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2020, pp. 867—874.
[20] X. Ye, W. S. Song, S. H. Hong, Y. C. Kim, and N. H. Yoo, "Toward data interoperability of enterprise and control applications via the industry 4.0 asset administration shell," *IEEE Access*, vol. 10, pp. 35795—35803, 2022.
[21] M. Both, J. Müller, and C. Diedrich, "Automatisierte Abbildung semantisch heterogener I4.0-Verwaltungsschalen durch Methoden des Natural Language Processing," *at — Automatisierungstechnik*, vol. 69, no. 11, pp. 940—951, 2021.
[22] A. Cartus, M. Both, N. Maisch, J. Müller, and C. Diedrich, "Interoperability of semantically heterogeneous digital twins through Natural Language Processing methods," in *CLIMA 2022 Conference*, 2022.
[23] S. Sierla, M. Azangoo, K. Rainio, N. Papakonstantinou, A. Fay, P. Honkamaa, and V. Vyatkin, "Roadmap to semi-automatic generation of digital twins for brownfield process plants," *J. Ind. Inf. Integr.*, vol. 27, p. 100282, 2022.
[24] Y. Xia, N. Jazdi, and M. Weyrich, "Automated generation of Asset Administration Shell: a transfer learning approach with neural language model and semantic fingerprints," in *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2022, pp. 1—4.
[25] C. Diedrich, T. Schröder, J. Arm, and F. Zezulka, "Information retrieval for configuration of I4.0 asset administration shells,"

in *Entwurf komplexer Automatisierungssysteme: EKA 2020*, Magdeburg, EKA, 2020.
[26] I. Garmaev, T. Miny, T. Kleinert, A. Schüller, and P. Bitterlich, "Verwaltungsschalen aus Excel? Automatisierte Erstellung von Verwaltungsschalen aus Bestandsdaten aus Excel-Tabellen," *ATP Magazin*, vol. 65, no. 3, pp. 80—86, 2023.
[27] C. Hildebrandt, S. Törsleff, B. Caesar, and A. Fay, "Ontology building for cyber-physical systems: a domain expert-centric approach," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, 2018, pp. 1079—1086.
[28] J. A. Domínguez, R. P. Fuentes, M. Vegetti, L. Roldán, S. Gonnet, and M. J. Diván, "Ontology implementation of OPC UA and AutomationML: a review," in *Advanced Intelligent Technologies for Industry*, K. Nakamatsu, R. Kountchev, S. Patnaik, J. M. Abe, and A. Tyugashev, Eds., 2022, pp. 17—26.
[29] S. Feldmann, K. Kernschmidt, M. Wimmer, and B. Vogel-Heuser, "Managing inter-model inconsistencies in model-based systems engineering: application in automated production systems engineering," *J. Syst. Softw.*, vol. 153, pp. 105—134, 2019.
[30] S. Robertson, "Understanding inverse document frequency: on theoretical arguments for IDF," *J. Doc.*, vol. 60, no. 5, pp. 503—520, 2004.
[31] ZVEI, *Examples of the Asset Administration Shell for Industrie 4.0 Components — Basic Part* , Frankfurt, ZVEI — German Electrical and Electronic Manufacturers' Association, 2017.
[32] S. Bader, *et al*., "Submodel Templates of the asset administration shell," *Tech. Rep.*, Plattform Industrie 4.0, 2020.
[33] AASX Package Explorer, "Samples for asset administration shell (AAS): festo demo box," [Online]. Available at: http://www.admin-shell-io.com/samples/ Accessed: Jan. 08, 2024.
[34] ROS, "urdf/XML/joint," [Online]. Available at: https://wiki.ros.org/urdf/XML/joint Accessed: Jan. 08, 2024.
[35] B. Vogel-Heuser, S. Bougouffa, and M. Sollfrank, *Researching Evolution in Industrial Plant Automation: Scenarios and Documentation of the Extended Pick and Place Unit*, Garching, Lehrstuhl für Automatisierungstechnik und Informationssysteme, Technische Universität München, 2018.
[36] B. Vogel-Heuser, C. Legat, J. Folmer, and S. Feldmann, *Researching Evolution in Industrial Plant Automation: Scenarios and Documentation of the Pick and Place Unit*, Garching, Institute of Automation and Information Systems, Technische Universität München, 2014.
[37] Franka robotics, "Franka production 3," [Online]. Available at: https://franka.de/production Accessed: Nov. 28, 2023.
[38] Wittenstein, "Products by WITTENSTEIN," [Online]. Available at: https://www.wittenstein.de/en-en/products/servo-gearboxes/ Accessed: Nov. 28, 2023.
[39] R. Drath, *et al*., "Diskussionspapier-Interoperabilität mit der Verwaltungsschale, OPC UA und AutomationML," [Online]. Available at: https://opcfoundation.org/wp-content/uploads/2023/04/Diskussionspapier-Zielbild-und-Handlungsempfehlungen-fur-industrielle-Interoperabilitat-5.3-protected.pdf Accessed: Aug. 29, 2023.
[40] M. U. Hadi, *et al*., "A survey on large language models: applications, challenges, limitations, and practical usage," *TechRxiv*, pp. 1—30, 2023. https://doi.org/10.36227/techrxiv.23589741.v1.

Kurzbiographien

Jingyun Zhao
Lehrstuhl für Automatisierung und Informationssysteme, TUM School of Engineering and Design, Technische Universität München, Garching, Deutschland;
jingyun.zhao@tum.de

Jingyun Zhao hat 2022 einen M.Sc. in Maschinenbau an der TU München erworben und promoviert derzeit am Lehrstuhl für Automatisierung und Informationssysteme an der TU München. Ihre Forschung konzentriert sich auf Informationsmodellierung, maschinelles Lernen und die Entwicklung von digitalen Zwillingen in automatisierten Produktionssystemen.

Birgit Vogel-Heuser
Lehrstuhl für Automatisierung und Informationssysteme, TUM School of Engineering and Design, Core Member der MDSI und Leiterin der Sector Work der MIRMI, Technische Universität München, Garching, Deutschland;
vogel-heuser@tum.de

Prof. Dr.-Ing. Birgit Vogel-Heuser erwarb ein Diplom in Elektrotechnik und einen Doktortitel in Maschinenbau an der RWTH Aachen. Seit 2009 ist sie Professorin und Ordinaria des Lehrstuhls für Automatisierung und Informationssysteme an der TU München. Ihre aktuellen Forschungsschwerpunkte liegen im Bereich Systems- und Software Engineering. Sie ist Mitglied der acatech, IEEE Fellow, Senior Editor der IEEE T-ASE und für das Thema Digitaler Zwilling in der KI.Fabrik verantwortlich.

Bernd Vojanec
WITTENSTEIN SE, Corporate Research & Development, Igersheim, Deuschland;
bernd.vojanec@wittenstein.de

Bernd Vojanec absolvierte 2018 seinen Master in Management und IT an der Donau-Universität Krems, Niederösterreich. Aktuell ist er in der WITTENSTEIN Gruppe tätig, wo er sich auf die Entwicklung und Optimierung von I4.0-Systemen spezialisiert. Seine Forschung konzentriert sich hauptsächlich auf den Einsatz von industriellen digitalen Zwillingen und datengetriebenen Diensten im Kontext von Industrie 4.0, insbesondere in Bezug auf intelligente Produkte & Services, Standardisierung und der Verwaltungsschale.

Timo Markert
Resense GmbH, Geschäftsführung,
Klingenberg, Deutschland;
timo.markert@resense.io

Timo Markert erwarb einen M.Sc. in Wirtschaftsinformatik an der Hochschule Aalen und promoviert derzeit an der Universität Osnabrück im Bereich der Robotik und des maschinellen Lernens. Seine Forschung konzentriert sich auf die Analyse taktiler Sensordaten in robotischen Manipulationsaufgaben. Gleichzeitig ist er Geschäftsführer der Resense GmbH, einem Joint Venture der Firmen WIKA und WITTENSTEIN, das innovative Sensorlösungen entwickelt.

Andrè Kraft
BMW AG, Digitalisierung Shopfloor,
München, Deutschland;
andre.kraft@bmw.de

André Kraft absolvierte 2013 sein Diplom im Maschinenbau an der TU Dresden und ist gegenwärtig in der Forschung und Entwicklung intelligenter Robotik-Plattformen bei der BMW AG tätig. Sein Forschungsschwerpunkt umfasst die innovative Integration von KI in die Fabrikumgebung durch Roboter und ähnliche Systeme, mit einem besonderen Fokus auf kundenspezifische Anpassungsfähigkeit in flexiblen Produktionsprozessen.