Diplomarbeit

ILR-LFT DA 24-12

Entwicklung einer Methodik zur ganzheitlichen Simulation eines auf Verwaltungsschalen basierten dezentralen Flugzeugproduktionssystems

Zum
Erlangen des akademischen Grades

DIPLOMINGENIEUR
(Dipl.-Ing.)

Betreuer:	Dipl.-Ing. Michael Mauersberger
	Yassine Ghanjaoui M.Sc.
Verantwortlicher Hochschullehrer:	Prof. Dr. Johannes Markmiller
Tag der Einreichung:	19.01.2025
Erster Gutachter:	Prof. Dr. Johannes Markmiller
Zweiter Gutachter:	Dr.-Ing. Falk Hähnel

Selbständigkeitserklärung

Hiermit erkläre ich, dass ich die von mir am heutigen Tage dem Prüfungsausschuss der Fakultät Maschinenwesen eingereichte Diplomarbeit zum Thema

Entwicklung einer Methodik zur ganzheitlichen Simulation eines auf Verwaltungsschalen basierten dezentralen Flugzeugproduktionssystems

vollkommen selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Hamburg, 2025. Thomas Hanl

.....

Kurzfassung

Zur Bekämpfung des menschengemachten Klimawandels muss die Luftfahrtindustrie ihren Beitrag leisten. Sie allein hat einen Anteil von 5% am menschengemachten Klimawandel. Um die Transformation zu einer klimaneutralen Luftfahrt zu ermöglichen sind neue und disruptive Technologien wie Wasserstoffantriebe, Brennstoffzellen und elektrisches Fliegen notwendig. Diese wiederum erfordern die Entwicklung neuer und unerprobter Flugzeugkonzepte. Um dies zu ermöglichen, ist es wichtig, die Entwicklungsprozesse in der Luftfahrt zu beschleunigen. Eine Methode hierzu ist der digitale Faden, welcher es erlaubt alle Lebenszyklusphasen eines Flugzeugs miteinander zu verknüpfen. Dies ermöglicht wiederum die Bewertung von Entscheidungen bereits früh im Entwicklungsprozess. Dies erlaubt es unter anderem die Produktion frühzeitig in die Entwicklung miteinzubeziehen, um den Einfluss von Entwicklungsentscheidungen auf die Produzierbarkeit zu überprüfen. Das Konzept der Verwaltungsschale (Asset Administration Shell) (AAS) im Besonderen bietet die Möglichkeit der Umsetzung eines modernen, flexiblen und robusten Produktionssystems.

Um die Produzierbarkeit eines Flugzeugs im Rahmen eines auf AAS basierenden Produktionssystems zu bewerten sind Methoden notwendig mithilfe derer die Produktionsprozesse in frühen Entwicklungsstadien modelliert und simuliert werden können. Zu diesem Zweck wurde im Rahmen dieser Arbeit eine Methodik entwickelt zur Modellierung und Simulation von auf AAS basierenden Produktionssystemen. Diese Methodik wurde anschließend auf einen Anwendungsfall in der Flugzeugproduktion angewendet. Zu diesem Zweck wurde ein reales auf AAS basierendes Produktionssystem modelliert. Der gewählte Anwendungsfall wurde sowohl im realen Produktionssystem ausgeführt und im modellierten Produktionssystem simuliert. Die Genauigkeit der Simulation wurde durch einen Vergleich zwischen dem simulierten Produktionsprozess und dem real ausgeführten Produktionsprozess diskutiert.

Summary

The aviation industry must play its part in combating man-made climate change, as it alone contributes 5% to it. To enable the transformation to climate-neutral aviation, new and disruptive technologies such as hydrogen propulsion, fuel cells and electric flying are needed. These in turn require the development of new and untested aircraft concepts. To make this possible, it is important to accelerate the development processes in aviation. One method for this is the digital thread, which allows all life cycle phases of an aircraft to be linked together. This in turn enables decisions to be evaluated early on in the development process. Among other things, this allows production to be involved in development at an early stage in order to check the influence of development decisions on producibility. The concept of Asset Administration Shell(AAS) in particular offers the possibility of implementing a modern, flexible and robust production system.

In order to evaluate the producibility of an aircraft within a production system based on AAS, methods are required to model and simulate the production processes in

the early stages of development. For this purpose, a methodology for modelling and simulating production systems based on AAS was developed as part of this work. This methodology was then applied to a use case in aircraft production. For this purpose, a real production system based on AAS was modelled. The selected use case was both executed in the real production system and simulated in the modelled production system. The accuracy of the simulation was discussed by comparing the simulated production process with the real production process.

Inhaltsverzeichnis

Abkürzungen

AAS	Verwaltungsschale (Asset Administration Shell) . . f., 6, 8, 13–16, 21, 23–28, 30–36, 38 ff., 54, 64, 68, 71–75
API	Application Programming Interface 13, 20
CPPS	cyber-physisches Produktionssystem . . . . . 10, 12 f.
CPS	cyber-physisches System . . . . . 9 f.
CSV	Comma Seperated Values 20
DES	Diskrete Event Simulation 8, 21
DEU	Decoder-Encoder-Unit . . . . . 37 f., 64, 71
DLR	Deutsches Zentrum für Luft- und Raumfahrt e. V. 6, 13, 35
EU	Europäische Union 5
HTTP	Hypertext Transfer Protocol 14
I4.0	Industrie 4.0 . . . . . 8, 10, 15
IDTA	Industrial Digital Twin Association 13
IoT	Internet der Dinge . . . . . 9 f.
M2M	Machine-zu-Maschine 10
MBSE	modellbasiertes Systems Engineering . . . . . 16 f., 24
MMI	Mensch-Maschine-Interaktion 10
MQTT	MQ Telemetry Transport 14
OMG	Object Management Group 17
PPC	Produktionsplanung und -steuerung (Production Planning and Control) 6, 8, 10–13
SP	Dienst Anbieter (Service Provider) 15
SPS	Speicherprogrammierbare Steuerung 9, 47, 61
SR	Dienst Anfrager (Service Requester) 15
SysML	Systems Modeling Language . . . . . 17 ff., 21, 24 f., 73
UML	Unified Modeling Language . . . . . 16 ff., 20

Formelzeichen

Zeichen	Benennung	Einheit
M_G	Anzugsmoment	Nm
f	elastische Verlängerung der Schraube	mm
δ	elastische Nachgiebigkeit	$\frac{mm}{N}$
ϕ	Flankenwinkel	$^\circ$
P	Ganghöhe	mm
ρ'	Reibungswinkel	$^\circ$
F_S	Schraubenkraft	N
v_s	Schraubgeschwindigkeit	$\frac{mm}{s}$
U	Umdrehungszahl	$\frac{1}{s}$

1 Einleitung

Zur Bekämpfung des menschengemachten Klimawandels hat sich die Europäische Union (EU) dazu verpflichtet, im Rahmen des europäischen grünen Deals 2050 klimaneutral zu werden. Das Ziel ist hierbei, nachhaltiges Wirtschaftswachstum in der EU zu ermöglichen, welches voraussetzt, dass dieses innerhalb der Grenzen der Belastbarkeit unseres Planeten bleibt[1]. Die Luftfahrt allein hat einen Anteil von 5% am menschengemachten Klimawandel. Um ihren Beitrag zur Vision der Klimaneutralität zu leisten, steht sie vor großen Herausforderungen[2]. Neben dem Einsatz klimaneutraler Treibstoffe sowie operationeller Optimierungen ist die Entwicklung neuer und verbesserter Flugzeugkonzepte, ausgestattet mit modernen und innovativen Antriebsmethoden, notwendig. Die Entwicklung von solchen disruptiven Konzepten in der Luftfahrt, wie Brennstoffzellen, Wasserstoffantriebe und elektrisches Fliegen, stellt eine große Herausforderung und Risiko dar[3].

Ein Grund dafür ist, dass die Entwicklung von Flugzeugen ein sehr kosten- und zeitaufwendiger Prozess ist[4]. Entscheidungen, die in frühen Entwicklungsstadien getroffen werden, haben sehr große Auswirkungen auf die Eigenschaften eines Flugzeugs, sowohl in der Produktion als auch im Betrieb. Insgesamt werden über 70% der Gesamtkosten eines Systems in den frühen Entwicklungsphasen festgelegt[5]. Es ist somit unabdingbar, dass die Auswirkungen von Entscheidungen in frühen Entwicklungsphasen bereits frühzeitig untersucht werden. Die Komplexität von Flugzeugen steigt dabei ebenfalls stetig an und eine große Bandbreite an Flugzeugeigenschaften wie Aerodynamik, Akustik, Sicherheit, Komfort und Wirtschaftlichkeit müssen bereits während der Vorentwurfsphase untersucht werden[6]. Mithilfe digitaler Methoden können Entwicklungsprozesse schneller und mit einer erhöhten Qualität und Umfang durchgeführt werden[7]. Eine Möglichkeit, mithilfe digitaler Methoden die verschiedenen Aspekte der Flugzeugentwicklung zu verknüpfen, ist der digitale Faden[8]. Der digitale Faden ist die chronologische Speicherung der Daten eines oder mehrerer Objekte über mehrere Lebenszyklusphasen hinweg. Die einzelnen Lebenszyklusphasen werden somit verknüpft und eine Nachverfolgung über diese hinweg wird ermöglicht[9]. Diese Verknüpfung von Daten kann Optimierungspotenzial freisetzen durch die Integration verschiedener Methoden und Fachbereiche[10]. Die Auswirkungen von Designentscheidungen können somit frühzeitig auf den Gesamtentwurf bezogen untersucht werden[8].

Ein Teil des digitalen Fadens ist die Produktion. Die Produktion ist hierbei eine wichtige Lebenszyklusphase eines Flugzeugs. Die Leistung eines Flugzeugproduktionssystems ist stark abhängig vom spezifischen Flugzeugdesign. Es ist somit von großer Signifikanz, dass die Produktion bereits frühzeitig mitbetrachtet wird um den Einfluss von Designentscheidungen auf die Produzierbarkeit des Flugzeugs zu betrachten. Eine frühzeitige Betrachtung der Produktion erlaubt, Änderungen an der Spezifikation des Flugzeugs

vorzunehmen, um Anforderungen der Produktion gerecht zu werden, bevor solche Änderungen zu kostspielig werden[11]. Um dies zu ermöglichen, sind Modellierungs- und Simulationsmethoden vonnöten, welche bereits frühzeitig eingesetzt werden können, da mithilfe dieser der Einfluss von Entwicklungsentscheidungen auf die Produzierbarkeit analysiert werden kann.

Das Institut für Systemarchitekturen in der Luftfahrt des Deutschen Zentrums für Luft- und Raumfahrttechnik e. V. (DLR) untersucht daher verschiedene Produktionssysteme und wie diese in den digitalen Faden integriert werden können. Eine Methode zum Aufbau von Produktionssystemen ist der Einsatz der proaktiven AAS. Hierbei wird allen Produktionsmitteln ein digitales Abbild in Form einer AAS gegeben, welches Informationen über die Produktionsmittel während der gesamten Lebensdauer speichert. Eine AAS kann somit einen Teil des digitalen Fades darstellen. Die proaktive AAS verfügt zusätzlich über ein flexibles autonomes Verhalten[12]. Im Unterschied zur klassischen Produktion wird hier der Produktionsablauf nicht durch ein zentrales System geplant und ausgeführt. Stattdessen ermöglicht das autonome Verhalten der AAS die dezentrale, eigenständige Organisation der Produktion. Diese Fähigkeit erlaubt AAS, sich eigenständig an sich ändernde Situationen anzupassen und entsprechend festgelegter Ziele selbst zu optimieren[13]. Dies führt jedoch ebenfalls zu einem hochkomplexen Verhalten innerhalb des Produktionssystems, welches bereits in der Entwicklung mitbetrachtet werden muss.

Sowohl um die Produzierbarkeit eines Produktes frühzeitig zu bewerten, als auch um das Verhalten von AAS zu untersuchen, sind Simulationen notwendig. Diese müssen in frühen Phasen der Produktentwicklung einsetzbar sein und müssen das Produktionssystem inklusive der digitalen Bestandteile und der Kommunikationsinfrastruktur ganzheitlich abbilden.

Das Ziel dieser Arbeit ist die Entwicklung einer Methodik zur ganzheitlichen Simulation eines auf AAS basierten Flugzeugproduktionssystems, welches in frühen Phasen der Produktentwicklung zur Bewertung der Produzierbarkeit eingesetzt werden kann. Zu diesem Zweck wurde eine Methodik zur Modellierung von AAS entwickelt. Die Kommunikationsinfrastruktur, welche den Nachrichtenaustausch zwischen AAS ermöglicht, wurde ebenfalls modelliert. Es wurde ebenso eine Möglichkeit definiert, zur Anbindung von Modellen realer Ressourcen an die AAS. Die Methodik wurde anschließend anhand eines Anwendungsfalls validiert. Zu diesem Zweck wurden spezifische Produktionsmittel modelliert und an die AAS angebunden. Es wurde ein Produktionsprozess gewählt, der sowohl simulativ als auch in der Realität umgesetzt wurde. Anschließend wurden die Ergebnisse der Simulation mit den Messwerten aus der Realität verglichen, um die Methodik und die Genauigkeit der Modelle zu überprüfen.

In Kapitel 2 werden zuerst die theoretischen Grundlagen zum Verständnis der Arbeit erklärt. Hierbei wird der Stand der Technik in der Produktionsplanung und -steuerung (Production Planning and Control) (PPC) beschrieben, sowie die verwendeten Methoden und Programme vorgestellt. In Kapitel 3 wird die allgemeine Struktur der Methodik vorgestellt sowie die Modelle zur Abbildung der AAS und deren Verhalten erläutert. Der Aufbau des Modells der Kommunikationsinfrastruktur wird hier ebenfalls dargestellt. In Kapitel 4 wird die zuvor erklärte Methodik auf ein spezifisches

Produktionssystem angewendet. Zu diesem Zweck werden Modelle von individuellen Ressourcen erstellt sowie ein Produktionsprozess modelliert. Anschließend werden die Zeiten zur Produktion in der Realität und in der Simulation gemessen. In Kapitel 5 werden die Ergebnisse der zuvor durchgeführten Messungen aufgeführt und grafisch dargestellt. Kapitel 6 diskutiert die Ergebnisse und erläutert die Gründe und Ursachen für Abweichungen zwischen der Simulation und der Realität. Kapitel 7 bietet eine Zusammenfassung der Arbeit sowie einen Ausblick auf nächsten Schritte.

2 Grundlagen der digitalen Produktionsplanung

Im Rahmen dieser Arbeit wird eine Methodik entwickelt, um Produktionssysteme, welche auf AAS basieren, zu modellieren und zu simulieren. In diesem Kapitel wird die Industrie 4.0 (I4.0) als ein Treiber von Veränderungen in der Produktion vorgestellt. Die PPC ist ein Teilbereich der Produktion, in dem durch den Einsatz von modernen Informations- und Kommunikationstechnologien im Rahmen der I4.0 neue Konzepte möglich sind. Die AAS wird vorgestellt als eine Möglichkeit, um moderne PPC-Systeme umzusetzen. Mithilfe des „Systems Engineering“ ist es möglich, Produktionssysteme zu modellieren, um ein größeres Verständnis für das System und dessen Verhalten zu gewinnen. In diesem Zusammenhang ist Diskrete Event Simulation (DES) eine Methode, um Simulationen in der Produktion umzusetzen. Es werden ebenfalls der UR10e-Roboter sowie Schraubverbindungen vorgestellt, da diese im Anwendungsfall dieser Arbeit zum Einsatz kommen. Abschließend wird die Forschungslücke untersucht und definiert.

2.1 Herkunft und Konzepte der Industrie 4.0

Der Begriff I4.0 beschreibt eine momentan ablaufende vierte industrielle Revolution und umfasst Konzepte und Methoden, wie sich Unternehmen an diese Entwicklung anpassen können[14].

Im Rahmen der I4.0 wird die Geschichte der industrialisierten Produktion in vier aufeinanderfolgende Revolutionen strukturiert. Diese unterscheiden sich sowohl im Einsatz unterschiedlicher Technologien als auch in der Produktlandschaft. Zwei relevante Eigenschaften der Produktlandschaft sind hierbei die Menge an Produktvarianten sowie die Stückzahl je Produktvariante[15]. Die Entwicklung dieser beiden Werte ist in Abbildung 2.1 im Lauf der Zeit dargestellt.

Die erste industrielle Revolution begann um das Jahr 1750. Sie ist durch den erstmaligen Einsatz von durch Dampfmaschinen angetriebenen Arbeits- und Kraftmaschinen geprägt. Dies hatte einen enormen Anstieg der Produktivität in der Herstellung von Grundversorgungsgütern zur Folge. Die unmittelbaren Folgen dieser Entwicklung waren der Beginn des Rückgangs der handwerklichen Produktion, welche sich durch eine geringe produzierte Stückzahl aber hohe Variantenvielfalt auszeichnet[15].

Um 1870 folgte die zweite industrielle Revolution. Diese ist maßgeblich geprägt durch den Übergang zur arbeitsteiligen Massenproduktion unter Anwendung von elektrischer

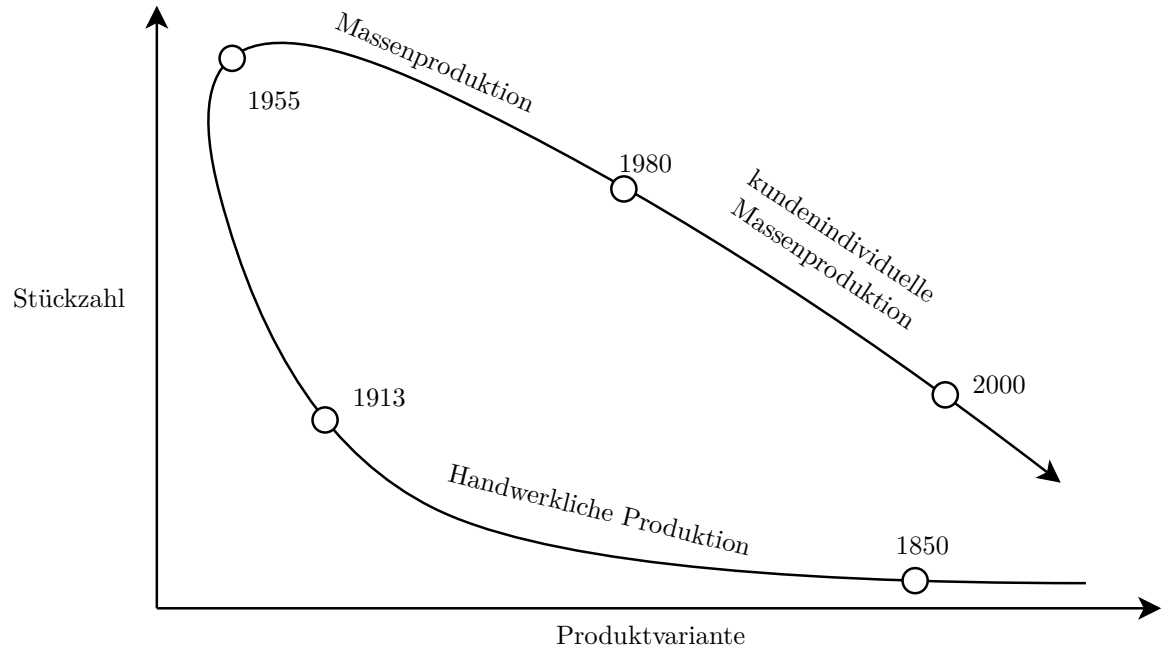


Abbildung 2.1: Entwicklung von Produktvielfalt und Stückzahl pro Produkt von 1850 bis heute nach [15]

Energie. Die produzierte Stückzahl explodierte infolge dieser Entwicklung, während die Menge an Produktvarianten abnahm. Im Besonderen kam es zur Entwicklung des Fließbands durch Henry Ford sowie die wissenschaftliche Betriebsführung durch Frederic W. Taylor[15].

Anfang der 1960er Jahre kam es anschließend zur dritten industriellen Revolution. Diese wurde primär getrieben durch die Elektronik und Informations- und Kommunikationstechnologien[15]. Durch den Einsatz dieser Technologien sowie der speicherprogrammierbaren Steuerung (SPS) war eine automatisierungsgetriebene Rationalisierung der Produktion möglich, was wiederum die kundenindividuelle Massenproduktion ermöglichte[16]. Dies führte zu einer Umkehrung des bisherigen Trends und erstmals seit der ersten industriellen Revolution kam es zu einer Senkung der produzierten Stückzahl und einer Steigerung der Variantenvielfalt[15].

Die vierte industrielle Revolution beschreibt die sich anbahnenden Umwälzungen in der industriellen Produktion infolge der Entwicklung der modernen digitalen Informations- und Kommunikationstechnologien. Diese ermöglichen komplett neue Ansätze, wie das Internet der Dinge (IoT), welches Produktionsmittel über das Internet untereinander verknüpft, sowie der Einsatz cyber-physischer Systeme (CPS), welche das Ziel einer Verschmelzung von realer und digitaler Welt verfolgen. Die vierte industrielle Revolution ist unter anderem durch sich ändernde Anforderungen in der Produktionsumgebung getrieben. Die Komplexität von individuellen Produkten nimmt zu, während die individuelle Stückzahl weiter abnimmt. Dies erfordert von Unternehmen, zunehmend flexibler und wandlungsfähiger zu werden, um sich an die Herausforderungen der neuen wirtschaftlichen Umgebung anpassen zu können[15].

Der Begriff I4.0 umfasst somit eine große Zahl von Konzepten und Technologien, welche im Zuge der vierten industriellen Revolution zur Anwendung kommen. Den Kern bildet dabei der Ansatz der intelligenten Fabrik, welcher das Ziel verfolgt, die steigende Komplexität in der Produktion für Unternehmen beherrschbar zu machen. Dies wird erzielt durch die Implementierung intelligenter Produkte und Ressourcen, welche im direkten Kontakt miteinander und mit dem Menschen über das IoT stehen. Das Ziel dabei ist es, sowohl die Effizienz als auch die Robustheit in solch einer Fabrik zu erhöhen, indem die individuellen Bestandteile des Produktionssystems mit der Fähigkeit zur Selbstoptimierung ausgestattet werden[16]. Zum Verständnis der Konzepte der I4.0 schlägt Roth [16] eine Aufteilung der I4.0 in 3 Stufen vor.

Bei Stufe 1 handelt es sich um cyber-physische Systeme CPS. Ein cyber-physisches System beschreibt ein System, in welchem Komponenten der realen Welt mit Komponenten der digitalen Welt verschmelzen[17]. Jedes reale Objekt verfügt dabei über eine digitale Identität innerhalb eines komplexen digitalen Netzwerks. Jedes Objekt besitzt des Weiteren die Fähigkeit, Daten zu sammeln, zu verarbeiten sowie Entscheidungen auf der Basis dieser Daten zu treffen. Sie sind somit mit einem gewissen Grad an Intelligenz und Autonomie ausgestattet. Die Technologie des IoT ermöglicht allen Objekten die Kommunikation über das Internet. Sämtliche Objekte können dementsprechend über das Internet angesprochen und relevante Daten abgerufen werden[16].

Als zweite Stufe der I4.0 definiert Roth das cyber-physische Produktionssystem (CPPS). In einem cyber-physischen Produktionssystem sind Produktionsressourcen in der Lage, Produktionsprozesse dezentral und autonom zu steuern. Produktionsressourcen entscheiden selbst, welche Tätigkeiten sie zu welchem Zeitpunkt durchführen. Es existiert keine zentrale Steuerung der Produktion und Produktionsprozesse werden durch die eigenständige Kooperation intelligenter Ressourcen umgesetzt. Dies bedarf der Definition einer allgemeinen Sprache für die Machine-zu-Machine (M2M) Kommunikation. Zur Einbindung des Menschen ist es ebenfalls notwendig, geeignete Technologien für die Mensch-Maschine-Interaktion (MMI) zu entwickeln[16].

Bei Stufe 3 handelt es sich final um die I4.0. Diese ergänzt die technischen Konzepte aus den vorherigen Stufen mit einem Umdenken auf der Betriebsführungsebene und verknüpft diese mit neuen Geschäftsmodellen und -prozessen in einer gemeinsamen Zukunftsvision[16].

Im Besonderen CPPS werden eine wichtige Rolle in der Produktion in der I4.0 spielen[17]. Diese neuen Konzepte erfordern ein Umdenken in vielen Bereichen der Produktion, wie zum Beispiel in der PPC.

2.2 Produktionsplanung und -steuerung

Die PPC befasst sich mit der Planung aller Produktionsprozesse sowie der anschließenden Steuerung dieser. Dies umfasst die Planung des Materialbedarfs, das Nachfragemanagement, die Planung vorhandener Kapazitäten sowie die Planung und Sequenzierung

einzelner Produktionschritte. Sie verfolgt dabei primär das Ziel der Definition von Produktionsplänen, welche über eine optimale Auslastung von Arbeitern, Maschinen und Materialien verfügen sowie die Methoden zur Umsetzung dieser Pläne[18].

In der klassischen PPC werden Aktivitäten detailliert geplant, koordiniert und überwacht. Hierbei wird die Produktion in Form eines Prozessplans umgesetzt. Innerhalb dieses ist festgelegt, welche Ressource zu welchem Zeitpunkt welchen Produktionsprozess ausführt. Dies umfasst die detaillierte Sequenzierung der einzelnen Prozessschritte und die anschließende Terminierung dieser[19].

Ein klassisches Modell der Prozessplanung ist in Abbildung 2.2 abgebildet. Das Ergebnis hier ist die Erstellung des Prozessplans. Zu diesem Zweck müssen eine Reihe an relevanten Informationen als Eingabedaten zusammengetragen werden. Diese umfassen die technischen Daten des zu produzierenden Produkts, wie das spezifische Design, Qualitätsanforderungen und Materialdaten. Ebenfalls enthalten sind Informationen über die Menge und Fähigkeiten der vorhandenen Produktionsressourcen sowie Informationen über den Produktionstypen, wie zum Beispiel die zu produzierende Stückzahl. Diese Daten werden im Rahmen der Prozessplanung verarbeitet und aus diesen wird schließlich der Prozessplan erstellt[19]. Die Prozessplanung erfolgt somit erst nach dem Abschluss der Entwicklung eines Produkts.

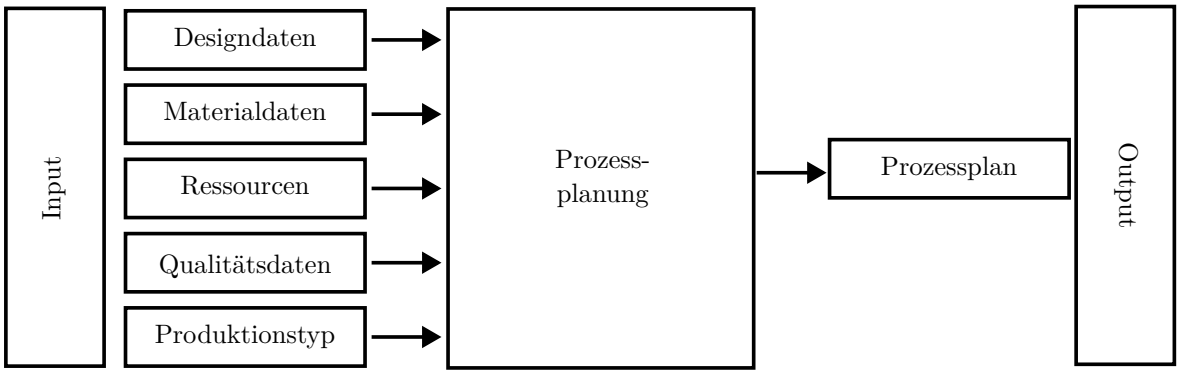


Abbildung 2.2: Ein Modell der Prozessplanung nach [19]

Der letzte Schritt innerhalb der PPC ist die Sequenzierung der Produktionsschritte. Hierbei werden die individuellen Produktionsschritte in ihrer endgültigen Reihenfolge aufgelistet und entsprechenden Ressourcen zugeordnet. Nach der Überprüfung und eventueller Neuplanung des Produktionsplans muss dieser anschließend lediglich durch die Ressourcen ausgeführt werden[20].

Die Steuerungsstruktur eines PPC kann im Rahmen des ANSI/ISA 95 Standards[21] betrachtet werden. Dieser Text beschreibt die Integration von Unternehmens- und Kontrollsystemen. Der Aufbau ist in mehrere Ebenen unterteilt, welche in einer Pyramide angeordnet sind. Dies ist in Abbildung 2.3 zu sehen. Die unterste Ebene, Ebene 0, ist die physikalische Ebene, in der die tatsächlichen physikalischen Produktionsprozesse ausgeführt werden. Ebene 1 umfasst intelligente Geräte, welche die physikalischen Prozesse manipulieren oder durch Sensoren erfassen. In Ebene 2 sind Steuerungssysteme zu verorten, welche die Produktionsläufe steuern. Die Fertigungsbetriebssysteme in Ebene 3 weisen die individuellen Produktionsschritte zentral den jeweiligen Ressourcen

zu. Innerhalb dieser Ebene ist die klassische Sequenzierung der Produktionsprozesse zu verorten. Ebene 4 umfasst schließlich die Unternehmenslogik. Dabei handelt es sich um alle Prozesse im Unternehmen, welche kaufmännischer Natur sind. Die Kommunikation innerhalb der Pyramide findet stets lediglich von einer vorherigen Ebene zur nächsten statt[20].

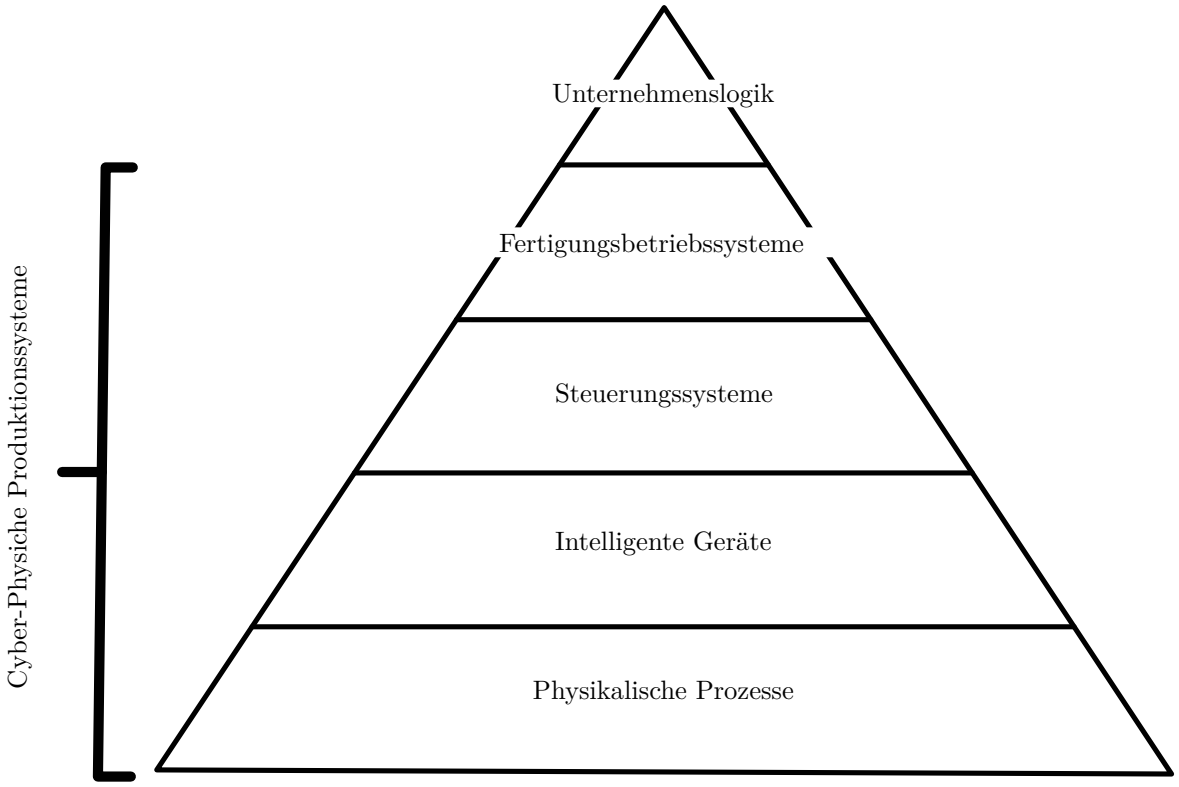


Abbildung 2.3: ISA 95 Pyramide nach [20]

Die einzelnen Ebenen verfügen zusätzlich über unterschiedliche Zeithorizonte, in denen sie ihre Funktionen ausführen. Die relevanten Zeithorizonte sinken dabei, je tiefer die jeweilige Ebene in der Pyramide angeordnet ist. Während auf der Ebene der Unternehmenslogik Entscheidungen mit Zeithorizonten von Tagen bis hin zu Monaten getroffen werden, sind die Zeithorizonte auf der Steuerungsebene im Stundenbereich. Die Zeithorizonte in der Ebene der intelligenten Geräte befinden sich im Sekundenbereich und auf der Ebene der physikalischen Prozesse sogar im Millisekundenbereich[22].

Der Einsatz von CPPS hat einen unmittelbaren Einfluss darauf, wie Entscheidungen in einem PPC-System getroffen werden. Ein CPPS ist in der Lage, mehrere Ebenen der ISA 95 ineinander zu integrieren. Dies ist in Abbildung 2.3 dargestellt. Das CPPS umfasst die physikalische Ebene, die intelligenten Geräte, die Steuerungssysteme sowie die Fertigungsbetriebssysteme[20].

Es gibt zwei Ansätze zur PPC in CPPS. Zum einen ist dies der Einsatz von in CPPS gesammelten Echtzeitdaten zur Neuplanung von Produktionsprozessen im Laufe der Produktion. Dabei handelt es sich um einen klassischen Ansatz der PPC, welcher durch den Einsatz der neuen Technologien der CPPS weiterentwickelt wird. Der Fokus liegt hierbei auf der Optimierung der Produktionsprozesse[20].

Der zweite Ansatz ist die Dezentralisierung der Entscheidungsfindung. Dieser Ansatz befasst sich mit der Entwicklung neuer Architekturen und Methoden zur PPC. Es gibt eine Reihe von Möglichkeiten, wie eine solche Dezentralisierung der Entscheidungsfindung erreicht werden kann. Hier hervorzuheben sind zum einen agentenbasierte Verhandlungen, in denen individuelle Agenten eigene Ziele verfolgen, zum anderen sind dies ausschreibungsbasierte Verfahren, in denen individuelle Produktagenten Angebote von Ressourcenagenten erhalten[20]. Dieser Ansatz erfordert die Entwicklung neuer Methoden und Technologien, welche die Dezentralisierung der Entscheidungsfindung unterstützen. Eine Methodik, um dies zu erreichen, ist die AAS.

2.3 Verwaltungsschale

Ein PPC kann in einem CPPS über verschiedene Methoden umgesetzt werden. Eine davon ist die AAS. Eine AAS ist eine digitale Repräsentation eines „Assets“ (dt. Asset). Ein Asset kann dabei ein physischer Gegenstand sein, aber auch Programme, komplexe Systeme oder Personen beschreiben[23]. Die AAS kann somit als eine Art digitaler Zwilling angesehen werden, geht jedoch über diese Definition hinaus durch den Einsatz proaktiver Bestandteile, welche über ein eigenständiges Verhalten verfügen[12]. Es werden dabei drei Stufen von AAS unterschieden, welche aufeinander aufbauen. Die erste Stufe ist die passive dateibasierte Verwaltungsschale. Eine AAS dieser Stufe besteht lediglich aus einer serialisierten Datei, welche die Daten der AAS enthält. Zum Informationsaustausch wird diese Datei mit Kommunikationspartnern geteilt. Die AAS der zweiten Stufe ist die passive serverbasierte Verwaltungsschale. Diese stellt neben der serialisierten Dateiform die in ihr enthaltenen Daten einzeln über eine Kommunikationsschnittstelle wie ein Application Programming Interface (API) zur Verfügung. Die AAS der dritten Stufe ist die proaktive Verwaltungsschale. Diese agiert autonom in der jeweiligen Produktionsumgebung und ist in der Lage, eigenständig Entscheidungen zu treffen. Sie tritt eigenständig in Verhandlungen mit anderen AAS ein, um ein bestimmtes Ziel zu erreichen[12]. Weiterführend wird in dieser Arbeit mit dem Begriff AAS immer eine AAS der dritten Stufe beschrieben.

Ein standardisiertes Meta-Modell der AAS wird durch die Industrial Digital Twin Association (IDTA) bereitgestellt[24]. Die IDTA definiert, dass eine AAS aus einem „Header“ (dt. Kopf) sowie einem „Body“ (dt. Körper) besteht. Dabei unterscheidet die AAS nicht explizit zwischen Kopf und Körper, legt aber fest, dass der Kopf Informationen für die Identifikation der AAS umfasst, während der Körper spezifische Information über das zu beschreibende Asset enthält[24]. Die AAS setzt sich zusammen aus individuellen Teilmodellen. Diese beschreiben jeweils einen klar getrennten und von anderen Teilmodellen unabhängigen Aspekt. Jedes Teilmodell verfügt über einen eindeutigen Identifikator. Komplexe Aspekte können durch mehrere Teilmodelle gemeinsam beschrieben werden[23]. Teilmodelle können inhaltliche Aspekte wie Zertifikationen und Dokumentationen enthalten, aber auch funktionale Aspekte wie die Kommunikation und Ausführung von Prozessen beschreiben[12].

Im Rahmen dieser Arbeit wurde eine AAS verwendet, welche vom DLR-Institut für Instandhaltung und Modifikation gemeinsam mit der Otto-von-Guericke-Universität

Magdeburg entwickelt wurde. Diese AAS besteht neben der in JSON serialisierten AAS aus einem AAS-Server. Diese Struktur ist in Abbildung 2.4 dargestellt. Der AAS-Server besteht aus mehreren Modulen, welche spezifische Aufgaben in der AAS übernehmen. Diese umfassen unter anderem die Einlesung der serialisierten AAS-Datei und die Bereitstellung eines internen Datenbusses zur Nachrichtenverwaltung. Die Kommunikation nach außen wird über Hypertext Transfer Protocol (HTTP)- und MQ Telemetry Transport (MQTT)-Schnittstellen ermöglicht. Bei diesen handelt es sich um standardisierte Internetkommunikationsprotokolle. Die Kommunikation zum Asset erfolgt über Sockets. Besonders hervorzuheben sind hier die Zustandsmaschinen. Die Zustandsmaschinen setzen funktionale Aspekte der AAS um. Sie verfügen über eine Menge an Zuständen sowie Transitionen zwischen diesen. Ein Anfangszustand ist ebenfalls definiert. Ein Verweis auf die jeweilige Zustandsmaschine sowie ihr Anfangszustand sind in der serialisierten AAS-Datei enthalten. Jeder individuelle Zustand wird wiederum durch ein Python-Skript umgesetzt[12].

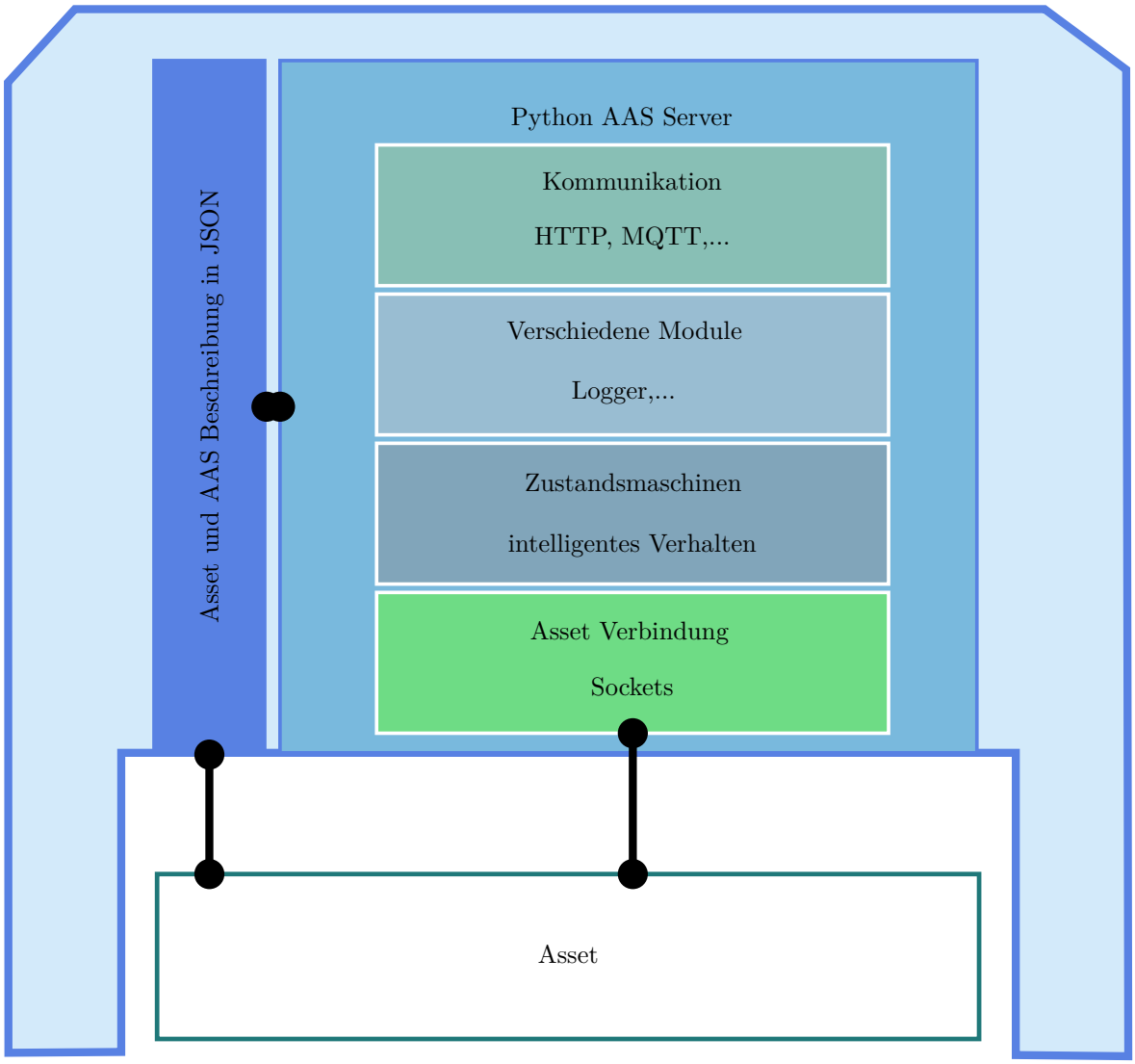


Abbildung 2.4: Schematischer Aufbau der proaktiven AAS nach [12]

Die in dieser Arbeit verwendete AAS folgt, zur Ermöglichung der Verhandlung zwischen

I4.0-Komponenten, der Definition des Ausschreibungsverfahrens in der I4.0 gemäß der VDI/VDE-Richtlinie 2193 Blatt 2[25]. Dieses Ausschreibungsverfahren ist in Abbildung 2.5 dargestellt. Das Verfahren folgt einem spezifischen Ablauf. Eine I4.0-Komponente A veröffentlicht eine Ausschreibung. Dabei handelt es sich um die Aufforderung zur Abgabe eines Angebots für eine spezifische Leistung, solange dies einem entsprechenden Anbieter möglich ist. Diese Ausschreibung kann zeitlich begrenzt sein. Als Antwort auf die Ausschreibung und unter der Voraussetzung, dass die Erfüllung der gewünschten Leistung möglich ist, versendet I4.0-Komponente B ein Angebot. Ein Angebot kann ebenso zeitlich befristet sein und ist für den Anbieter bindend. Die I4.0-Komponente A kann das Angebot annehmen, welches sie als das beste erachtet. Dies führt zu einem Vertragsabschluss. Die I4.0-Komponente B ist nun zur Erbringung der entsprechenden Leistung verpflichtet[25].

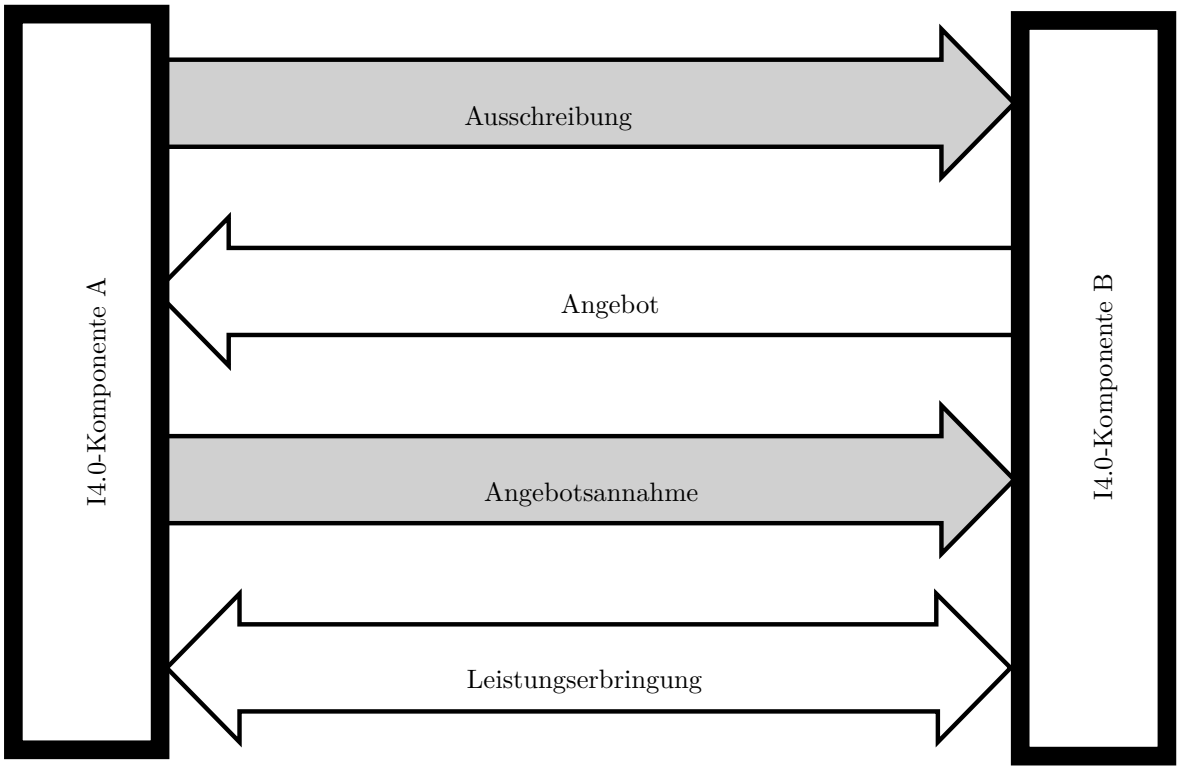


Abbildung 2.5: Schematischer Aufbau der ProAktiven AAS nach [12]

Die in dieser Arbeit verwendete AAS nutzt zwei Teilmodelle zur Umsetzung dieses Verfahrens. Diese sind der Service Provider (dt. Dienstanbieter, SP) und der Service Requester (dt. Dienstanfrager, SR). Die individuellen Schritte des Ausschreibungsverfahrens sind mithilfe von Zustandsmaschinen innerhalb dieser Teilmodelle umgesetzt[12].

Zur Beurteilung von Ausschreibungen nutzt die hier verwendete AAS das Capability-Skill Modell (dt. Fähigkeit-Kompetenz Modell)[12]. Eine Fähigkeit ist „eine implementierungsunabhängige Spezifikation einer Funktion in der industriellen Produktion, um einen Effekt in der physischen oder virtuellen Welt zu erzielen“[26]. Eine Fähigkeit ist somit eine abstrakte Modellierung einer industriellen Tätigkeit. Ein Beispiel einer

Fähigkeit könnte sein „ein Loch mit bestimmter Breite und Tiefe in ein bestimmtes Material zu bohren“[26].

Die Kompetenz ist „eine ausführbare Implementierung einer Funktion, welche durch eine Fähigkeit spezifiziert ist“[26]. Im Vergleich zur Fähigkeit ist die Kompetenz somit eine spezifische Methode, um die jeweilige Fähigkeit zu ermöglichen. Eine Kompetenz setzt eine Fähigkeit um. Die Trennung von Fähigkeit und Kompetenz entkoppelt die Beschreibung einer Funktion von ihrer spezifischen Implementierung, was mehr Flexibilität in der Entwicklung von spezifischen Implementierungen erlaubt[26].

Die Fähigkeit und Kompetenz sind in den jeweiligen Teilmodellen der AAS definiert und werden im Rahmen des Ausschreibungsverfahrens durch die Zustandsmaschinen ausgelesen. Auf ihrer Basis wird die Durchführbarkeit überprüft und ein Angebot für die entsprechende Leistung erstellt. Nach der Auftragserteilung wird die jeweilige Kompetenz durch die AAS ausgeführt[12].

Aufgrund der Struktur von AAS in Form von Teilmodellen und einem auf der Unified Modeling Language (UML) basierenden Metamodell[25] eignen sich die Methoden des modellbasierten Systems Engineering (MBSE) zur weiteren Untersuchung von AAS[27][28].

2.4 modellbasiertes Systems Engineering

Der Begriff „Systems Engineering“ lässt sich am ehesten als Systemtechnik oder Systementwicklung übersetzen. Diese deutschen Begriffe decken jedoch nicht alles ab, was mit „Systems Engineering“ ausgedrückt wird. Das „Systems Engineering“ wird durch den International Council on Systems Engineering definiert als: „ein transdisziplinärer und integrativer Ansatz zur erfolgreichen Realisierung, Nutzung und Stilllegung entwickelter Systeme, unter Anwendung systemischer Prinzipien und Konzepte sowie wissenschaftlicher, technologischer und managerieller Methoden“[29]. Nach Alt[30] kann „Systems Engineering“ in einer vereinfachten Definition beschrieben werden. Entsprechend dieser besteht „Systems Engineering“ aus drei grundlegenden Teilen: Systemarchitektur, Systemanforderungen und Systemverhalten. Diese drei Teile ergeben, wie in Abbildung 2.6 dargestellt, gemeinsam das „Systems Engineering“. Die Systemarchitektur beschreibt die Struktur des Systems. Sie definiert die Komponenten eines Systems sowie, wie diese miteinander verknüpft sind. Schnittstellen zwischen den Systemkomponenten sowie Schnittstellen zu den Systemgrenzen sind hier ebenfalls definiert. Es handelt sich hierbei um eine rein statische Sicht auf das System, aus welcher nicht ersichtlich wird, welche Daten und Materialien wann über die jeweiligen Schnittstellen ausgetauscht werden. Bei den Anforderungen handelt es sich um die Definition, was das zu entwickelnde System können und leisten muss. Hierbei wird unterschieden zwischen funktionalen Anforderungen, welche das Verhalten des Systems spezifizieren, sowie nichtfunktionalen Anforderungen, welche mit dem Verhalten des Systems nur indirekt zusammenhängen. Der letzte Teil ist das Systemverhalten. Hier wird formal das Verhalten des Systems so definiert, dass daraus Arbeitsprodukte und Informationen

abgeleitet werden können[30]. Im Rahmen dieser Arbeit wird primär die Systemarchitektur sowie das Systemverhalten betrachtet.

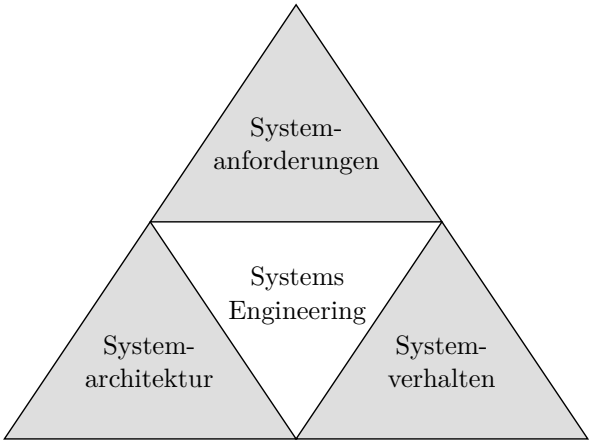


Abbildung 2.6: Die drei Bestandteile des „Systems Engineering“ nach [30]

Das modellbasierte Systems Engineering (MBSE) unterscheidet sich vom klassischen dokumentenbasierten „Systems Engineering“ dadurch, dass bei ihm die Systembeschreibung durch den Einsatz von Modellen anstatt von Texten und Dokumenten erfolgt. Ein Modell ist dabei eine abstrakte Repräsentation der Realität[30].

MBSE kommt aus unterschiedlichen Gründen zur Anwendung. Der üblichste Grund ist die Modellierung eines Problems, um dieses auf einem abstrakten Niveau verstehen zu können. Andere typische Zwecke sind die domänenspezifische Modellierung sowie die Umsetzung von Simulationen[31]. Die Definition eines Modells erfordert den Einsatz einer geeigneten Modellierungssprache[30]. Die am meisten verwendete Sprache ist die Systems Modeling Language (SysML)[31].

2.4.1 Systems Modeling Language

Die SysML ist eine grafische Sprache zur Modellierung von technischen Systemen. Es handelt sich um einen offiziellen Standard der Object Management Group (OMG). SysML ermöglicht es, die Struktur, das Verhalten sowie die Anforderungen eines Systems formal zu beschreiben. SysML wurde auf der Basis von UML entwickelt. Es handelt sich bei SysML technisch um ein UML-Profil, welches das UML-Metamodell auf eine technische Anwendung anpasst und gewisse Teile ausblendet[30].

Da es sich bei UML um eine Sprache handelt, die primär im Softwareentwicklungsbereich eingesetzt wird, wurden in SysML eine Reihe von Konzepten abgeändert und umbenannt, um sie auf die Ingenieurswissenschaften anwendbar zu machen. Das Konzept der „Class“ (dt. Klasse) wurde angepasst zum „Block“ (dt. Block). Blöcke sind das oberste Element SysMLs. Ihm werden untergeordnete Bestandteile zugewiesen, welche „Properties“ (dt. Eigenschaften) oder „Block-Properties“ (dt. Blockeigenschaften) genannt werden. Die äußerste Systemgrenze wird als Block modelliert und die jeweiligen Systemkomponenten als Blockeigenschaften. Blockeigenschaften können weitere Eigenschaften und Blockeigenschaften als Unterelemente enthalten[30].

SysML ist eine graphische Modellierungssprache. Sie trennt die Konzepte Modell und Sicht eindeutig voneinander. Sicht beschreibt hier, dass die abgebildeten Modellelemente nicht die Gesamtheit des Modells zeigen müssen. Die Sicht ist lediglich ein Ausschnitt einiger Modellelemente aus dem Gesamtmodell. Dies ermöglicht es, Modellelemente in mehreren Sichten darzustellen, um unterschiedliche Aspekte des Systems in entsprechenden Sichten zu verdeutlichen[30].

SysML setzt sich aus insgesamt neun Arten von Diagrammen zusammen. Ein Diagramm bietet eine Sicht auf das Modell und ist die Schnittstelle zwischen Modell und Anwender. Die Diagrammarten sind in Abbildung 2.7 dargestellt. Die strukturellen Diagramme bilden statische Aspekte des Systems ab. Sie werden zur Modellierung der Systemarchitektur eingesetzt. Die Verhaltensdiagramme bilden wiederum das Systemverhalten ab. Das Anforderungsdiagramm ist in SysML im Vergleich zu UML ergänzt worden. Es erlaubt, Aspekte im Bezug auf die Systemanforderungen darzustellen[30]. Für diese Arbeit relevant sind das Blockdefinitionsdiagramm, das interne Blockdiagramm und das Zusicherungsdiagramm im Bereich der strukturellen Diagramme und das Aktivitätsdiagramm und Zustandsdiagramm im Bereich der Verhaltensdiagramme.

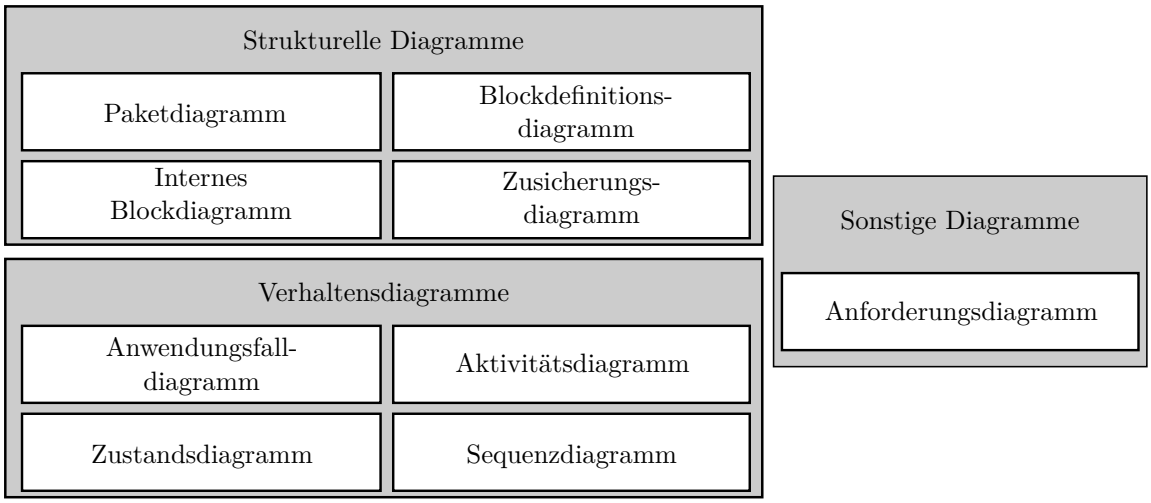


Abbildung 2.7: SysML Diagramme nach [30]

Das Blockdefinitionsdiagramm dient der Definition von Blöcken in SysML. Es funktioniert dabei ähnlich wie einem Klassendiagramm in UML. Beziehungen zwischen Blöcken können in einem Blockdefinitionsdiagramm dargestellt werden. Diese Beziehungen sind die Vererbungs- und die Dekompositionsbeziehung. Dekompositionsbeziehungen lassen sich ebenfalls Kardinalitäten zuweisen. Diese Kardinalitäten definieren, wie oft ein Element in einem anderen enthalten ist[30].

Das interne Blockdiagramm kommt zur statischen Definition der Struktur eines Systems zum Einsatz. Systemkomponenten werden als Instanzen von Blöcken dargestellt. Diese Blöcke verfügen über Schnittstellen in Form von „Ports“ (dt. Port). Diese Ports sind wiederum über Konnektoren miteinander verbunden. Die Ports und entsprechenden Konnektoren ermöglichen einen Daten- und Materialfluss zwischen den individuellen Systemkomponenten[30].

Das parametrische Zusicherungsdiagramm verknüpft Eigenschaften mit „Constraints“ (dt. Zusicherung). Diese Zusicherungen umfassen parametrische mathematische Gleichungen, welche die Eigenschaften verknüpfen und einschränken. Die Zusicherungen sind als „Constraint Blocks“ (dt. Zusicherungsblöcke) definiert, welche innerhalb des Diagramms als „Constraint Property“-Elemente (dt. Zusicherungseigenschaft-Elemente) instanziert werden. Die Zusicherungen und Eigenschaften werden über Konnektoren verbunden. Mithilfe der Zusicherungsblöcke lassen sich beliebig komplexe Gleichungen definieren. Der Zweck des parametrischen Zusicherungsdiagramms ist die mathematische Beschreibung der Zusammenhänge in einem System, mit dem Zweck, Eingangsparameter zu variieren[30].

Das Aktivitätsdiagramm dient zur Modellierung des Verhaltens des Systems. Es definiert Aktivitäten und Aktionen sowie die Abfolge derselben. Aktivitäten sind wiederverwendbare Verhaltenselemente und beschreiben das Verhalten von Aktionen. Aktionen sind durch Kontrollflüsse, die als gestrichelte Pfeile dargestellt werden, miteinander verbunden. Der Kontrollfluss definiert den Ablauf der Aktionen. Parallel zu diesem existiert der Objektfluss, welcher als durchgezogene Linie dargestellt wird. Dieser kann zusätzlich Daten und Material zwischen Aktionen übertragen. Das Aktivitätsdiagramm folgt dabei dem Tokenkonzept. An einem Startknoten wird ein Token erzeugt. Dieser wandert entlang der gegebenen Kontroll- und Objektflüsse. Erreicht das Token eine Aktion, wird diese ausgeführt und danach entlang der verbundenen Kontroll- und Objektflüsse weitergereicht. Neben Aktionen gibt es ebenfalls Entscheidungsknoten. Diese erlauben es, Kontroll- und Objektflüsse mit Bedingungen zu versehen. Das Token folgt einem Kontroll- oder Objektfluss nur, wenn dessen Bedingung erfüllt ist. Es ist ebenfalls möglich, mithilfe einer „Fork“ (dt. Gabelung) zusätzliche parallele Token zu erzeugen, um gleichzeitig ablaufende Aktionen zu starten. Mithilfe des „Join“-Elements (dt. Zusammenführungs-Element) können diese parallelen Token wieder zu einem Token synchronisiert werden. An einem Endknoten wird das Token zerstört. Dies bedeutet ein Ende des Ablaufs, es werden keine weiteren Aktionen ausgeführt. Neben diesen normalen Elementen existieren ebenfalls die „SendSignal“ (dt. Sende Signal)- und „AcceptEvent“ (dt. akzeptiere Ereigniss)-Elemente. Diese ermöglichen es, Signale auszusenden und auf bestimmte Events zu warten[30].

Das Zustandsdiagramm ist ein Verhaltensdiagramm. Es strukturiert das Verhalten eines Systems in individuellen Zuständen, welche über Transitionen miteinander verbunden sind. Durch das Folgen einer Transition ist es möglich, von einem Zustand in einen anderen zu gelangen. Transitionen können mit Bedingungen belegt werden, so dass ein Übergang nur bei Erfüllung einer Bedingung oder der Ausführung eines Ereignisses durchgeführt wird[30].

Zur Modellierung mit SysML werden eine Reihe von Programmen verwendet[31]. Diese Arbeit verwendet den „Cameo Systems Modeler“.

2.4.2 Cameo Systems Modeler

Zur Modellierung wurde in dieser Arbeit der „Cameo Systems Modeler“ verwendet. Dieser erlaubt die Modellierung von Systemen mithilfe von SysML. Des Weiteren bie-

tet er über das „Cameo Simulation Toolkit“ die Möglichkeit das modellierte System zu simulieren. Dies wird ermöglicht durch die Ausführung der Zustands- und Aktivitätsdiagramme mithilfe des „Cameo Simulation Toolkit“s[32]. Der „Cameo Systems Modeler“ und das „Cameo Simulation Toolkit“ werden weiterführend vereinfacht als Cameo bezeichnet.

Cameo ermöglicht die Simulation von Systemen durch vier Simulations-Engines. Diese beziehen sich jeweils auf Aktivitäts-, Zustands-, Sequenz- und Zusicherungsdiagramme. Blöcken kann in Cameo ein einziges Verhalten zugewiesen werden, welches bei der Simulation des Blocks ausgeführt wird[32].

Cameo verfügt für zeitsensitive Simulationen über drei Methoden der Zeitmessung. Die Standard-Simulationsuhr misst die Zeit kontinuierlich während der Ausführung einer Simulation. Die interne Simulationsuhr steigt ausschließlich an durch im Modell definierte Zeiten und Dauern. Die Zeit schreitet in der Simulation somit nur voran, wenn ein Verhaltenselement mit einem entsprechenden „Duration Constraint“ (dt. Dauerzusicherung) versehen ist. Die letzte Möglichkeit ist die modellbasierte Uhr, welche es erfordet, die Zeitmessung selbst im Modell umzusetzen[32].

Aktivitätsdiagramme können mithilfe der entsprechenden Simulations-Engine ausgeführt werden. Die einzelnen Aktionen werden dabei entsprechend der Definition im Diagramm nacheinander ausgeführt. Die Simulations-Engine unterstützt die Ausführung sämtlicher UML-Elemente sowie das Versenden und Empfangen von Signalen durch Ports. Aktionen können ebenfalls mit „opaque behaviors“ versehen werden. Dies erlaubt die Definition eines Verhaltens mithilfe einer Reihe unterstützter Skriptsprachen wie zum Beispiel JavaScript und Python. Auf die simulierte Modellinstanz selbst kann durch diese Skriptsprachen über eine API zugegriffen werden. Jenseits davon definiert Cameo eine Reihe von Aktionen, welche die Manipulation der Modellinstanz durch Aktionen erlauben[32].

Die Simulations-Engine zur Simulation von Zustandsdiagrammen unterstützt die meisten Elemente eines Zustandsdiagramms. Individuellen Zuständen können Aktivitäten zugewiesen werden. Diese Aktivitäten können entweder beim Eintritt in den Zustand, während des Zustandes oder beim Verlassen des Zustandes ausgeführt werden[32].

Cameo ist ebenfalls in der Lage, die verknüpften Zusicherungen des Zusicherungsdiagramms zu lösen. Das Diagramm wird erneut gelöst, wenn eine der modellierten Eigenschaften geändert wird[32].

Cameo bietet ebenfalls die Möglichkeit der Erstellung eines Sequenzdiagramms auf Basis der Simulation sowie der Verfolgung von Werten während der Simulation und deren anschließende Ausgabe im Comma Seperated Values (CSV)-Datenformat[32].

Cameo bietet somit die Möglichkeit, ein System zu simulieren. Dabei können Methoden wie die diskrete Event-Simulation zur Erstellung der Modelle zum Einsatz kommen.

2.5 Simulation von Produktionssystemen

Zur Simulation von Produktionssystemen ist DES eine sehr weit verbreitete Methode. DES ist eine Methode zur quantitativen Darstellung der Realität. Sie stellt die Dynamiken einer simulierten Umgebung auf einer Event-für-Event-Basis dar[33]. Prozesse werden modelliert als eine Abfolge konkreter Events, welche die Möglichkeit besitzen, den Zustand des Modells zu beeinflussen. Innerhalb der Simulation sind sowohl Ursache als auch Effekt in Bezug auf Events ausgedrückt[34].

DES wird traditionell für industrielle Anwendungen, wie die Produktionssimulation, eingesetzt[33]. Es ist ebenfalls möglich DES mithilfe von SysML zu modellieren. Dies wurde bereits in einer Reihe von Feldern umgesetzt[35][36][37][38]. Individuelle Events können hier in der Form von Aktionen und Zuständen modelliert werden[38].

Ebenso ist es möglich AAS mit einer DES eines Assets zu verknüpfen zur wissenschaftlichen Untersuchung von AAS-Verhalten[39]. Das Verhalten einer AAS kann ebenfalls mithilfe DES beschrieben werden[40].

2.6 Bewegungsverhalten des UR10e Roboters

Im Rahmen dieser Arbeit wird ein UR10e-Roboter eingesetzt, zur Validierung der hier entwickelten Methodik. Die Bewegung muss dabei in der zu erstellenden Simulation abgebildet werden.

Der UR10e Roboter besteht aus insgesamt sechs Gelenken, welche entlang des Arms des Roboters verteilt sind. Diese sind: das Basisgelenk, das Schultergelenk, das Ellenbogengelenk und drei Handgelenke. Jedes Gelenk ist in der Lage, entlang einer Achse zu rotieren. Die Position des Gelenks ist ausgedrückt durch den Winkel, um welche die momentane Position der Achse von einem vordefinierten Nullpunkt abweicht[41].

Der UR10e verfügt über die Möglichkeit der Montage eines Endeffektors[41]. Dieser erlaubt es dem UR10e, mit seiner Umgebung zu interagieren. Im Rahmen dieser Arbeit wird ein Greifer und ein Schraubenzieher verwendet.

Die Gelenke des UR10e verfügen über eine Maximalgeschwindigkeit von $180\frac{^\circ}{s}$, mit Ausnahme des Basis- und Schultergelenks, welche eine Maximalgeschwindigkeit von $120\frac{^\circ}{s}$ besitzen[41].

Der UR10e bewegt somit seinen Endeffektor zu einem spezifischen Punkt, durch die Anpassung der Position der jeweiligen Gelenkachsen[41].

Die Gelenkachsenbewegung wird durch den UR10e so geregelt, dass sämtliche Achsen gleichzeitig die Bewegung vollenden. Die individuellen Achsen verfolgen dabei ein Bewegungsmodell wie in Abbildung 2.8 gezeigt. Die Bewegung lässt sich in drei Phasen aufteilen. Dies sind eine Beschleunigungsphase, eine Bewegungsphase und eine Entschleunigungsphase. Die Höhe der Bewegungsphase wird durch die gegebene Geschwindigkeit definiert. Der Anstieg der Beschleunigungs- und Entschleunigungsphase wird durch die Beschleunigung gegeben[42].

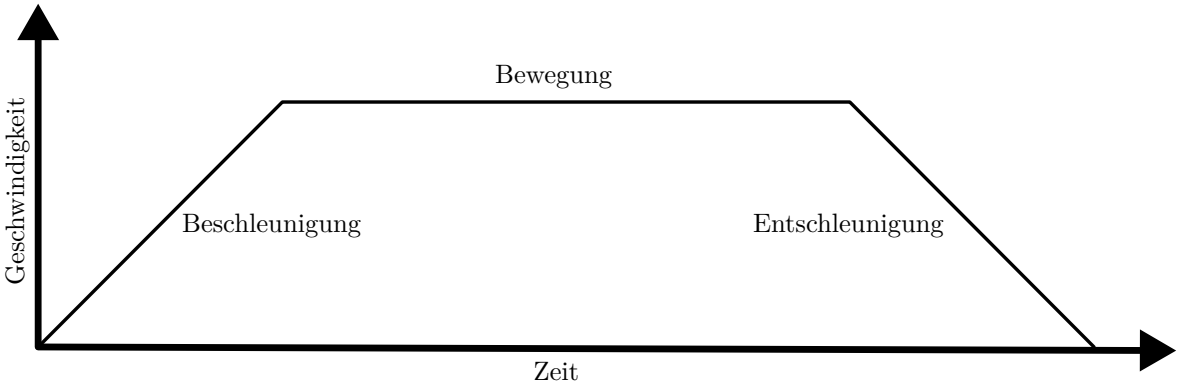


Abbildung 2.8: Geschwindigkeitsprofil einer Achse während der Bewegung nach [42]

Die Zeit, welche eine Achse benötigt, um eine Bewegung vollständig abzuschließen, ist somit die Summe der Zeiten der einzelnen Phasen des Bewegungsprofils. Die Zeit zur Beschleunigung und Entschleunigung ist hierbei gleich, da beide Bewegungen mit demselben Betrag der Beschleunigung durchgeführt werden. Die Gesamtzeit ergibt sich somit entsprechend Formel 2.1.

$$t = 2 * t_a + t_v \tag{2.1}$$

Die zurückgelegte Distanz während der Beschleunigungsphase wird entsprechend der Bewegungsgleichung einer beschleunigten Bewegung definiert. Formel 2.2 stellt dieses Verhältnis dar. Die Zeit t_a ist die Zeit, für welche die Beschleunigungsphase anhält.

$$d_a = 0,5 * a * (t_a)^2 \tag{2.2}$$

Die Bewegungsphase ist eine einfache lineare Bewegung bei gleichbleibender Geschwindigkeit. Sie kann entsprechend Formel 2.3 beschrieben werden. t_v ist hierbei die Zeit, für welche die Bewegungsphase andauert.

$$d_v = v * t_v \tag{2.3}$$

Sollte die Hälfte der zurückzulegenden Distanz kleiner sein als die Entfernung, welche während der Beschleunigung auf die Maximalgeschwindigkeit zurückgelegt wird, entfällt die Bewegungsphase.

Der Greifer verfügt über 2 Finger, welche geöffnet oder geschlossen werden können. Diese Finger sind nicht individuell ansteuerbar. Die Bewegung eines Fingers führt immer zur Bewegung des anderen Fingers. Die maximale Öffnungsweite des Greifers ist $85mm$. Der Greifer kann zu einer beliebigen Öffnungsweite bewegt werden. Die Geschwindigkeit dieser Bewegung beträgt $150\frac{mm}{s}$[43].

Der Schraubenzieher wiederum besteht aus einem Schaft, auf welchem ein Schraubbit angebracht wird. Dieser Schaft kann in das Gehäuse eingezogen werden. Der Schraubenzieher kann eine Reihe von Bewegungen ausführen. Diese sind: Bewege Schaft, Auf-

heben, Festziehen und Vorschrauben. Bewege Schaft ändert die Position des Schaftes. Aufheben ermöglicht es, Schrauben aufzunehmen. Zu diesem Zweck wird der Schaft nach außen bewegt und gleichzeitig rotiert. Vorschrauben schraubt eine Schraube über eine gegebene Länge in ein Gewinde ein. Festziehen wiederum schraubt eine Schraube so weit ein, bis ein Zielmoment erreicht ist[44].

2.7 Physik von Schraubverbindungen

Teil des in dieser Arbeit betrachteten Anwendungsfalls ist die Herstellung von zwei Schraubverbindungen. Dieser Prozess bedarf ebenfalls einer physischen Beschreibung.

Die für diese Arbeit relevante geometrische Größe ist die Ganghöhe P. Diese beschreibt die Höhe eines Umlaufs des Gewindes einer Schraube oder Mutter[45]. In Kombination mit einer gegebenen Umdrehungszahl U lässt sich daraus die vertikale Einschraubgeschwindigkeit errechnen. Diese ergibt sich, wie in Formel 2.4 dargestellt, aus dem Produkt der Ganghöhe und der Umdrehungszahl.

$$v_s = P * U \tag{2.4}$$

Die Schrauben in dieser Arbeit werden bis zum Erreichen eines gegebenen Moments angezogen. Dieses Gewindeanzugsmoment errechnet sich, entsprechend Formel 2.5[45], aus der Schraubenkraft F_S, dem Flankendurchmesser d_2, dem Flankenwinkel ϕ sowie dem Reibungswinkel ρ'.

$$M_G = 0,5 * F_S * d_2 * tan(\phi + \rho') \tag{2.5}$$

Die Schraubenkraft F_S steht wiederum in direktem Zusammenhang mit der elastischen Verlängerung der Schraube f in einem vorgespannten Zustand. Die Schraube lässt sich hierbei als Feder modellieren. Das reziproke der Federsteifigkeit für die Schraube ist hierbei die elastische Nachgiebigkeit δ. Sie ist die auf die Schraubenkraft F_S bezogene Verlängerung der Schraube. Sie lässt sich anhand der Geometrie- und Materialwerte einer Schraube bestimmen. Die elastische Verlängerung f der Schraube lässt sich somit entsprechend Formel 2.6 bestimmen.

$$f = \delta * F_S \tag{2.6}$$

2.8 Forschungslücke

Das Ziel dieser Arbeit ist die Entwicklung einer Methodik zur ganzheitlichen Simulation eines auf AAS basierten Produktionssystems. Hierbei sollen sowohl die Kommunikationsinfrastruktur, das Verhalten der AAS als auch das physische Verhalten der realen Ressourcen mitbetrachtet werden.

Zu diesem Zweck wurde in der Literatur untersucht, in welchem Umfang bereits SysML sowie allgemein Methoden des MBSE eingesetzt wurden zur Modellierung von AAS sowie der Verknüpfung von AAS mit Simulationsmodellen.

Molina und Treichel[46] haben gezeigt, wie AAS mit Simulationsmodellen verknüpft werden können. Sie haben zu diesem Zweck eine Implementierung einer AAS mit einem Simulationsmodell verbunden, welches Daten an die AAS zurückgibt. Sie konnten damit eine Methode zeigen, die auf MBSE-basierten Simulationsmodellen mit einer AAS verknüpft. Hier wurden jedoch lediglich die Assets mithilfe von MBSE abgebildet. Die AAS wurde nicht mithilfe von MBSE betrachtet. Wilking et al.[47] schlagen eine Methodik vor, mit der das Verhalten eines digitalen Zwillings in SysML modelliert werden kann. Das Ziel hierbei ist der Export des modellierten Verhaltens zu einem digitalen Zwilling. Es wurde zusätzlich ein Modell des Gesamtsystems erstellt, um die Simulation zu vereinfachen. Er stellt darin fest, dass die Nutzung von SysML zur Definition des Verhaltens von digitalen Zwillingen vorteilhaft sein kann. Die Kommunikationsinfrastruktur zwischen den einzelnen digitalen Zwillingen wurde hier jedoch nicht abgebildet. Dies könnte jedoch einen besseren Überblick über das Gesamtsystem bieten.

Das Ziel dieser Arbeit ist, diese individuellen Methoden zu integrieren, um eine ganzheitliche Betrachtung des Produktionsprozesses zu ermöglichen. Im Rahmen dieser Arbeit soll die Modellierung mit SysML auf alle Teile des Produktionssystems erweitert werden. Im Unterschied zu Molina und Treichel[46] wird die AAS in dieser Arbeit ebenfalls mit SysML modelliert. Anders als bei Wilking et al.[47] wird in dieser Arbeit nicht das Verhalten von allgemeinen digitalen Zwillingen abgebildet, sondern das Verhalten der standardisierten AAS. Es wird im Rahmen dieser Arbeit ebenfalls ein explizites Modell der Kommunikationsinfrastruktur ergänzt.

3 Methodik zur Modellierung des Produktionssystems

Das Ziel der Arbeit ist es, ein Produktionssystem ganzheitlich in SysML zu modellieren, um eine Simulation auf Basis dieses Modells zu ermöglichen. Zu diesem Zweck werden alle Systembestandteile in SysML modelliert, inklusive ihres Verhaltens. Diese einzelnen Modelle werden im Rahmen des Modells eines Produktionssystems miteinander verknüpft.

Das Modell des Produktionssystems ist in drei Ebenen aufgeteilt. Eine schematische Darstellung dieser Ebenen ist in Abbildung 3.1 zu sehen. Die Ebenen sind: die Kommunikationsebene, die AAS-Ebene und die Asset-Ebene. Der Austausch von Nachrichten erfolgt nur über Ebenen hinweg. Die Produkte und Ressourcen in der Asset-Ebene kommunizieren ausschließlich mit ihren jeweiligen AAS in der AAS-Ebene und die AAS kommunizieren ausschließlich miteinander über die Kommunikationsebene.

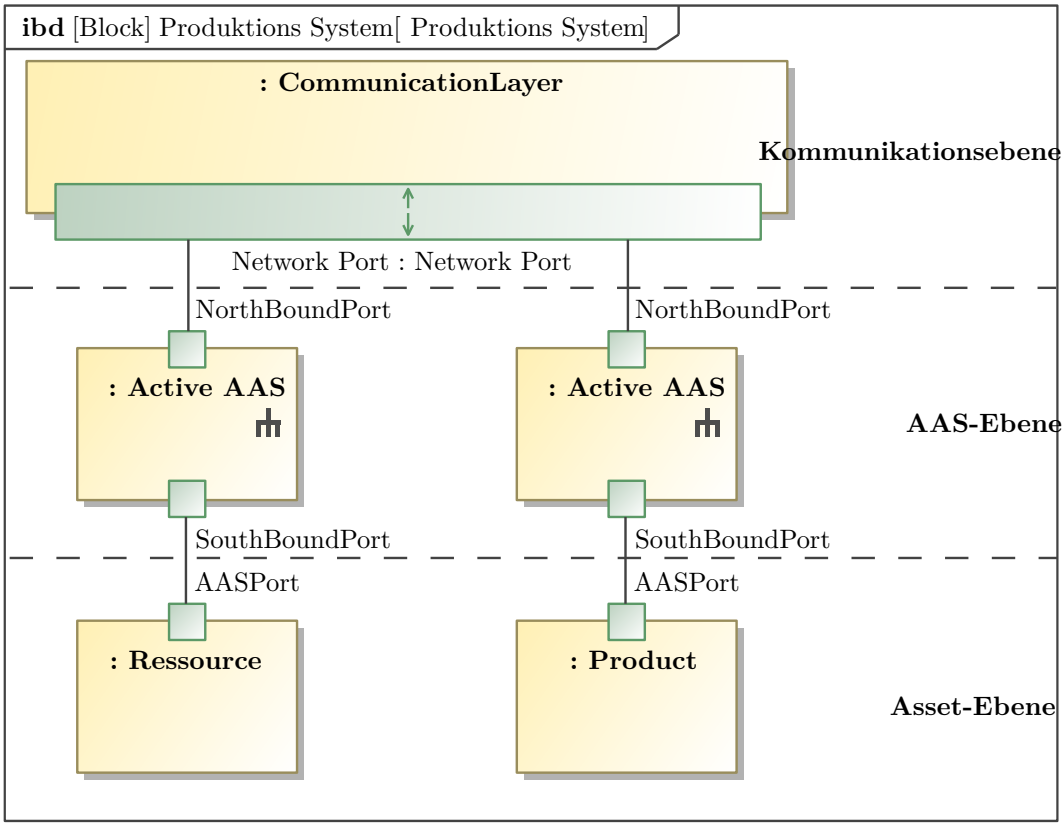


Abbildung 3.1: Internes Blockdiagramm des Produktionssystems strukturiert in Asset-Ebene, AAS-Ebene und Kommunikationsebene

Die Kommunikationsebene modelliert den Nachrichtenaustausch zwischen den AAS. Ihre Aufgabe ist es, die Kommunikationsinfrastruktur des Produktionssystems abzubilden.

Die AAS-Ebene umfasst die Modelle der AAS der jeweiligen Assets. Die AAS können untereinander über die Kommunikationsebene Nachrichten austauschen. Sie können ebenfalls Befehle an die Asset-Ebene übertragen und Nachrichten von dieser erhalten. Verbindungen von der AAS zur Kommunikationsebene werden als „northbound“ (dt. nordgerichtete) Verbindungen bezeichnet. Verbindungen zu den Assets werden als „southbound“ (dt. südgerichtete) Verbindungen bezeichnet.

Auf der Asset-Ebene sind die Modelle aller Ressourcen und Produkte zu finden. Bei diesen Modellen handelt es sich um spezifische Abbildungen der realen Ressourcen und Produkte. Sie sind in der Lage, Befehle von den AAS zu empfangen und Daten an diese zu übertragen.

Das Produktionssystem selbst verfügt über kein eigenes, durch Zustands- oder Aktivitätsdiagramme definiertes Verhalten. Das Verhalten des Produktionssystems ergibt sich ausschließlich aus dem Verhalten der einzelnen Systemkomponenten.

3.1 Modellierung der Assets

Die Beziehungen der Assets sind in Abbildung 3.2 dargestellt. Das Asset verfügt über keine eigenen Eigenschaften außer einem „AASPort“. Diese müssen durch spezifische Implementierungen der abzubildenden Assets modelliert werden. Der „AASPort“ ist die Kommunikationsschnittstelle zur AAS. Aus dem Asset leiten sich die Ressourcen und Produkte ab.

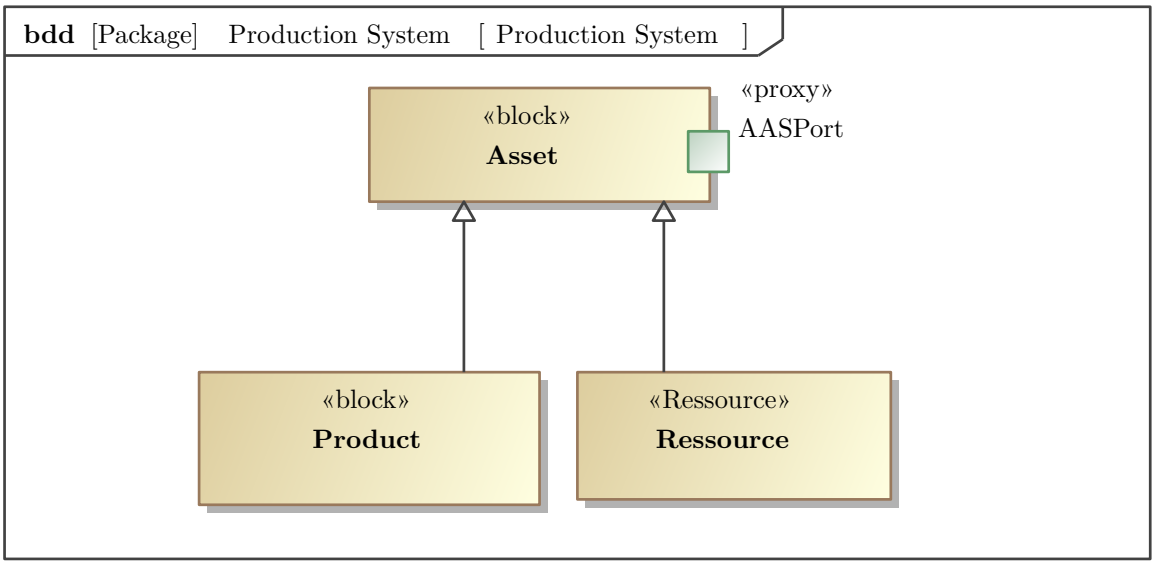


Abbildung 3.2: Blockdefinitionsdiagramm des Assets

3.2 Modellierung der Verwaltungsschale

Die AAS sind im Rahmen dieser Arbeit lediglich teilweise modelliert. Ausschließlich die Teile, welche relevant für das Verhalten der AAS sind, wurden abgebildet. In Abbildung 3.3 ist die allgemeine Struktur der Definition der AAS zu sehen. Die Grundlage für alle AAS bildet ein allgemeiner AAS-Block, der lediglich aus einem Bestandteil besteht. Dieser ist der „AAS Properties“-Block. Dieser Block umfasst sämtliche Eigenschaften, welche die AAS beschreiben. Im Rahmen dieser Arbeit ist hier lediglich der Name der AAS abgebildet, welcher als Identifikator zur eindeutigen Identifizierung der AAS dient.

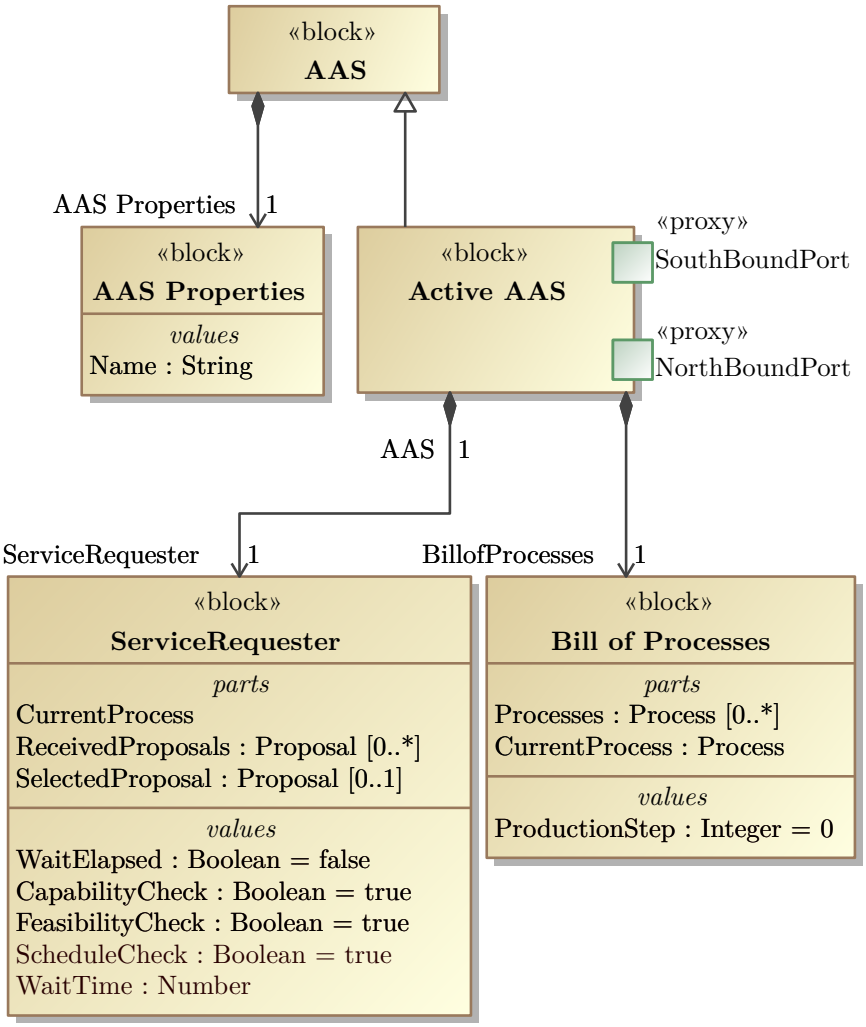


Abbildung 3.3: Blockdefinition der Verwaltungsschale

Von der allgemeinen AAS abgeleitet ist die „Active AAS“ (dt. Aktive AAS). Entsprechend der Definition in Abschnitt 2.3 bildet dieser Block eine AAS der Stufe 3 ab. Diese AAS ist ausgestattet mit 2 Ports. Der „NorthBoundPort“ (dt. nordgerichtete Port) ermöglicht die nordgerichtete Kommunikation nach außen und mit anderen AAS. Der

„SouthBoundPort“ (dt. südgerichtete Port) ermöglicht die südgerichtete Kommunikation mit den jeweiligen Assets. Neben diesen Ports verfügt jede aktive AAS über einen „Service Requester“ (dt. Dienstanfrager), sowie eine „Bill of Processes“ (dt. Prozessliste). Der Dienstanfrager übernimmt dabei die Aufgabe, in Verhandlung mit anderen AAS zu treten, zur Ausführung von Prozessen, welche in der Prozessliste definiert sind.

Eine weitere Spezifizierung der aktiven AAS ist die „ServiceProvider AAS“ (dt. Dienstanbieter AAS). Deren Definition ist in Abbildung 3.4 dargestellt. Die Dienstanbieter AAS erweitert die aktive AAS um den „ServiceProvider“ (dt. Dienstanbieter), dessen Aufgabe es ist, die der AAS verfügbaren Dienste an andere AAS anzubieten. Der „SouthBoundIntegrator“ (dt. südgerichteter Integrator) ist eine assetspezifische Implementierung der Kommunikation mit dem Asset. Final verfügt die „AAS“ über eine Referenz auf eine Reihe von „Capabilities“ (dt. Fähigkeiten), welche beschreiben, welche Prozesse die AAS ausführen kann. Fähigkeiten sind im Rahmen dieser Arbeit vereinfacht durch lediglich einen Identifikator definiert.

Die Integration der AAS mit den technischen Ressourcen erfolgt durch den in Abbildung 3.4 abgebildeten südgerichteten Integratorblock. Dieser verfügt über eine Eigenschaft zur Speicherung des momentanen Prozesses, besitzt jedoch kein eigenes Verhalten. Die assetspezifischen Integratoren implementieren ihr eigenes Verhalten, welches auf das jeweilige Asset angepasst ist.

Der AAS-Block verfügt über kein eigenes Verhalten. Stattdessen wird das Verhalten der AAS ausschließlich durch die Blöcke Dienstanfrager, Dienstanbieter, Prozessliste und südgerichteter Integrator definiert.

3.2.1 Prozessliste

Die Prozessliste sammelt und verwaltet alle Prozesse, die ausgeführt werden müssen, um ein entsprechendes Produkt fertigzustellen. Die Prozesse werden in einer nach Reihenfolge geordneten Liste gespeichert. Dies wird umgesetzt durch die Eigenschaft „Processes“ (dt. Prozesse) mit einer Multiplizität, welche definiert, dass die Eigenschaft minimal null Elemente und maximal beliebig viele Elemente umfassen kann. Der momentane Bearbeitungszustand wird durch eine Prozessschrittnummer dargestellt, welche angibt, wie viele Prozesse in der Liste bereits abgeschlossen wurden.

Das Verhalten der Prozessliste besteht darin, den nächsten zu bearbeitenden Prozess zu finden und anschließend an die restlichen Bestandteile der AAS zu übertragen, so dass dieser ausgeführt werden kann. Die Prozessliste wartet auf die Ausführung des Prozesses und fährt anschließend mit dem nächsten Prozess fort, solange bis alle Prozesse bearbeitet sind.

Der Prozess selbst ist abstrakt modelliert. Das Blockdefinitionsdiagramm des Prozess-Blocks ist in Abbildung 3.5 zu sehen. Der Prozess selbst verfügt über einen Wahrheitswert, welcher angibt, ob der Prozess bereits ausgeführt wurde. Eine Referenz auf eine Fähigkeit definiert, welche Fähigkeit benötigt wird zur Ausführung des Prozesses. Die weitere Beschreibung des Prozesses erfolgt durch einen „Process Description“-Block

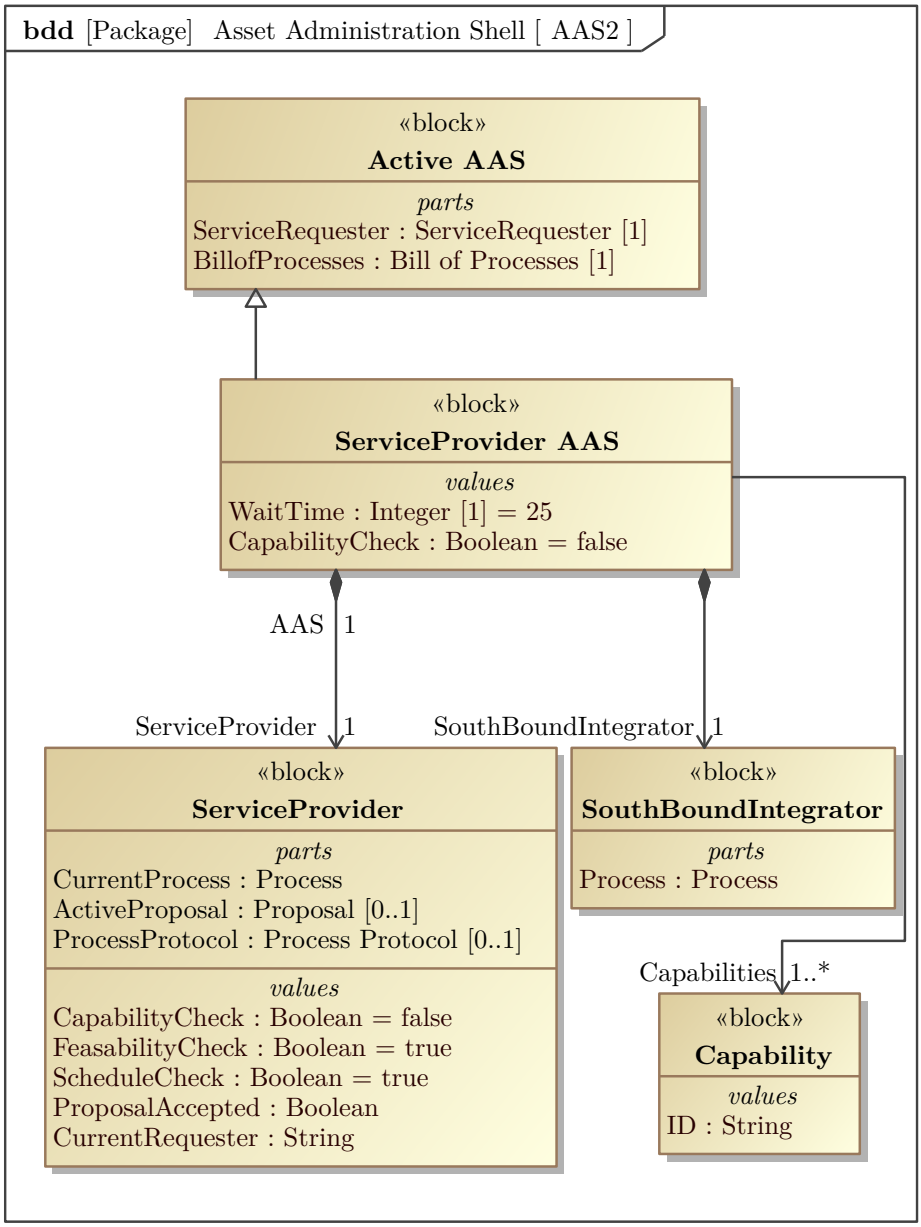


Abbildung 3.4: Blockdefinition der Dienstanbieter Verwaltungsschale

(dt. Prozessbeschreibungsblock). Dieser definiert den Prozess durch den Einsatz von drei Textwerten: einem Verb, einem oder mehreren Objekten und einem Ort. Das Verb definiert in abstrakter Weise die Tätigkeit, welche im Rahmen des Prozesses auszuführen ist. Es hängt somit direkt mit der Fähigkeit zusammen. Das Verb kann zum Beispiel „Schraube“ oder „Platziere“ sein. Das Objekt definiert die Bauteile, die zur Ausführung des Prozesses benötigt werden. Bauteile werden hier durch einen eindeutigen Typenidentifikator spezifiziert. Der Ort definiert, wo etwas mit einem Bauteil passieren soll.

Die Prozessbeschreibung kann bei Bedarf durch eine beliebige Menge an „ProcessProperties“ (dt. Prozesseigenschaften) weiter spezifiziert werden. Diese werden individuell durch einen Namen und einen Wert definiert.

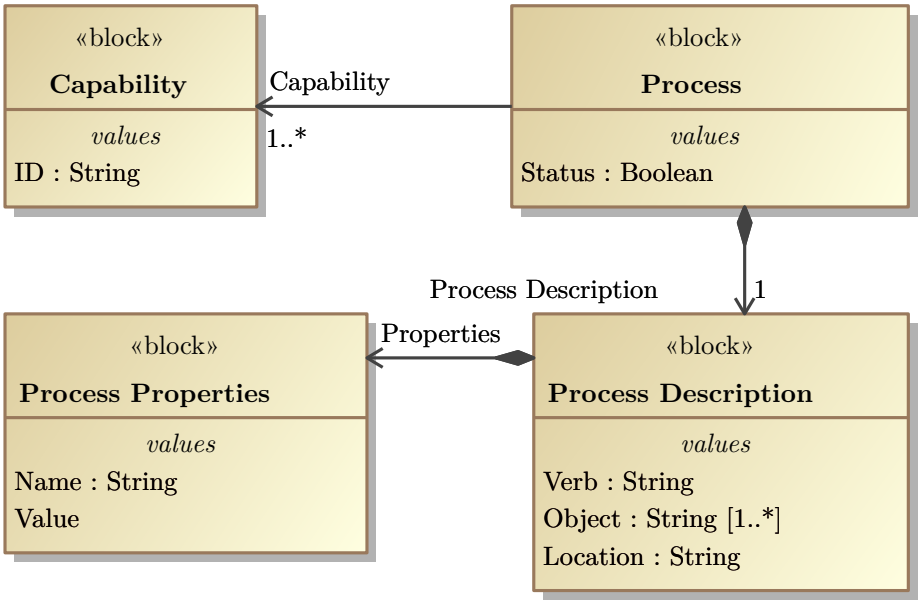


Abbildung 3.5: Modellierung des Prozesses

3.2.2 Dienstanfrager

Der Dienstanfrager ist ein Bestandteil der AAS mit eigenem Verhalten. Seine Bestandteile sind in Abbildung 3.3 dargestellt. Er verfügt über den momentanen Prozess, eine Liste an eingetroffenen Angeboten sowie das angenommene Angebot. Des Weiteren verfügt er über eine Reihe an Wahrheitswerten, welche den Übergang zwischen den einzelnen Zuständen kontrollieren.

Das Verhalten des Dienstanfragers ist in Abbildung 3.6 dargestellt. Der Dienstanfrager folgt dabei dem in Abschnitt 2.3 beschriebenen Ausschreibungsverfahren und bildet die in der AAS genutzte Zustandsmaschine ab. Die Zustände sind nach Farben gruppiert. Der Anfangszustand ist in grün dargestellt. Die Versendung der Ausschreibung und das Warten auf Antworten sind in Orange abgebildet. Sämtliche Zustände, die mit der Bearbeitung der eingetroffenen Antworten betraut sind, sind in Blautönen dargestellt. Zustände, die über kein abgebildetes Verhalten verfügen, wurden hier mit in Hellblau abgebildet. Die Zustände, die sich mit der Ausführung des Prozesses befassen, sind in Rot abgebildet. Der Dienstanfrager verweilt im Zustand „WaitForNewOrder“ (dt. Warten auf neuen Auftrag), bis ein Arbeitsbedarf in Form eines in der AAS internen Signals bei ihm eingeht. Dieses Signal umfasst eine Beschreibung des Prozesses, welcher ausgeführt werden soll. Der Dienstanfrager veröffentlicht daraufhin einen „Call For Proposal“ (dt. Ausschreibung) für den jeweiligen Prozess. Anschließend wartet der Dienstanfrager auf den Eingang möglicher Angebote. Im Rahmen dieser Arbeit wurde eine Wartezeit von 5 s gewählt. Es handelt sich dabei um einen Kompromiss, dazwischen zum einen allen Ressourcen die Möglichkeit zu geben, auf die Ausschreibung zu antworten, und zum anderen so schnell wie möglich mit der Ausführung des Prozesses zu beginnen.

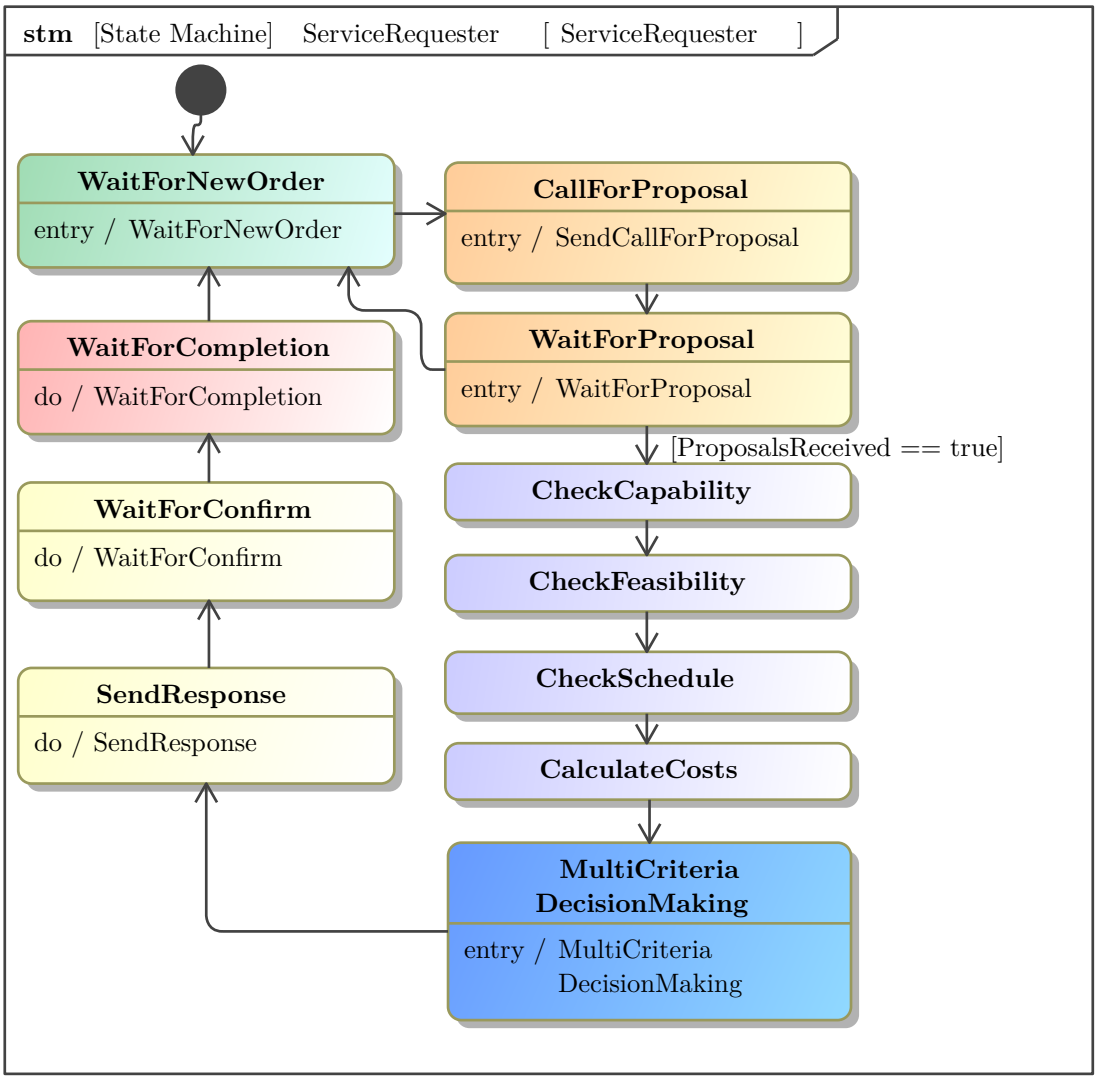


Abbildung 3.6: Zustandsdiagramm des Dienstanfragers

Nach dem Verstreichen der Wartezeit von 5s überprüft der Dienstanfrager, ob Angebote eingegangen sind. Ist dies nicht der Fall, geht er zurück in den Zustand Warten auf neuen Auftrag. Sollten Angebote eingegangen sein, werden diese weiterführend verarbeitet. Die als Grundlage für die Modellierungen verwendete AAS verfügt zu diesem Zweck über die Zustände „CheckCapability“ (dt. Überprüfe Fähigkeit), „CheckFeasibility“ (dt. Überprüfe Machbarkeit), „CheckSchedule“ (dt. Überprüfe Zeitplan) und „CalculateCosts“ (dt. Berechne Kosten). Diese Zustände sind zum Zeitpunkt der Erstellung dieser Arbeit lediglich Platzhalter für die Implementierung zukünftigen Verhaltens. Sie sind in der Modellierung hier ebenfalls lediglich Platzhalter, welche zur Vollständigkeit ebenfalls aufgelistet sind.

Im Schritt „MultiCriteriaDecisionMaking“ (dt. Multi-Kriterielle Entscheidungsfindung) wird, innerhalb der in Abschnitt 2.3 beschriebenen AAS, auf der Basis einer Vielzahl von Kriterien das beste Angebot ausgewählt. Dieses Verhalten ist im Modell in sehr vereinfachter Form dargestellt. Die Entscheidungsfindung erfolgt auf der Basis lediglich

eines Kriteriums.

Nach der Auswahl des besten Angebots wird die Ressource, welche das Angebot eingereicht hat, im Zustand „SendResponse“ (dt. Sende Antwort) mit der Ausführung des Prozesses beauftragt. Der Dienstanbieter wartet anschließend auf eine Bestätigung von der Seite der Ressource im Zustand „WaitForConfirm“ (dt. Warte auf Bestätigung). Nach Eingang dieser Bestätigung werden alle anderen Ressourcen über die Absage informiert. Abschließend wartet der Dienstanfrager, bis der Prozess im Zustand „WaitForCompletion“ (dt. Warte auf Abschluss) abgeschlossen ist. Über den Abschluss des Prozesses wird dieser durch eine Nachricht informiert, welche ebenfalls ein Protokoll der Prozessausführung beinhaltet. Nach Abschluss des Prozesses geht der Dienstanfrager wieder in den Anfangszustand Warten auf neuen Auftrag über.

3.2.3 Dienstanbieter

Das Gegenstück zum Dienstanfrager innerhalb des Ausschreibungsverfahrens ist der Dienstanbieter. Dessen Verhalten ist in Abbildung 3.7 dargestellt. Auch hier sind die Zustände nach Farben gruppiert. Der Anfangszustand ist grün markiert. Sämtliche Prozesse, welche mit der Erstellung des Angebots betraut sind, sind blau markiert, mit Zuständen ohne Verhalten in einem hellblauen Farbton. In rot abgebildet sind die Zustände, welche die Ausführung des Produktionsprozesses umfassen. Der Dienstanbieter verschickt Angebote im Rahmen des Ausschreibungsverfahrens zur Bereitstellung der jeweiligen Dienste des Assets. Der Dienstanbieter verbleibt im Zustand „WaitForCallForProposal“ (dt. Warte auf Ausschreibung), solange bis eine Ausschreibung bei ihm eingeht. Es kommt daraufhin zum Übergang in den Zustand „CheckCapability“ (dt. Überprüfe Fähigkeit). In diesem Zustand wird die Liste der Fähigkeiten des Assets mit der Fähigkeit abgeglichen, welche dem der Ausschreibung anhängenden Prozess beiliegt. Verfügt das Asset nicht über die notwendige Fähigkeit, wird der Ausschreibung eine Absage erteilt und der Dienstanbieter geht wieder in den Zustand Warte auf Ausschreibung über.

Sollte das Asset die benötigte Fähigkeit besitzen, wird ein Angebot erstellt. Ähnlich wie beim in Abschnitt 3.2.2 beschriebenen Dienstanfrager erfolgt dies über eine Reihe von Zuständen, die lediglich als Platzhalter im Modell abgebildet sind. Dies sind die Zustände „CheckFeasibility“ (dt. Überprüfe Machbarkeit), „CheckSchedule“ (dt. Überprüfe Zeitplan) und „MultiCriteriaDecisionMaking“ (dt. Multi-Kriterielle Entscheidungsfindung). Das Angebot ergibt sich im Rahmen dieser Arbeit lediglich aus dem Zustand „CalculateCost“ (dt. Berechne Kosten). Hierbei werden die Kosten als ein einfacher, für das Asset spezifischer Zahlenwert ausgedrückt und dem Angebot angehangen.

Im Zustand „SendProposal“ (dt. Sende Angebot) wird das Angebot anschließend der Dienstanfrager-AAS übermittelt. Daraufhin verweilt der Dienstanbieter im Zustand „WaitForResponse“ (dt. Warte auf Antwort), bis er eine Antwort auf das Angebot erhält. Wird das Angebot abgelehnt, kehrt er in den Zustand Warte auf Ausschreibung zurück. Wird das Angebot angenommen, wird das Ausschreibungsverfahren abgeschlossen, durch die Übertragung einer Bestätigung im Zustand „SendConfirmation“ (dt. Sende Bestätigung).

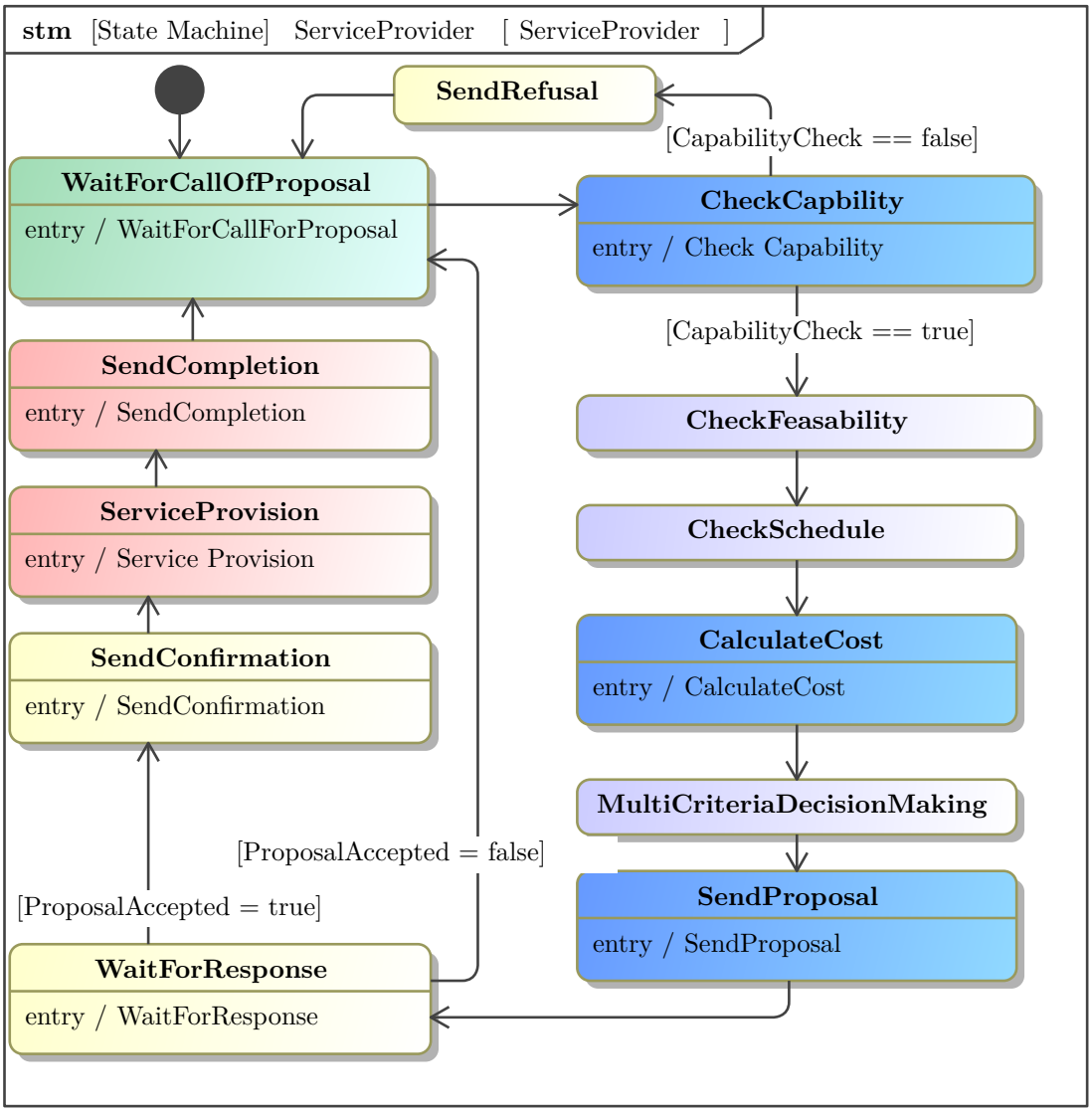


Abbildung 3.7: Zustandsdiagramm des Dienstanbieters

Darauf folgt die Dienstbereitstellung im Zustand „ServiceProvision“ (dt. Dienstbereitstellung). Dies umfasst die Übertragung der Beschreibung des auszuführenden Prozesses an den Integrator zum Asset. Nach der erfolgreichen Ausführung des Prozesses wird eine Nachricht mit einem Protokoll an den Dienstanbieter übermittelt, um diesen über die Fertigstellung des Prozesses zu informieren. Abschließend kehrt der Dienstanbieter in den Zustand Warte auf Ausschreibung zurück.

3.3 Modell der Kommunikationsinfrastruktur

Zur Abbildung der Kommunikation zwischen den AAS über ein Netzwerk wird ein Block eingesetzt. Dieser stellt die Kommunikationsebene dar.

Zur Kommunikation tauschen die AAS Netzwerknachrichten untereinander aus. Diese

Netzwerknachrichten sind, wie in Abbildung 3.8 abgebildet, modelliert. Jede Netzwerknachricht verfügt über zwei Eigenschaften zur Definition von allgemeinen Informationen. Dies sind das Ziel und der Typ der Nachricht. Das Ziel spezifiziert, für welchen Empfänger diese Nachricht bestimmt ist. Der Typ der Nachricht gibt an, welchen Inhalt die Nachricht hat und erlaubt es dem Empfänger, die Informationen korrekt auszulesen.

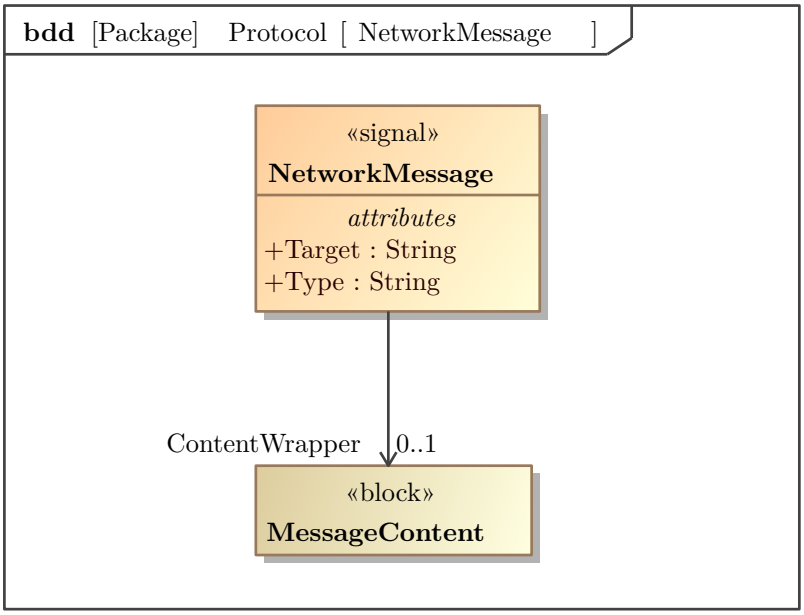


Abbildung 3.8: Blockdefinitionsdiagramm der Netzwerknachricht

Der Inhalt der Nachricht ist als extra Block modelliert. Dies ist der „MessageContent“-Block (dt. Nachrichteninhaltblock). Verschiedene Nachrichten können verschiedene Informationen enthalten, daher werden typspezifische Inhaltsblöcke erstellt, mit eigenem Inhalt, welche in einer Generalisierungsbeziehung zum Nachrichteninhaltblock stehen.

Diese Nachrichten werden an den Kommunikationsebenenblock durch die AAS übertragen. Dessen Verhalten besteht ausschließlich aus dem Empfangen und Verteilen von Nachrichten. Aus diesen Nachrichten wird das gewollte Ziel ausgelesen und die Nachricht anschließend an den, entsprechend dem Ziel, spezifizierten Empfänger unverändert weitergeleitet.

Bei jeder Übertragung einer Nachricht wird zusätzlich eine Verzögerung in der Kommunikation simuliert. Die Dauer dieser Verzögerung ist abhängig vom spezifischen zu modellierenden Produktionssystem und muss dementsprechend für jedes individuell bestimmt werden.

4 Anwendung anhand der Vormontage von Großbaumodulen in der Luftfahrt

Mithilfe der in Kapitel 3 vorgestellten Modelle kann ein beliebiges auf AAS basierendes Produktionssystem abgebildet werden. Zur Überprüfung der Methodik wurde sie auf einen spezifischen Anwendungsfall in der Luftfahrt angewandt. Anschließend wurde die Ausführung des Anwendungsfalls durch ein reales Produktionssystem mit der Simulation der Modelle verglichen.

4.1 Anwendungsfall

Zur Validierung des in dieser Arbeit zu entwickelnden Modells wird die Montage von Großmodulen in der Flugzeugkabine verwendet. Zu diesem Zweck wird die am Institut für Systemarchitekturen in der Luftfahrt des DLR vorhandene Vormontagezelle verwendet. Ein schematischer Aufbau dieser ist in Abbildung 4.1 dargestellt. Die Vormontagezelle besteht für den Zweck des Anwendungsfalls aus drei Teilen: einem UR10e-Roboter, einer Vorrichtung und einer Lagerfläche. Der UR10e verfügt über zwei Endeffektoren, einem Greifer und einem Schraubenzieher.

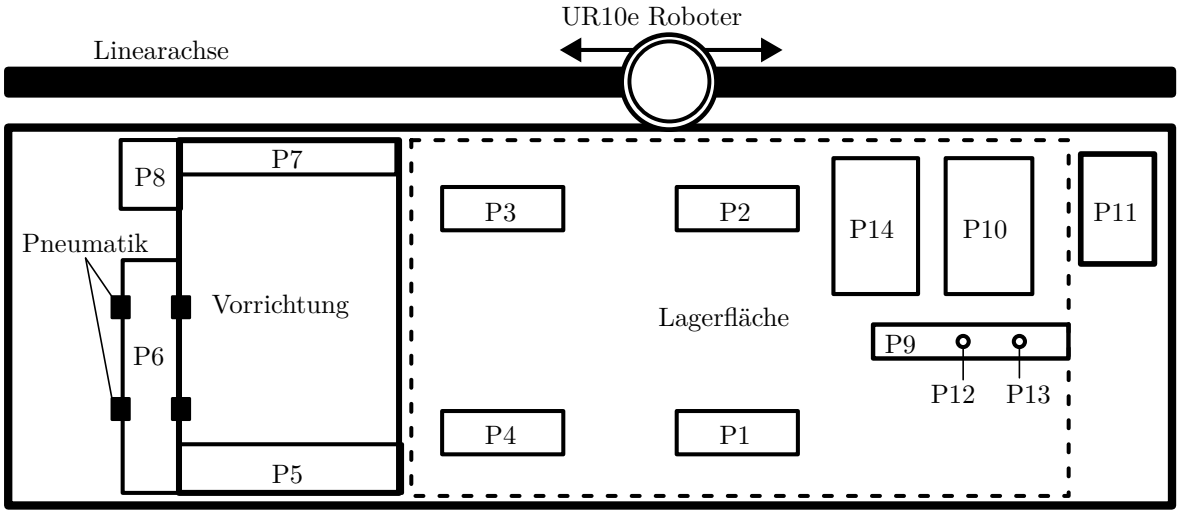


Abbildung 4.1: Schematischer Aufbau der Vormontagezelle

Sowohl der Roboter als auch der Lagerplatz sind mit einer AAS ausgestattet. Beide AAS werden auf einem zentralen Rechner ausgeführt. Dieser Rechner verfügt über einen Intel Core i7-6600U Prozessor und 32 GB RAM. Er ist mit dem Betriebssystem Windows 10 ausgestattet. Dieser ist durch ein LAN-Kabel mit einem lokalen Netzwerk verbunden, über welches ein Zugriff auf den Roboter möglich ist. Die Bewegung des Roboters wird durch die AAS über eine Socketverbindung gesteuert.

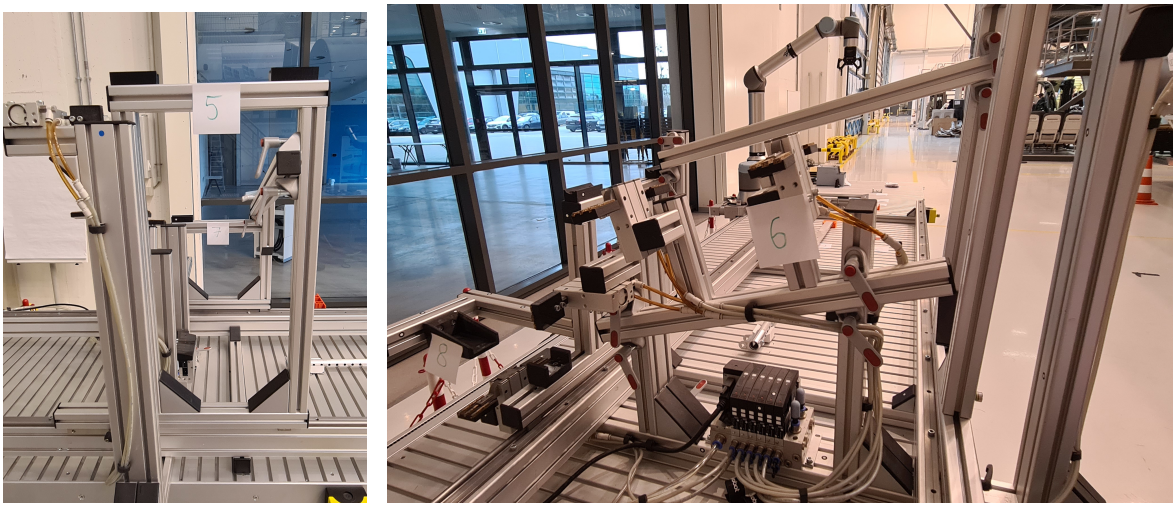
Der Roboter kann sich entlang der Linearachse frei bewegen. Zur Ausführung von Roboterprozessen wurden dem Roboter die Bewegungen zu einer Reihe von Punkten antrainiert. Diese Punkte sind in Abbildung 4.1 abgebildet und als PX bezeichnet, wobei X die Nummer des Punktes ist. Der Roboter wurde so vortrainiert, dass für jeden Punkt eine Liste an Befehlen vorliegt, welche es dem Roboter erlaubt, diesen Punkt zu erreichen. Die Befehle sind dabei so strukturiert, dass ein Durchlaufen dieser von vorne nach hinten dem Aufheben eines Bauteils am entsprechenden Punkt entspricht. Ein Durchlaufen von hinten nach vorne entspricht dem Ablegen eines Bauteils. Der Roboter beginnt und beendet eine Bewegung immer in derselben Position.

Die Punkte P11, P12 und P13 unterscheiden sich von den anderen Punkten. Die Befehle für diese Punkte umfassen nicht das Aufheben und Ablegen von Bauteilen, sondern beschreiben das Einschrauben von Schrauben an P12 und P13. Der Punkt P11 ist die Schraubenzufuhr. An diesem Punkt ist ein Gerät vorhanden, welches stetig Schrauben nachführt. Die Bewegung zum Punkt P11 umfasst das Aufnehmen einer Schraube. Die Punkte P12 und P13 beschreiben die Positionen, an welchen sich die Löcher befinden, in welche Schrauben eingeführt werden können. Die Positionen verfügen über je eine Variation, welche P12t und P13t genannt werden. Die Bewegungen P12 und P13 umfassen das Vorschrauben. Dies ist die Einführung einer Schraube bis zu einer gewissen Länge. P12t und P13t umfassen das Festziehen der Schrauben. Die Schrauben werden hier solange eingeschraubt, bis ein Zielmoment erreicht ist.

Abbildung 4.2 zeigt die Vorrichtung. Es ist ein Aufbau, auf dem Bauteile zur Montage abgelegt werden können. Die Punkte P5, P6, P7 und P8 liegen auf der Vorrichtung. Die Vorrichtung ist der Bereich, in dem das gewünschte Großmodul zusammengebaut wird. Der Punkt P6 verfügt über zwei Pneumatikgreifer, welche durch den UR10e-Roboter gesteuert werden. Diese erlauben es, das Bauteil an Position P6 zu platzieren.

Im Lagerplatz liegen alle benötigten Einzelteile. Die Punkte P1, P2, P3, P4 und P14 sind Teil des Lagerplatzes. Die Punkte P9 und P10 sind spezielle Positionen, in denen Teile abgelegt werden können, die zusammengeschraubt werden sollen.

Das in dieser Arbeit betrachtete Großmodul ist das Crownmodul. Das Crownmodul ist momentaner Gegenstand der Forschung zur Optimierung der Montageprozesse in der Flugzeugkabine. Es ist im Deckenbereich der Kabine zu verorten und umfasst dabei die Gepäckablage, elektrische Komponenten, Luftversorgung und mechanische Verbindungen. Da der Einbau jeder dieser einzelnen Komponenten ein komplexer und langwieriger Vorgang ist, werden diese nicht direkt in der Kabine, sondern auf dem Crownmodul aufgebracht. Dieses wird dann als Ganzes in das Flugzeug hineingeschoben und am Rumpf befestigt[48]. Das Grundgerüst des Crownmoduls ist in Abbildung 4.3 in Rot dargestellt.



(a) Vorderansicht (b) Seitenansicht

Abbildung 4.2: die Vorrichtung von vorne und von der Seite

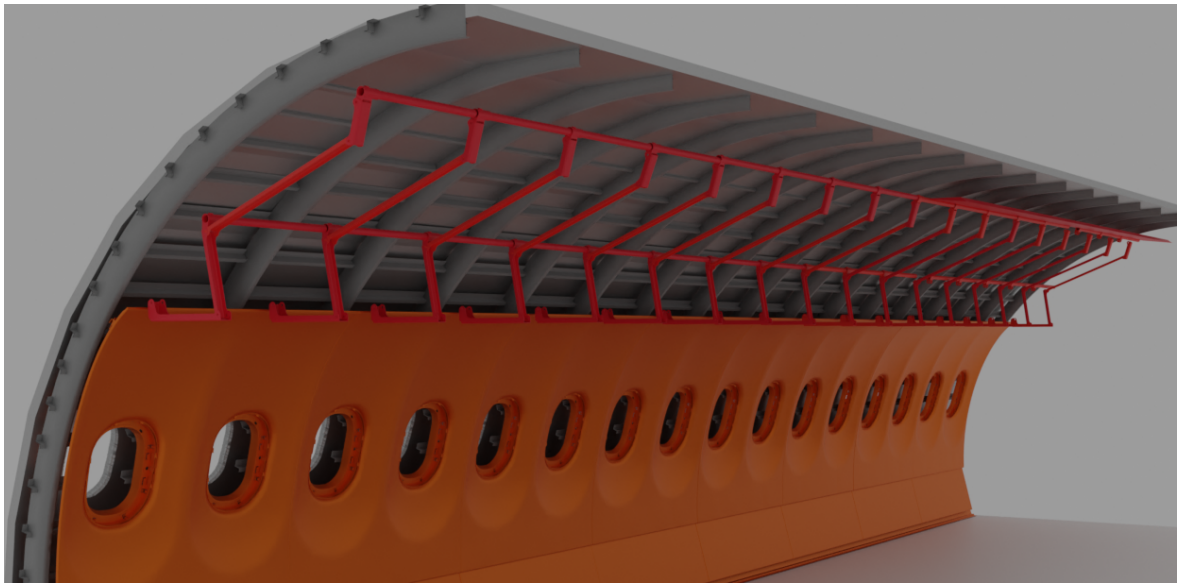


Abbildung 4.3: das Grundgerüst des Crownmoduls

Im Rahmen dieser Arbeit wird ein Abschnitt des Grundgerüsts des Crownmoduls teilweise zusammengebaut. Zusätzlich wird eine Decoder-Encoder-Unit (DEU) auf das Gerüst des Crownmoduls montiert. Zur Montage des Grundgerüsts werden 4 Teile verwendet: die VFork, die XRodA, die XRodB und die HFork. Die Teile sind in Abbildung 4.4 abgebildet.

An der HFork wird die DEU mit zwei M5-Schrauben befestigt. Die DEU ist in Abbildung 4.5 abgebildet. Es handelt sich bei ihr um eine DEU-A mit der Teilnummer 200RH13, welche unter anderem in A319 zum Einsatz kommt[49]. Die DEU und HFork werden an den Positionen P10 und P9 platziert, dort zusammengeschraubt und anschließend auf P6 auf der Vorrichtung abgelegt und dort mithilfe der Pneumatik festgehalten.

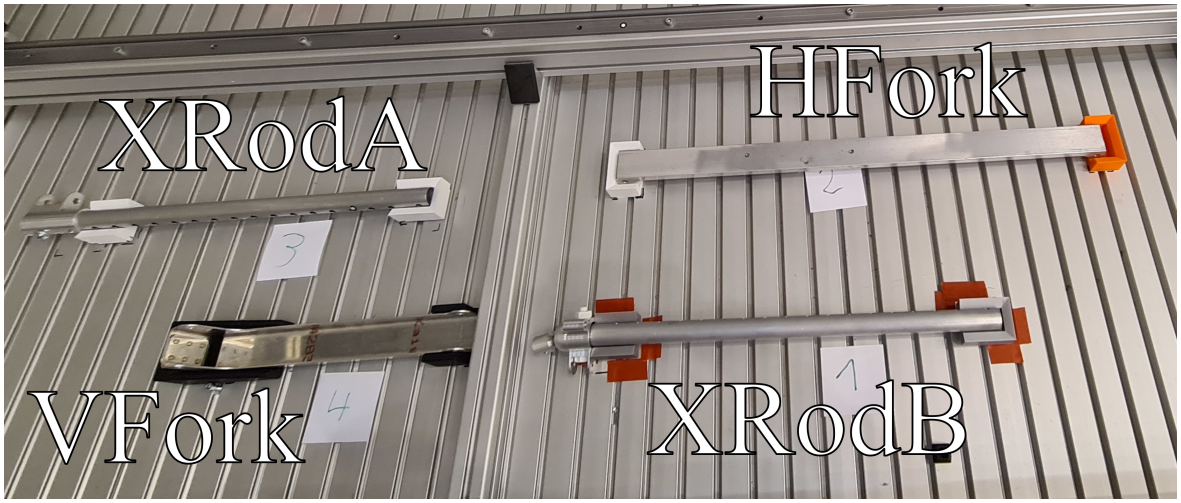


Abbildung 4.4: Bauteile des Gerüsts des Crownmoduls

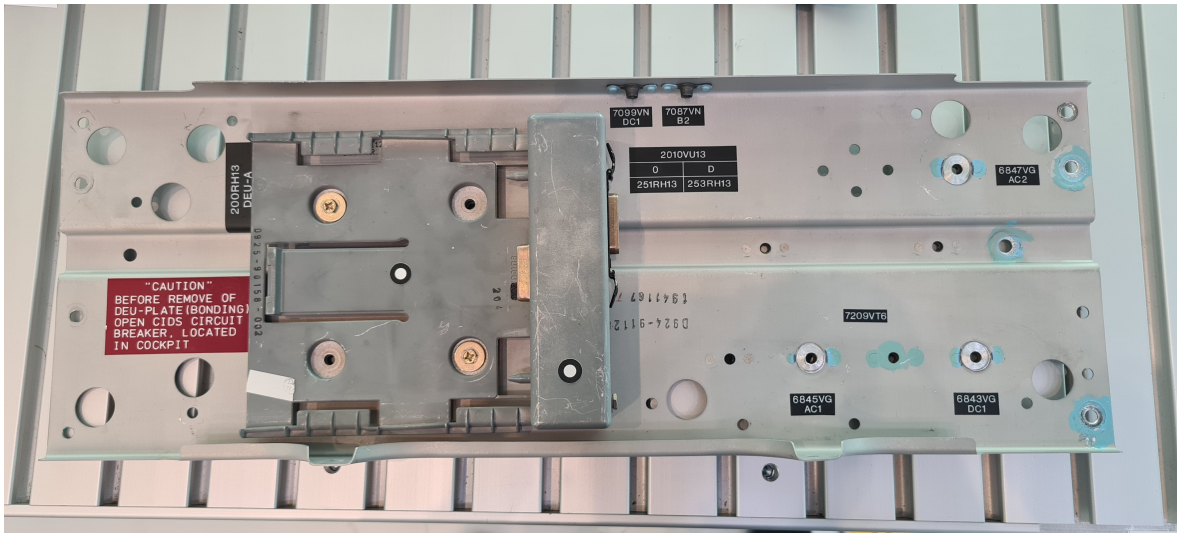
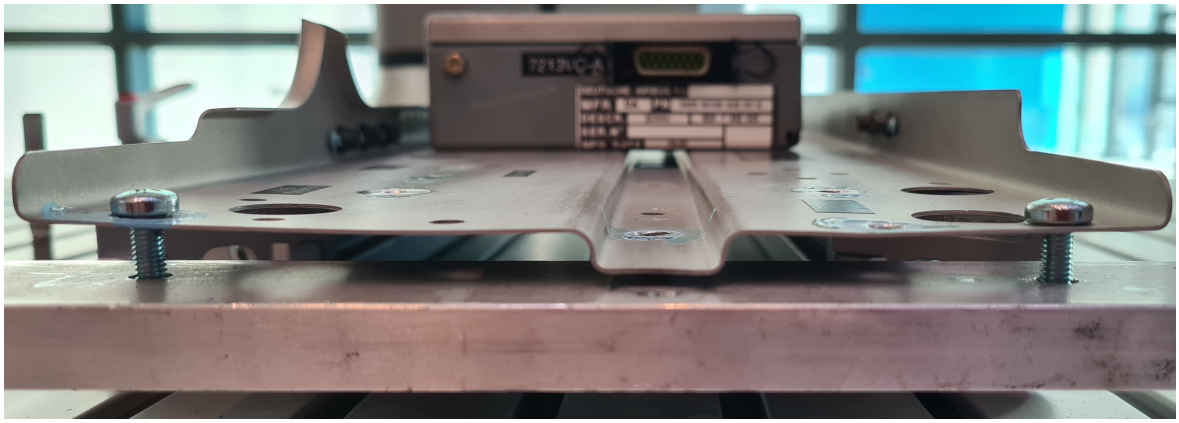


Abbildung 4.5: Die verwendete DEU

Beim Zusammenschrauben der HFork und der DEU liegen, wie in Abbildung 4.6a abgebildet, die Bauteile nicht glatt aufeinander. Stattdessen gibt es einen Luftspalt zwischen HFork und DEU an den Punkten der Schraubverbindung. Beim Festziehen der Schrauben schließt sich dieser Luftspalt und es kommt, wie in Abbildung 4.6b zu sehen ist, zu einer elastischen Verformung der DEU.

Die Produktion ist dabei in 5 Prozesse aufgeteilt: Ablage VFork, Ablage XRodA, Ablage XRodB, Zusammenschrauben von HFork und DEU und Ablage von HFork und DEU. Diese Prozesse werden einzeln zur Ausschreibung gestellt. Die UR10e AAS gibt im Rahmen des Anwendungsfalls als einzige ein Angebot ab und erhält somit den Zuschlag für diese Prozesse. Zur Ausführung der einzelnen Prozesse wird durch die UR10e AAS ein weiteres Ausschreibungsverfahren zur Bereitstellung der Einzelteile gestartet. Auf diese Ausschreibung antwortet die Lagerplatz AAS und erhält somit den Auftrag für die Bereitstellungsprozesse. Der Lagerplatz teilt darüber die Position der benötigten Einzelteile mit. Anschließend werden die Prozesse ausgeführt, wobei der Roboter



(a) Vorgeschraubte Zustand

(b) Festgezogene Zustand

Abbildung 4.6: Ansicht der DEU und HFork im vorgeschraubten und festgezogenen Zustand

einer Reihe von Befehlen folgt, um die für die Prozessausführung notwendigen Positionen zu erreichen. Die einzelnen Prozesse sowie die jeweiligen Positionen, welche zur Ausführung angefahren werden müssen, sind in Tabelle 4.1 dargestellt.

Die Ablageprozesse bestehen jeweils aus 2 Positionen, der Position, von welcher das Bauteil aufgenommen wird, und der Position, wo es abgelegt wird. Zur Ausführung der Prozesse zur Ablage von Teilen ist die Fähigkeit Aufnehmen und Ablegen notwendig. Für den Schritt des Zusammenschraubens muss die Position P11 zweimal eingefahren werden, um die Schrauben für die Löcher an Position P12 und P13 aus der Schraubenzufuhr zu entnehmen. P12 und P13 umfassen die notwendigen Befehle, um die Schrauben an P12 und P13 festzuziehen. Der Schraubprozess benötigt die Fähigkeit Schrauben. Die Fähigkeit Schrauben beschreibt hierbei sowohl das spezifische Einschrauben von Schrauben als auch die Fähigkeit, die zusammenzuschraubenden Teile in einen Arbeitsbereich zu bewegen.

Für den spezifischen Versuchsaufbau entstehen Signalverzögerungen in der Kommunikation der einzelnen AAS. Diese Signalverzögerung in der Kommunikation zwischen den AAS wurde 50 mal gemessen und der Durchschnitt bestimmt. Das Ergebnis war eine Signalverzögerung von $3ms$ mit einer Standardabweichung von $0,6ms$. In das in Abschnitt 3.3 beschriebene Modell der Kommunikationsinfrastruktur wurde die Signalverzögerung von $3ms$ als Dauerzusicherung eingepflegt.

Tabelle 4.1: Ablauf des Gesamtprozesses

Produktions-schritt	Beschreibung	Positionen
VFork	Das Einzelteil VFork wird aus dem Lager entnommen und auf der Vorrichtung abgelegt.	P4, P8
XRodA	Das Einzelteil XRodA wird aus dem Lager entnommen und auf der Vorrichtung abgelegt.	P3, P5
XRodB	Das Einzelteil XRodB wird aus dem Lager entnommen und auf der Vorrichtung abgelegt.	P1, P7
Schrauben	Die Einzelteile HFork und DEU werden aus dem Lager entnommen und im Arbeitsbereich abgelegt. Der Schraubenzieher entnimmt 2 Schrauben aus der Schraubenzufuhr und setzt diese in die jeweiligen Löcher ein. Der Schraubenzieher dreht die Schrauben fest. Die zusammengeschraubte HFork und DEU wird eingelagert.	P2, P9, P14, P10, P11, P12, P11, P13, P12t, P13t
HFork+DEU	Das Einzelteil HFork und DEU wird aus dem Lager entnommen und auf der Vorrichtung abgelegt.	P9, P6

4.2 Modell des Produktionssystems

Auf der Basis der in Abbildung 3.1 dargestellten allgemeinen Struktur wurde ein spezifisches Produktionssystem modelliert. Dieses ist in Abbildung 4.7 abgebildet.

Die Kommunikationsebene ist durch das in Abschnitt 3.3 beschriebene Modell dargestellt. Hierbei wurde für jede der verbundenen AAS ein eigener Port ergänzt, um Nachrichten an diese verschicken zu können.

Die AAS-Ebene umfasst drei AAS. Diese sind: die UR10e AAS, die Produkt AAS und die „Storage AAS“ (dt. Lagerplatz AAS). Die UR10e AAS und der Lagerplatz AAS leiten sich aus der Dienstanbieter AAS ab. Sie verfügen über eine spezifische Implementierung des südgerichteten Integrators. Die Produkt AAS leitet sich aus der Dienstanfrage AAS ab.

Auf der Asset-Ebene wurden nur die Ressourcen modelliert. Ein Modell des Produkts wurde im Rahmen dieser Arbeit nicht erstellt. Die modellierten Ressourcen sind der UR10e sowie der Lagerplatz. Diese sind über Konnektoren mit ihren jeweiligen AAS verbunden. Das UR10e-Modell spiegelt die Bewegung des UR10e wieder und ist über Konnektoren mit den Endeffektoren verbunden, die als eigene Modelle abgebildet sind. Das Lagerplatzmodell ist lediglich ein Abbild des momentanen Zustands des Lagerplatzes. Es zeigt an, welche Plätze momentan mit Teilen belegt sind und welche leer sind.

Sowohl der UR10e als auch der Lagerplatz sind entsprechend Abbildung 4.8 als spezifische Implementierungen des Ressourcenblocks modelliert.

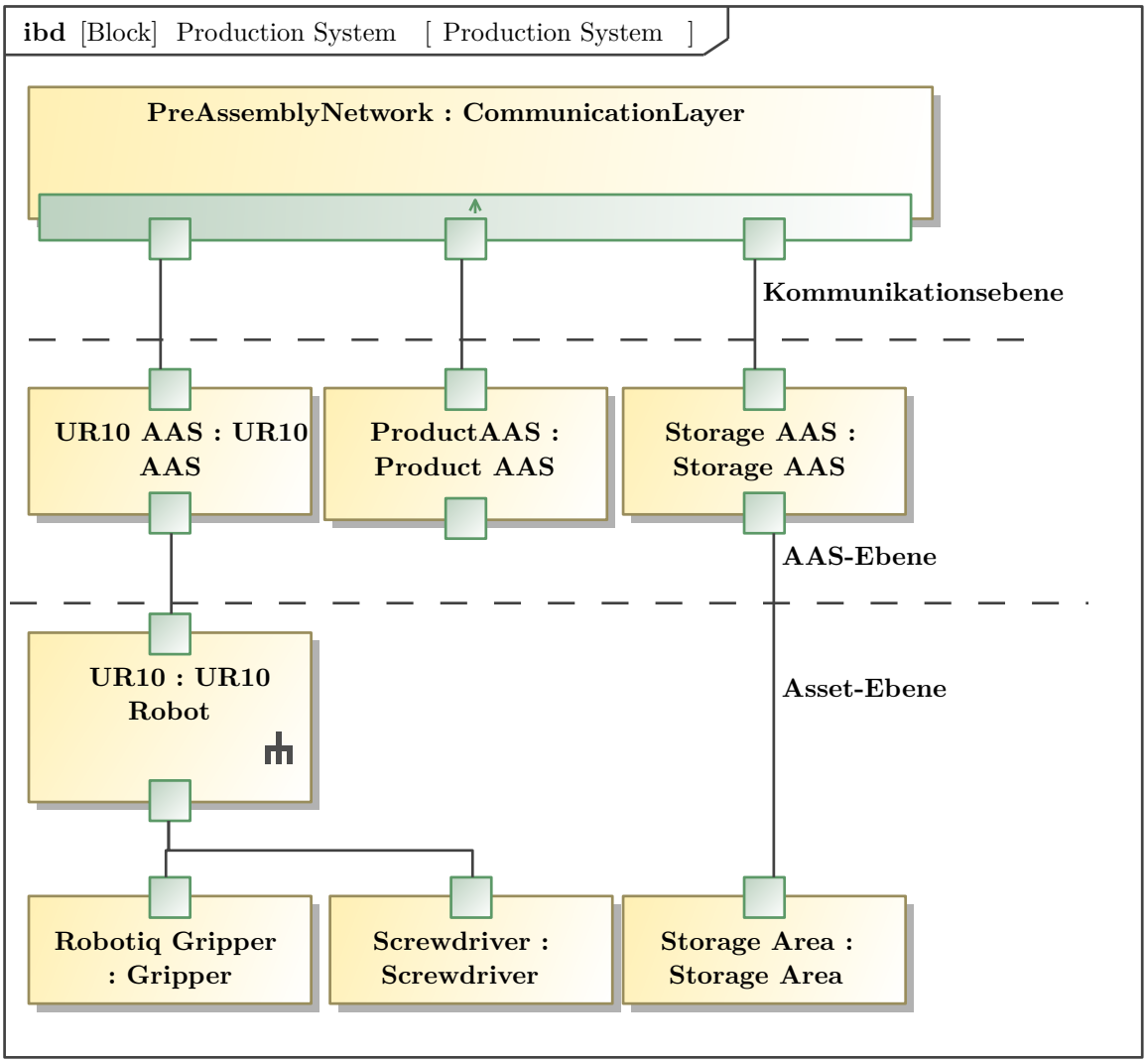


Abbildung 4.7: Internes Blockdiagramm des Produktionssystems

4.3 UR10e Roboterarm

Der UR10e-Roboter ist einer der in dieser Arbeit modellierten Ressourcen. Ein Abbild des Roboters im Modell ist in Abbildung 4.8 zu sehen. Er besteht aus zwei Teilen: dem Endeffektor sowie dem Roboterarm. Diese beiden Teile werden als individuelle Systemkomponenten im in Abbildung 4.7 abgebildeten Produktionssystem dargestellt. Die Endeffektoren sind über jeweilige Ports und Konnektoren mit dem Roboterarm verbunden.

Um dies zu ermöglichen, erweitert der Roboterarm die technische Ressource um den „EndeffectorConnector“ (dt. Endeffektor-Konnektor). Über diesen werden Befehle an die Endeffektoren übertragen.

Neben dem zusätzlichen Port verfügt der Endeffektor über eine Reihe von Werten zur Beschreibung seines Zustandes. Diese sind in Tabelle 4.2 abgebildet.

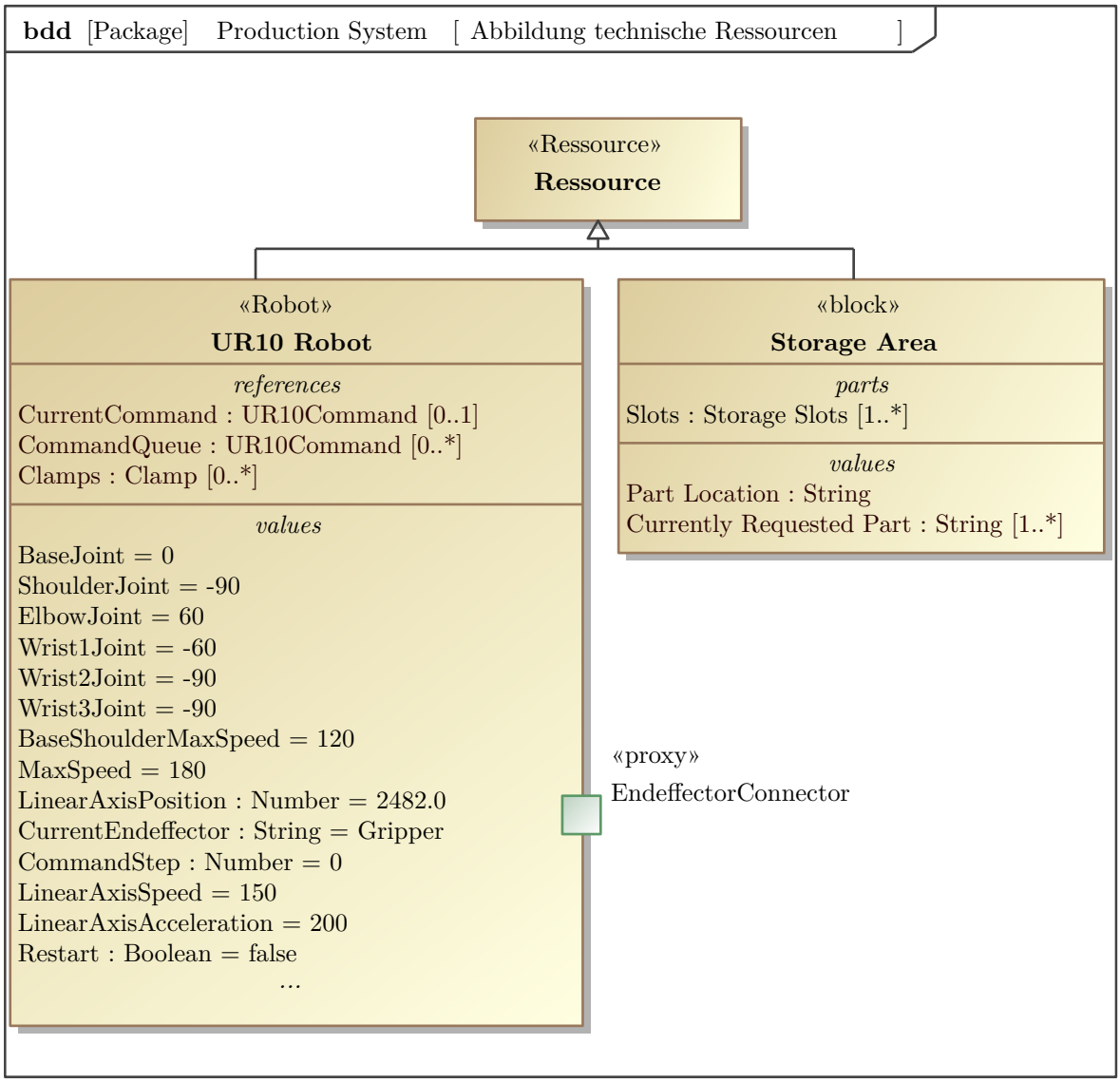


Abbildung 4.8: Blockdefinitionsdiagramm der technischen Ressourcen

Die in Abbildung 4.8 abgebildeten Standardwerte für die jeweiligen Gelenkpositionen beschreiben die Startposition des Roboters. Das Verhalten des Roboterarms wird durch das in Abbildung 4.9 dargestellte Zustandsdiagramm modelliert. Der Roboterarm verfügt über einen Startzustand. Von diesem geht er über in den Zustand „Start Endeffektor“ (dt. Start Endeffektor). Dieser Zustand sendet ein Signal an den zu diesem Zeitpunkt montierten Endeffektor, um diesen ebenfalls zu starten.

Es folgt darauf eine Wertabfrage. In einem normalen Startvorgang ist die Eigenschaft „Restart“ (dt. Neustart) auf „false“ (dt. Falsch) gesetzt. Der Roboter geht somit in den Zustand „Wait for Command“ (dt. Warte auf Befehl) über, in welchem er solange verbleibt, bis ein Befehl bei ihm eingeht. Geht ein Befehl ein, beginnt der Roboter mit der Abarbeitung der Befehlsliste. Bei einem Neustart des UR10e ist der Wert Neustart auf „true“ (dt. Wahr) gesetzt. In diesem Fall setzt der UR10e sofort die Abarbeitung der Befehlliste fort.

Tabelle 4.2: Physikalische Werte des UR10e

Wert	**Beschreibung**	**Einheit**
Base-, Shoulder-, Elbow-, Wrist1-, Wrist2-, Wrist3Joint	Diese Werte beschreiben die Positionen der jeweiligen Gelenke des Roboterarms. Dabei wird die Position als der Winkel ausgedrückt um welchen das Gelenk zur Nullposition verschoben ist.	$^\circ$
BaseShoulderMaxSpeed & MaxSpeed	Diese Werte beschreiben die maximal mögliche Geschwindigkeit der Gelenke. Die Maximalgeschwindigkeit des Base-Gelenks und Shoulder-Gelenks weicht von dem der Restlichen[41] ab und ist gesondert aufgeführt.	$\frac{^\circ}{s}$
LinearAxisPosition	Dieser Wert beschreibt die momentane Position des Roboters auf der Linearachse ausgehend von der linken Seite der Vormontagezelle.	mm
LinearAxisSpeed	Dies beschreibt die Geschwindigkeit der Positionsänderung auf der Linearachse.	$\frac{m}{s}$
LinearAxisAcceleration	Hierbei handelt es sich um die Beschleunigung der Positionsänderung auf der Linearachse	$\frac{m}{s^2}$
CurrentEndeffector	Dieser Wert gibt an welcher Endeffektor zum momentanen Zeitpunkt am Roboterarm angebracht ist.	-
CommandStep	Dies gibt an in welchem Schritt der Befehlschlange sich der Roboter momentan befindet.	-
Restart	Dieser Wahrheitswert gibt an ob sich der Roboter momentan in eine Neustartzustand befindet.	-

Ein beim UR10e eintreffender Befehl besteht aus einer Liste an individuellen Bewegungsbefehlen. Im Zustand „Resolve Command Queue“ (dt. Befehlswarteschlange Auflösen) wird diese Liste ausgelesen und der aktuelle Bewegungsbefehl ausgewählt. Wurde die Liste komplett bearbeitet, geht der UR10e wieder in den Warte auf Befehl Zustand über.

Im Zustand „Execute Command“ (dt. Befehlsausführung) werden die einzelnen Bewegungsbefehle ausgeführt. Nach der erfolgreichen Ausführung eines Bewegungsbefehls fährt der UR10e mit dem nächsten Bewegungsbefehl fort. Sollte ein Bewegungsbefehl einen Neustart des UR10es benötigen, wird das „UR10 Restart“ (dt. UR10 Neustart)

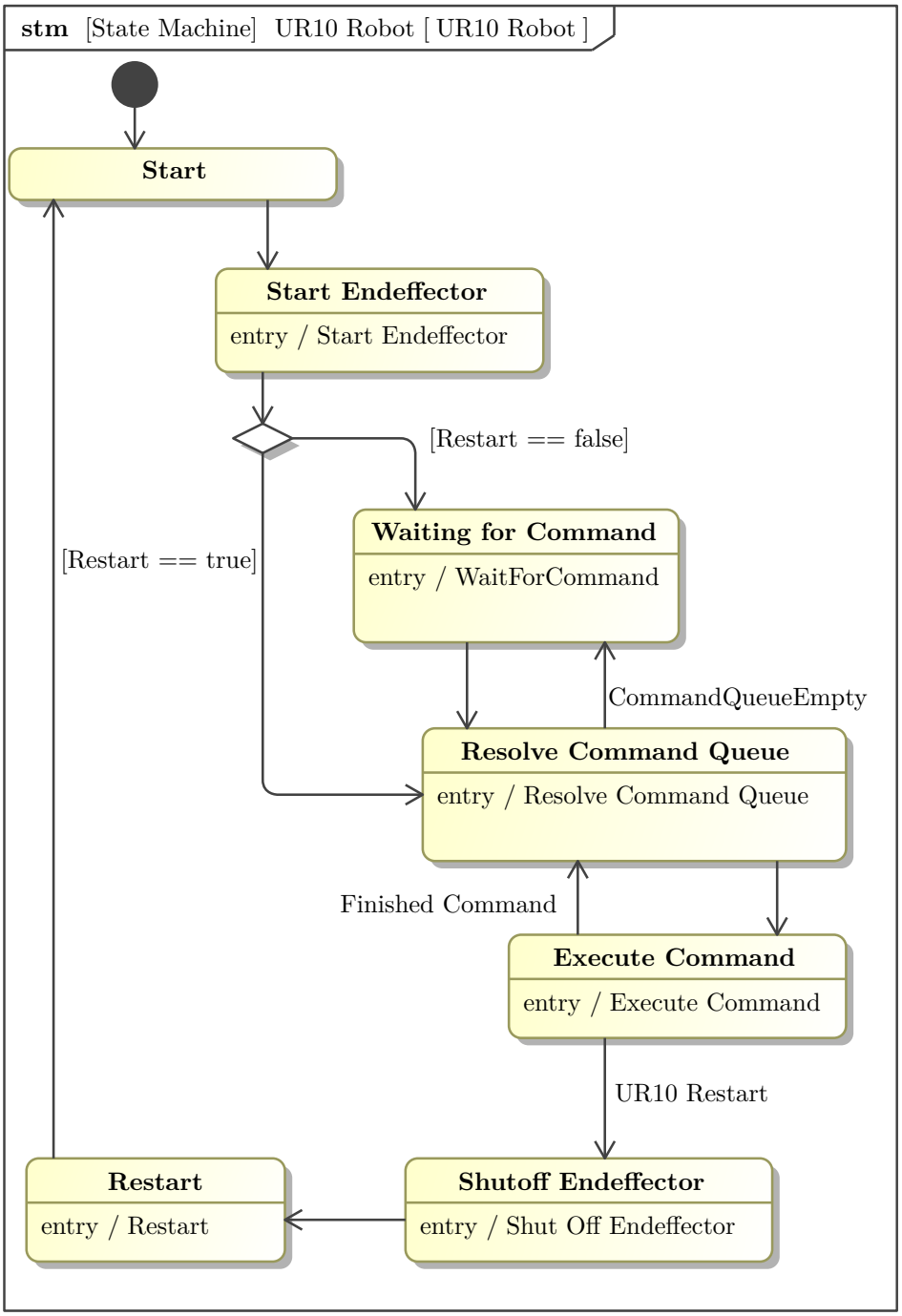


Abbildung 4.9: Zustandsmaschine des UR10e Roboters

Signal ausgelöst. Daraufhin wird der Endeffektor ausgeschaltet und ein Neustart durchgeführt.

Bewegungsbefehle verfügen über eine Reihe an Werten, welche die auszuführende Bewegung beschreiben. Zum einen besitzen Bewegungsbefehle einen Typ. Dieser definiert, welche Bewegung auszuführen ist. Die restlichen, dem Bewegungsbefehl beiliegenden Werte, sind abhängig vom Befehlstyp.

Abbildung 4.10 zeigt das Aktivitätsdiagramm, welches die Ausführung der Bewegungs-

befehle beschreibt. Das Aktivitätsdiagramm lässt sich in 6 Bereiche einteilen, wobei jeder Bereich eine spezifische Rolle in der Ausführung von Befehlen einnimmt. Bereich 0 und Bereich 6 sind allgemeine Bereiche. In Bereich 0 wird die Ausführung gestartet. Zu diesem Zweck wird der Befehltyp ausgelesen, so dass anschließend die dem Typ entsprechenden Aktionen ausgeführt werden können. Bereich 6 ist der Endbereich. Dieser überträgt das Signal „Finished Command“ (dt. Befehl Fertig). Bereich 1 bis Bereich 5 sind jeweilige Befehlstypenbereiche, welche eine Art von Bewegung modellieren.

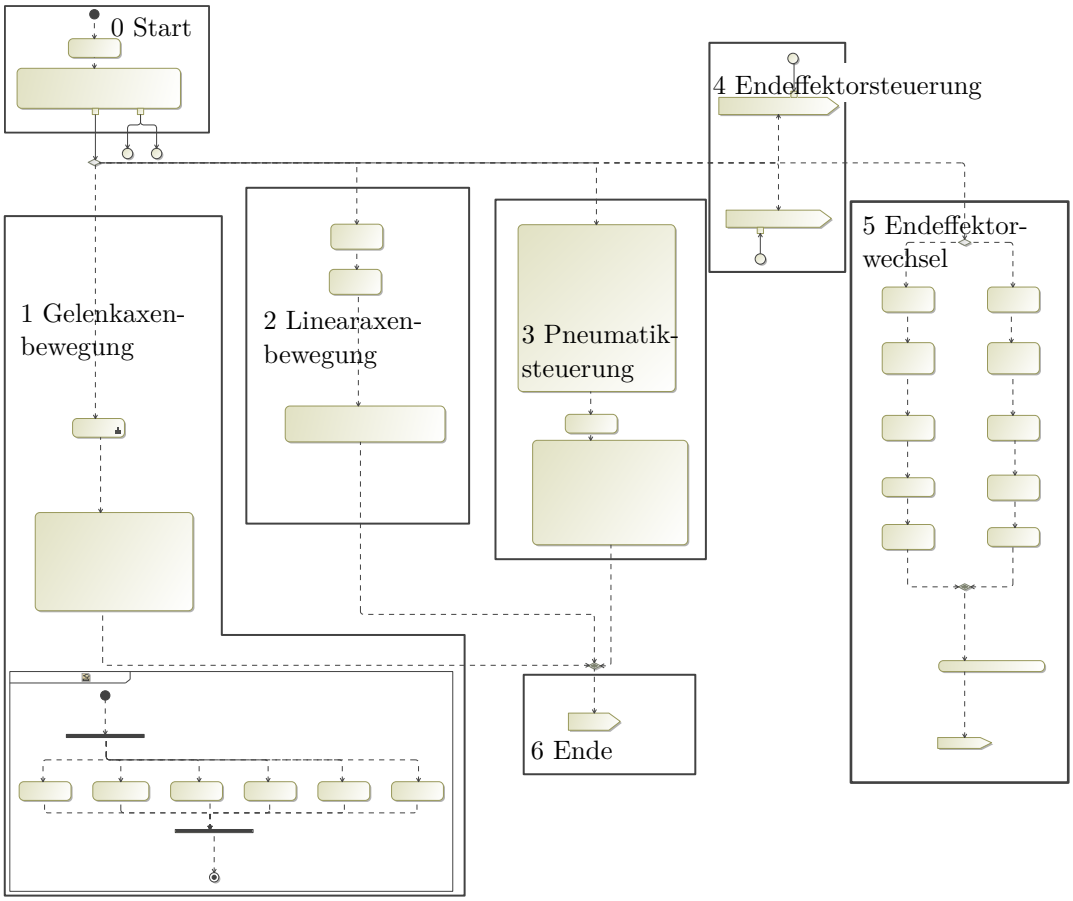


Abbildung 4.10: Aktivitätsdiagramm der Ausführung von Befehlen durch den UR10e Roboter

In Bereich 1 ist ebenfalls eine zusätzliche Verzögerung simuliert, die sich experimentell ergeben hat. Diese wurde dadurch bestimmt, dass dem Roboter Befehle übertragen wurden, zu einer Position zu fahren, in welcher er bereits war. Die Zeit, bis der Roboter eine erfolgreiche Ausführung des Befehls meldete, wurde gemessen. Dies wurde 20-mal ausgeführt und der Durchschnitt gebildet. Daraus ergab sich eine Zeit von $15ms$ mit einer Standardabweichung von $4ms$. Der Bereich 0 umfasst somit eine Aktion mit einer Dauerzusicherung von $15ms$, um diese Verzögerung im Modell abzubilden.

4.3.1 Gelenkachsenbewegung

Bereich 1 modelliert die Bewegung der Gelenkachsen. Hierbei handelt es sich um eine mathematische Beschreibung der in Abschnitt 2.6 vorgestellten Bewegung. Diese Bewegung ist pro Achse durch 4 Werte beschrieben. Diese sind die Momentanposition d_{mi} und die Zielposition d_{zi} einer Achse sowie die Geschwindigkeit v und Beschleunigung a der Achsen. Wobei die Beschleunigung und Geschwindigkeit für jede Achse gleich sind.

Die Bewegung jeder Achse wird in der Modellierung individuell und voneinander unabhängig betrachtet.

Wie in Abschnitt 2.6 beschrieben, besteht das Bewegungsprofil der Gelenke eines UR10e Roboters aus den Abschnitten Beschleunigungsphase, Bewegungsphase und Entschleunigungsphase. Da die Bewegungsphase bei kleinen Distanzen entfallen kann, müssen zur Bestimmung der benötigten Zeit zwei Fälle betrachtet werden.

Der erste Fall ist, dass die gewünschte Entfernung nicht innerhalb der Beschleunigungs- und Entschleunigungsphase zurückgelegt werden kann. Hierbei ergibt sich die Zeit aus der in Formel 2.1 beschriebenen Summe der Zeiten der einzelnen Bewegungen. Die Zeit der Beschleunigungs- und Entschleunigungsphase ist dabei die, die benötigt wird, um auf die gewünschte Geschwindigkeit zu beschleunigen. Die Zeit der Bewegungsphase ergibt sich aus der Zeit, welche benötigt wird, um die Entfernung abzüglich der Distanz der Beschleunigungs- und Entschleunigungsphase bei der gegebenen Geschwindigkeit zurückzulegen.

Der zweite Fall ist, dass die Entfernung bereits in der Beschleunigungs- und Entschleunigungsphase zurückgelegt wird. Hierbei muss die in Formel 2.2 angegebene Gleichung nach der Zeit umgestellt werden. d_a ist für diesen Fall die Hälfte der zurückzulegenden Entfernung. Anschließend kann wieder die Formel 2.1 angewendet werden, wobei $t_v = 0$ ist.

Die Zeit zur Bewegung der individuellen Gelenkachsen t_i lässt sich somit entsprechend Formel 4.1 beschreiben. Die obere Bedingung beschreibt die Bewegung, sollte die Distanz zur Erreichung der gewünschten Geschwindigkeit ausreichen. Die obere beschreibt die Bewegung, wenn dies nicht der Fall ist.

$$t_i = \begin{cases} 2 * \frac{v}{a} + \frac{|d_{mi} - d_{zi}| - a * (\frac{v}{a})^2}{v}, & |d_{mi} - d_{zi}| > a * (\frac{v}{a})^2 \\ 2 * \sqrt{\frac{|d_{mi} - d_{zi}|}{a}}, & |d_{mi} - d_{zi}| \leq a * (\frac{v}{a})^2 \end{cases} \tag{4.1}$$

Formel 4.1 wurde im Rahmen eines Parameterdiagramms für das Modell umgesetzt. Wie in Abschnitt 2.6 beschrieben, regelt der Roboter die Bewegung so, dass alle Achsen die gleiche Zeit zur Bewegung benötigen. Dies ist in der Simulation nicht abgebildet, da es lediglich von Interesse ist, die Gesamtdauer der Bewegung abzubilden. Jede Achse wird somit simuliert, als würde sie sich so schnell wie möglich unter Einhaltung der Parameter bewegen. Dies wird umgesetzt durch die Erstellung eines Zweigs im Zustandsdiagramm für jede individuelle Achse. Alle Zweige werden parallel ausgeführt und umfassen eine Aktion, bei welcher die Zeit zur Bewegung der jeweiligen Achse als

Dauerzusicherung beigelegt ist. Die Zweige werden anschließend wieder zusammengeführt. Die Simulation fährt erst fort, wenn alle Zweige ausgeführt wurden. Die Dauer der Bewegung innerhalb der Simulation ist somit gleich der Zeit, welche die langsamste Achse benötigt.

4.3.2 Linearachsenbewegung

Bereich 2 in Abbildung 4.10 umfasst die Linearachsenbewegung. Sie wird durch die Momentanposition, die Zielposition sowie die Beschleunigung und Geschwindigkeit beschrieben. Die Werte sind hier in mm, $\frac{mm}{s}$ und $\frac{mm}{s^2}$ angegeben.

Es wird in diesem Fall das gleiche Bewegungsmodell wie in Abschnitt 4.3.1 angewendet. Jedoch gibt es hier nur eine Achse und somit nur eine Aktion, welche mit der Dauer als Dauerzusicherung versehen wird.

Des Weiteren wurde festgestellt, dass die Linearachsenbewegung eine zusätzliche Verzögerung beinhaltet. Bei dieser Verzögerung handelt es sich um Signal- und Rechenzeit von zum einen dem UR10e und zum anderen der SPS, welche die Linearachse steuert. Die Verzögerung wurde experimentell bestimmt, durch die Übertragung des Befehls, die Linearachse zu einer Position zu bewegen, an welche diese bereits ist. Dies wurde 20-mal wiederholt und der Durchschnitt aus den Zeiten gebildet. Zusätzlich wurden die in Abschnitt 2.6 beschriebenen $15ms$ Verzögerung abgezogen. Daraus ergibt sich eine Verzögerung für die Linearachse von einer Sekunde. Welche hier ebenfalls als Aktion mit einer Dauerzusicherung abgebildet ist.

4.3.3 Pneumatiksteuerung

Bereich 3 ist die Steuerung der Pneumatik. Diese wird definiert durch einen Klammerindex, welcher die anzusteuernde Klammer identifiziert sowie einen Wahrheitswert, welcher angibt, ob die Klammer auf oder zu sein soll.

Der reale Versuchsaufbau umfasst keine Möglichkeit zur Überprüfung des momentanen Zustands der jeweiligen Klammern. Es wird daher nach jeder Ansteuerung der Klammer 2 Sekunden lang gewartet, um ihnen zu erlauben, sich zu schließen. Diese 2 Sekunden sind als Dauerzusicherung einer Aktion beigefügt. Die restlichen Aktionen befassen sich lediglich mit der Umstellung der Klammer auf den gegebenen Wahrheitswert.

4.3.4 Endeffektorsteuerung

Bereich 4 zeigt die Steuerung der Endeffektoren. Dies passiert hier nicht innerhalb des in Abbildung 4.10 gezeigten Aktivitätsdiagramms. Stattdessen kommt es hier lediglich zur Übertragung des Signals an die jeweiligen Endeffektoren.

Die Endeffektoren sind entsprechend des Blockdefinitionsdiagramms in Abbildung 4.11 modelliert. Es existiert ein allgemeiner Endeffektorblock, der über einen „Robot-Connector“

(dt. Roboter-Konnektor)-Port verfügt. Dies ermöglicht die Anbindung an den Roboterarm. Die individuellen Endeffektoren leiten sich aus diesem allgemeinen Block ab.

Eine Gemeinsamkeit der Endeffektoren ist dabei die Referenz auf das „CurrentCommand“ (dt. momentaner Befehl). Diese Information wird über den Port übertragen und bietet die notwendigen Informationen zur Ausführung der gewünschten Bewegung.

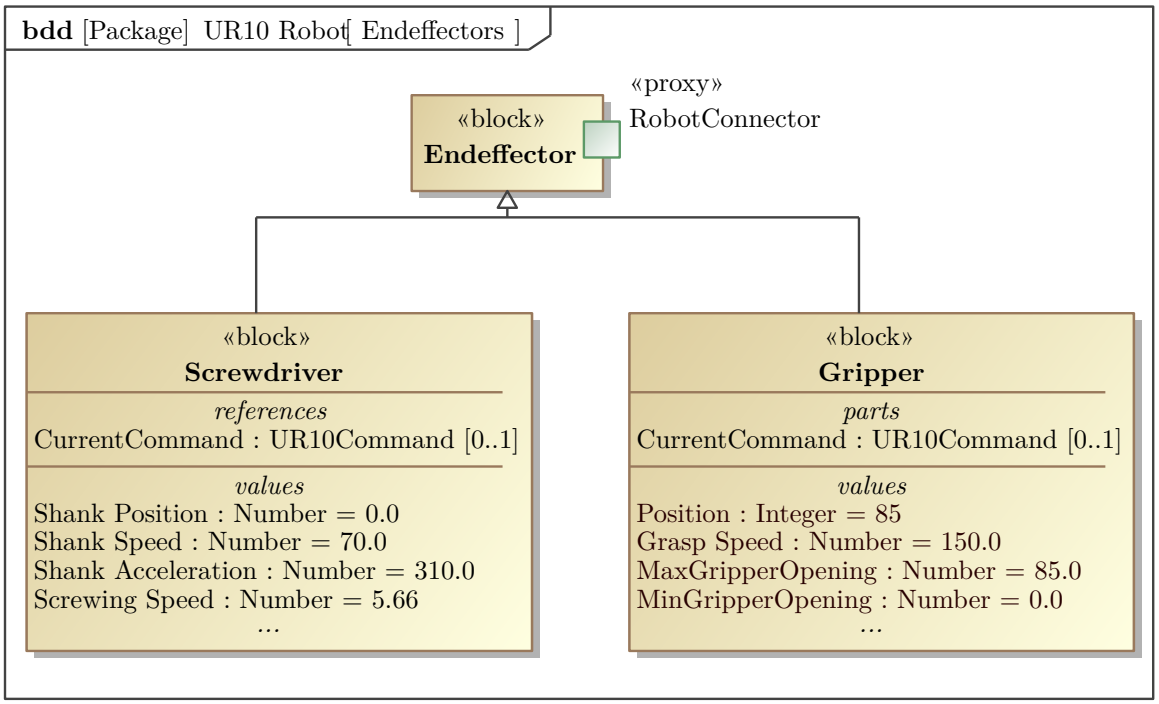


Abbildung 4.11: Blockdefinition der Endeffektoren

Der Greifer und Schraubenzieher definieren sich über eine Reihe von Werten, die ihren jeweiligen physikalischen Zustand beschreiben. Für den Greifer ist dies der Wert Position. Er bildet die Öffnung des Greifers in mm ab. „MaxGripperOpening“ und „MinGripperOpening“ bilden dabei die obere und untere Limitierung dieses Werts. Der Wert „Grasp Speed“ gibt in $\frac{mm}{s}$ an, wie schnell der Greifer sich schließen kann.

Der Schraubenzieher ist definiert durch die „Shank Position“ in mm. Diese gibt an, an welcher Position sich der interne Schaft befindet. „Shank Speed“ ($\frac{mm}{s}$) und „Shank Acceleration“ ($\frac{mm}{s^2}$) beschreiben die Bewegung des Schafts. Der Wert „Screwing Speed“ gibt die vertikale Bewegung des Schaftes während des Schraubvorgangs in $\frac{mm}{s}$ an.

Das Verhalten von sowohl Greifer als auch Schraubenzieher ist weitgehend ähnlich modelliert. Das Zustandsdiagramm des Greifers ist in Abbildung 4.12 dargestellt. Der Greifer beginnt im Zustand „Gripper Turned Off“ (dt. Greifer ausgeschaltet). Der Greifer ist in diesem Zustand ausgeschaltet und nicht aktiv. Nach Erhalt des Signals „Turn on Gripper“ (dt. Schalte Greifer ein) über den Roboter-Konnektor-Port geht der Greifer in einen aktiven Zustand über. Im Zustand „WaitForCommand“ (dt. Warte auf Befehl) verweilt er, bis ein weiteres Signal eingeht. Sollte er das Signal „Turn Off Endeffector“ (dt. Schalte Endeffektor aus) erhalten, schaltet sich der Greifer wieder ab. Das Signal

„MoveGripper“ (dt. Bewege Greifer) führt zur Ausführung des an dieses Signal angehängten Befehls. Nach erfolgreicher Ausführung wartet der Greifer erneut auf den nächsten Befehl. Das Signal Bewege Greifer wird in Bereich 4 des in Abbildung 4.10 dargestellten Aktivitätsdiagramms übermittelt. Die Signale Schalte Endeffektor aus und Schalte Greifer ein werden in Abbildung 4.9 als Teil des Start- und Neustartvorgangs übertragen.

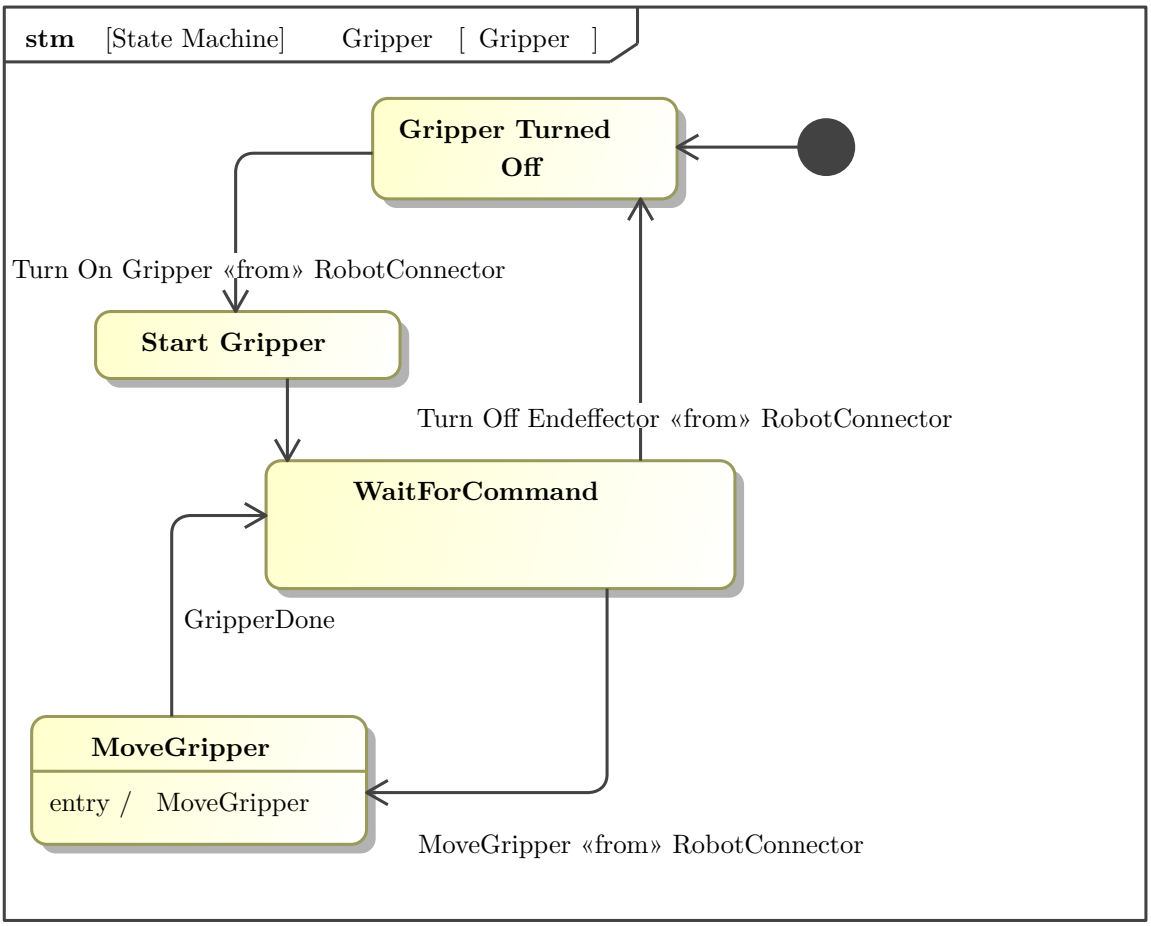


Abbildung 4.12: Zustandsdiagramm des Grippers

Das Zustandsdiagramm des Schraubenziehers ist nahezu exakt gleich aufgebaut, mit lediglich einer abweichenden Benennung der Zustände und Signale.

Die tatsächliche Ausführung des Befehls erfolgt ähnlich wie in Abschnitt 4.3 innerhalb eines Aktivitätsdiagramms, welches bei Eintritt in den Zustand Bewege Greifer ausgeführt wird. Wie in Abschnitt 4.3.1 wurden die notwendigen Zeiten mithilfe von Dauerzusicherungen modelliert.

Die Zeit zur Bewegung des Greifers wird berechnet anhand der Momentanposition d_m der Zielposition d_z sowie der Geschwindigkeit v. Die Zeit t der Bewegung des Greifers ergibt sich aus Formel 4.2.

$$t = \frac{|d_m - d_z|}{150\frac{mm}{s}} \tag{4.2}$$

Die in Abschnitt 2.6 beschriebenen Bewegungsformen des Schraubenziehers sind jeweils einzeln modelliert. Die Unterscheidung zwischen diesen Bewegungen erfolgt durch eine zusätzliche Variable, die als Teil des Befehls übertragen wird.

Die Bewegungen Bewege Schaft und Aufheben wurden gleich modelliert. Es wurde hier wieder das in Abschnitt 4.3.1 beschriebene Bewegungsmodell angewendet. Da die Geschwindigkeit und Beschleunigung des Schafts nicht in der technischen Spezifikation des Schraubenziehers angegeben sind[44], war es notwendig, diese Werte experimentell zu bestimmen.

Zu diesem Zweck wurde innerhalb des in Abschnitt 4.1 gezeigten Versuchsaufbaus eine Untersuchung dieser Werte durchgeführt. In $5mm$ Schritten von $5mm$ bis $50mm$ wurden die Zeiten gemessen, welche der Schaft zur Zurücklegung dieser Entfernungen benötigt. Die Ergebnisse dieser Messung sind in Blau in Abbildung 4.13 dargestellt.

Die gemessenen Werte wurden mit den errechneten Werten verglichen und deren Abweichung bestimmt. Zur Berechnung der Werte wurden $a = 100\frac{mm}{s^2}$ und $v = 30\frac{mm}{s}$ als Anfangswerte gewählt. Mithilfe des Excel Add-In Programms „Solver“[50] wurden diese Anfangswerte optimiert, mit dem Ziel der Minimierung des quadratischen durchschnittlichen Fehlers zwischen gemessenen und berechneten Werten. Die gewählte Lösungsmethode war GRG Nonlinear. Die Geschwindigkeit des Schafts wurde mithilfe dieser Methode als $300\frac{mm}{s}$ und die Beschleunigung als $70\frac{mm}{s^2}$ bestimmt. Die errechneten Zeiten auf der Basis dieser Werte sind in Abbildung 4.13 in Rot dargestellt.

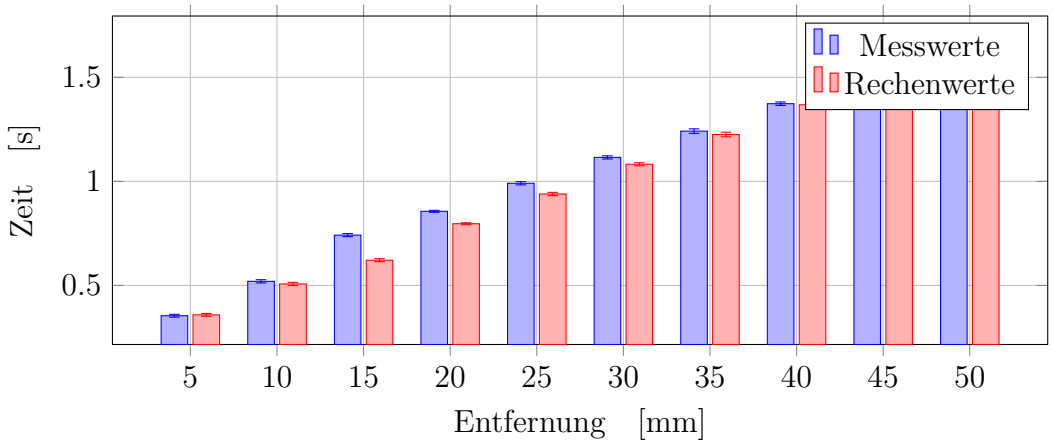


Abbildung 4.13: Dauer der Bewegung des Schaftes für verschiedene Entfernungen

Die Bewegungen Festziehen und Vorschrauben wurden anhand der Schraubengeometrie modelliert.

Für das Vorschrauben ergibt sich somit die benötigte Zeit aus der einzuschraubenden Länge l_s sowie der Geschwindigkeit des Schraubprozesses v_s. Entsprechend Abschnitt 2.7 ergibt sich diese Geschwindigkeit aus der Umdrehungszahl und der Ganghöhe der

Schraube. Für den Anwendungsfall wurde eine M5-Schraube eingesetzt, welche eine Ganghöhe von $P = 0,8mm$[45] hat. Die maximale Umdrehungszahl des Schraubenziehers ist $U = 340\frac{1}{\text{min}} = 5,66\frac{1}{\text{s}}$[44]. Dementsprechend ergibt sich nach Formel 2.4 eine Schraubgeschwindigkeit von $v_s = 4,528\frac{mm}{s}$.

Die Zeit t_s für den Vorschraubprozess berechnet sich somit nach Formel 4.3.

$$t_s = \frac{l_s}{4,528\frac{mm}{s}} \tag{4.3}$$

Die Zeit für den Prozess Festziehen t_f errechnet sich entsprechend Formel 4.4 aus der Zeit t_s, welche zum Reinschrauben der Länge der Schraube benötigt wird, sowie die Zeit t_M, bis das Nennmoment aufgebracht ist.

$$t_f = t_s + t_M \tag{4.4}$$

Die Zeit t_M wurde hier so modelliert, dass diese die Zeit ist, welche benötigt wird, um die Schraube um die elastische Verlängerung der Schraube einzuführen. Wie in Abschnitt 2.7 beschrieben, ist die elastische Verlängerung abhängig von der Schraubenkraft und der elastischen Nachgiebigkeit der Schraube. In dieser Arbeit wurde eine standardisierte M5-Schraube eingesetzt. Diese ist gefertigt aus Stahl, welches ein Elastizitätsmodul von $210.000\frac{N}{mm^2}$[45] besitzt. Basierend darauf ergibt sich eine elastische Nachgiebigkeit von $\delta = 2,36 * 10^{-6}\frac{mm}{N}$.

Die Schraubenkraft hängt entsprechend der Formel 2.5 in Abschnitt 2.7 direkt mit dem Anzugsmoment zusammen. Der Flankendurchmesser d_2 ergibt sich aus der Schraubengeometrie und beträgt $d_2 = 4,480mm$. Der Flankenwinkel ϕ beträgt für metrische Gewinde $\phi = 60°$. Der Reibungswinkel ρ' ist abhängig vom Reibwert μ, für den $\mu = 0,15$ gewählt wurde. Der Reibungswinkel beträgt somit $\rho' = \arctan(\mu) = 8,53°$[45]. Die Schraubenkraft in Abhängigkeit des Gewindenanzugsmomentes lässt sich somit für diese Schraubenverbindung entsprechend Formel 4.5 darstellen.

$$F_S = \frac{M_G}{5,7 * 10^{-3}m} \tag{4.5}$$

Für das maximale Anzugsmoment des Schraubenziehers von $M_G = 5Nm$. Ergibt sich somit eine Verlängerung von $f = 0,002mm$, welche einer Dauer zum Erreichen des Moments nach Formel 4.3 von $t_M = 4,58 * 10^{-4}s$ entspricht. Dies ist kleiner als die Schrittgröße der Zeitmessung innerhalb der Simulation und wird somit weiterführend vernachlässigt. Die Zeit für die Bewegung Festziehen ergibt sich somit ausschließlich aus der notwendigen Zeit zum Einschrauben des Gewindes.

4.3.5 Endeffektorwechsel

Aufgrund der Beschaffenheit der verfügbaren Ressourcen sowie des gewählten Anwendungsfalls ist ein Endeffektorwechsel durch den Menschen während der Prozessausführung notwendig. Dieser Endeffektorwechsel besteht aus 6 Schritten. 5 dieser Schritte sind in Abbildung 4.10 im Bereich 5 als Aktionen modelliert. Der letzte Schritt ist in Abbildung 4.9 im Zustand Neustart modelliert. Die Schritte für den Wechsel vom Greifer auf den Schraubenzieher sind: 1. Abbau des Greifers, 2. Holen des „OnRobot QuickChangers“, 3. Montieren des „OnRobot QuickChangers“, 4. Sicherung des Kabels, 5. Montieren des Schraubenziehers und 6. Rekonfiguration der UR10e Software. Diese Schritte wurden jeweils fünfmal durchgeführt und anschließend der Durchschnitt aus ihnen gebildet. Die Ergebnisse sind in Tabelle 4.3 aufgeführt.

Tabelle 4.3: Dauer des Wechsels von Greifer auf Schraubenzieher, 1. Abbau des Greifers, 2. Holen des „OnRobot QuickChangers“, 3. Montieren des „OnRobot QuickChangers“, 4. Sicherung des Kabels, 5. Montieren des Schraubenziehers, 6. Rekonfigurierung der UR10e Software

$[s]$	1	2	3	4	5	6	Summe
Zeit	106,112	22,442	46,996	39,376	7,944	187,384	410,254
Standardabweichung	18,206	9,501	12,507	5,71	0,709	9,054	55,688

Die Schritte zum Wechsel von Schraubenzieher auf Greifer sind: 1. Abbau des Schraubenziehers, 2. Demontage des „OnRobot QuickChangers“, 3. Entfernen des Kabels, 4. Verstauen des „OnRobot QuickChangers“, 5. Montieren des Grippers und 6. Rekonfigurierung der UR10e Software. Die gemessenen Werte der Schritte sind in Tabelle 4.4 dargestellt.

Tabelle 4.4: Dauer des Wechsels von Schraubenzieher auf Greifer, 1. Abbau des Schraubenziehers, 2. Demontage des „OnRobot QuickChangers“, 3. Entfernen des Kabels, 4. Verstauen des „OnRobot QuickChangers“, 5. Montieren des Grippers, 6. Rekonfigurierung der UR10e Software

$[s]$	1	2	3	4	5	6	Summe
Zeit	6,318	45,432	25,356	18,748	121,552	180,67	398,08
Standardabweichung	0,634	8,232	6,944	2,143	12,024	24,92	54,899

Insgesamt benötigt der Endeffektorwechsel in beiden Fällen insgesamt rund $400s$. Es ist jedoch ebenfalls zu erkennen, dass beide Fälle eine sehr hohe Standardabweichung von rund $50s$ aufweisen.

4.4 Lagerplatz

Der Lagerplatz ist die zweite modellierte Ressource. Anders als der UR10e-Roboter führt der Lagerplatz keine aktiven Produktionsschritte aus. Stattdessen übernimmt er

die wichtige logistische Aufgabe der Bereitstellung von Einzelteilen für die Produktion. Der Arbeitsplatz verfügt zu diesem Zweck über fest definierte Bereiche, in denen Einzelteile bereitliegen. Diese Bereiche sind, wie in Abbildung 4.8 zu sehen, als Slots modelliert. Ein Slot ist dabei definiert durch einen Positionsidentifikator, einen Objekttypen sowie den Wahrheitswert Belegt, welcher definiert, ob der Slot belegt ist oder nicht. Der Positionsidentifikator gibt an, an welcher Position sich der Slot befindet, während der Objekttyp definiert, welche Typen von Einzelteilen an einem Platz liegen können.

Das Verhalten des Lagerplatzes ist digitaler Natur. Der Lagerplatz ist lediglich in der Lage, Einzelteile in sich selbst zu finden und deren Position mitzuteilen sowie den eigenen Zustand zu verwalten. Das Verhalten ist durch das in Abbildung 4.14 dargestellte Zustandsdiagramm modelliert.

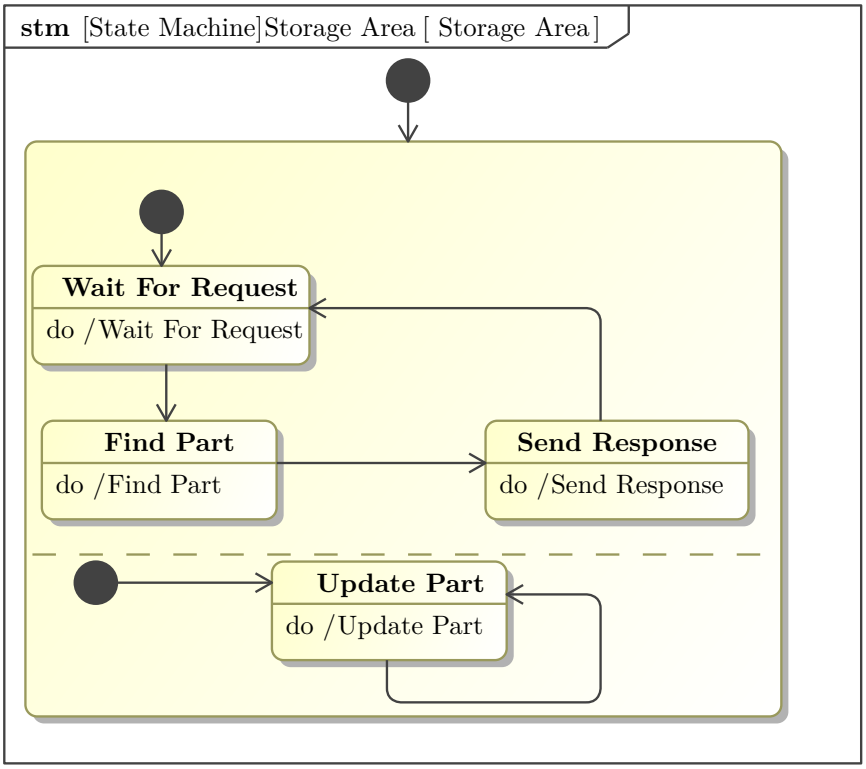


Abbildung 4.14: Zustandsdiagramm der Lagerfläche

Das Zustandsdiagramm besteht aus zwei Teilen, die parallel ablaufen. Der obere Bereich befasst sich mit dem Auffinden von Einzelteilen. Der Lagerplatz verweilt im Zustand „Wait for Request“ (dt. Warte auf Anfrage), bis eine Anfrage an ihn übermittelt wird. Eine Anfrage umfasst dabei die Namen eines oder mehrerer Einzelteile, welche vom Lagerplatz bereitzustellen sind. Der Lagerplatz durchsucht seine Slots nach einem Bereich, in dem die Einzelteile vom entsprechenden Typ vorhanden sind, und überträgt anschließend die Position dieser Bereiche.

Der untere Teil befasst sich lediglich mit der Zustandsverwaltung. Der Zustand „Update Part“ (dt. Aktualisiere Teil) wartet, bis eine Nachricht eingeht, dass ein Einzelteil aus

einem Slot entnommen wurde, und setzt daraufhin den Wert Belegt innerhalb des Slots auf falsch.

4.5 Anbindung der technischen Ressourcen an die Verwaltungsschale

Im Rahmen dieser Arbeit wurden zwei spezifische Integratoren modelliert, welche beide in einer Generalisierungsbeziehung zum südgerichteten Integratorblock stehen. Diese sind der UR10-Integrator und der Lagerflächen-Integrator.

Der Lagerflächen-Integrator verfügt über ein relativ simples eigenes Verhalten. Dies besteht lediglich aus dem Empfang von internen Signalen zur Ausführung eines Prozesses und der anschließenden Weiterleitung dieses Signals an den in Abschnitt 4.4 beschriebenen Lagerplatz.

Das Verhalten des UR10e-Integrators ist in Abbildung 4.15 abgebildet. Der UR10e-Integrator überprüft den auszuführenden Prozess und fährt fort, je nachdem, ob dieser Prozess ein Greifprozess oder ein Schraubprozess ist. Der Greifprozess ist in dem Zustand „Pick and Place“ (dt. Nehme und Platziere) abgebildet. Er besteht aus zwei Kompositzuständen. Dieser ist zum einen das Finden des im Prozess definierten Objekts. Dies wird erreicht durch das Starten eines neuen Ausschreibungsverfahrens für einen Prozess zur Bereitstellung des Objekts. Das Ergebnis der Ausführung des Bereitstellungsprozesses durch eine andere AAS ist der Ort des gewollten Objekts. Diese Information ist im Ausführungsprotokoll abgespeichert.

Zur Ausführung des Prozesses verfügt der UR10e-Integrator über eine Liste von Punkten, welche der UR10e-Roboterarm abfahren muss, um ein Objekt von einem spezifischen Punkt aufzunehmen. Diese Punkte sind so gespeichert, dass eine Ausführung der Bewegung vom ersten zum letzten Punkt dem Aufheben eines Objekts entsprechen würde. Das Abfahren der Punkte vom letzten zum ersten Punkt entspricht dem Ablegen eines Objekts. Der UR10e-Integrator entsendet dementsprechend Bewegungsbefehle an den UR10e, so dass das im Prozess spezifizierte Objekt von dem aus dem vorherigen Schritt in Erfahrung gebrachten Ort aufgenommen und in dem in der Prozessbeschreibung spezifizierten Ort abgelegt wird.

Zur Ausführung des Schraubenprozesses müssen, ähnlich dem Greifprozess, erst die Orte der benötigten Einzelteile in Erfahrung gebracht werden. Dies wird mit einer Bereitstellungsprozessausschreibung umgesetzt, welche die zusammenzuschraubenden Teile sowie den Ort der Schrauben anfragt. Auf der Basis des Ergebnisses dieser Ausschreibung wird anschließend eine Liste von Befehlen an den UR10e übertragen. Diese Liste besteht aus den folgenden Teilen: 1. Platzieren der beiden Einzelteile, 2. Wechsel des Endeffektors von Greifer auf Schraubenzieher, 3. Vorschrauben der Schraubverbindungen, 4. Festziehen der Schraubverbindungen, 5. Wechsel des Endeffektors vom Schraubenzieher zum Greifer.

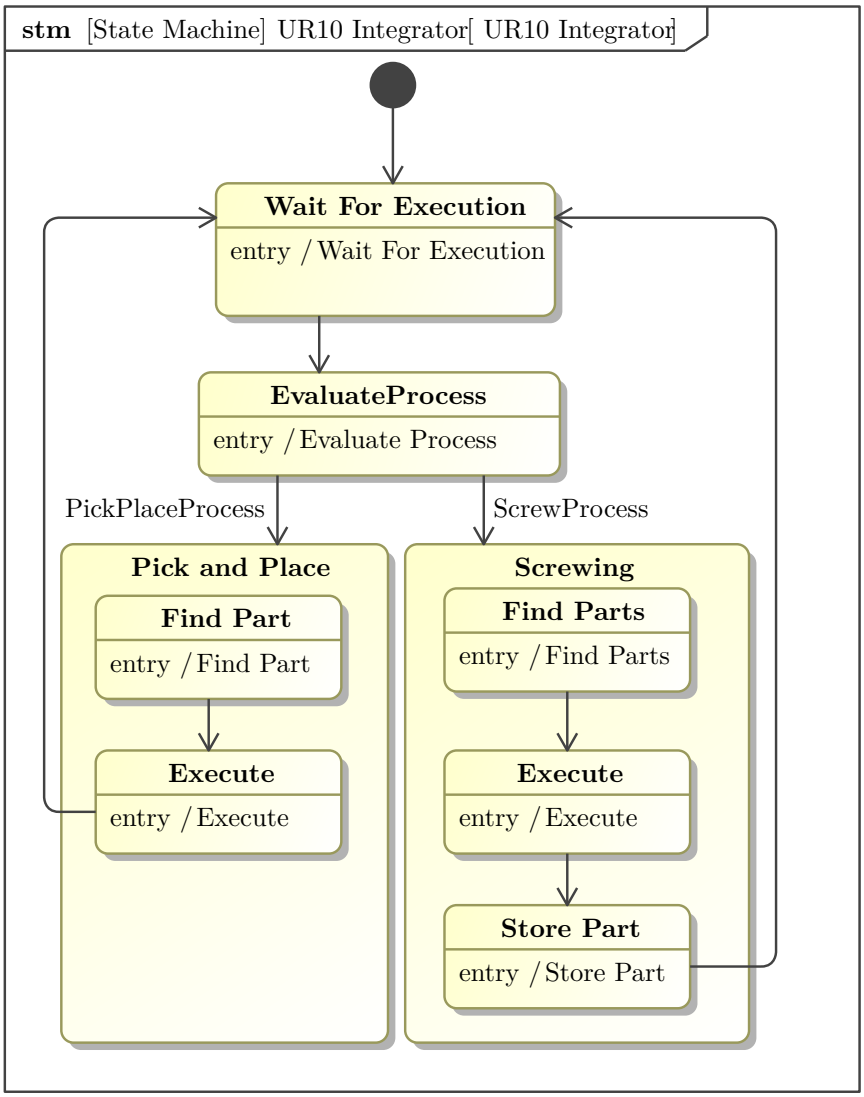


Abbildung 4.15: Zustandsdiagramm des südgerichteten Integrators der UR10e AAS

Nach dem Abschluss der Ausführung eines Prozesses wird ein internes Signal verschickt, um andere Bestandteile über den Abschluss der Ausführung zu informieren. Anschließend kehrt der Integrator in den Anfangszustand zurück und wartet auf den nächsten Prozess.

4.6 Untersuchung der Modelle

Sowohl das Gesamtmodell als auch die Teilmodelle wurden auf ihre Genauigkeit untersucht. Zu diesem Zweck wurden eine Reihe von Experimenten durchgeführt und die realen Ergebnisse mit Ergebnissen aus der Simulation verglichen. Zuerst wurden die Teilmodelle untersucht.

Die Gelenkbewegung des Roboters wurde ausführlich experimentell betrachtet, da die-

se einen Großteil der Bewegungen während der Produktion ausmacht. Es wurde daher untersucht, wie sich der echte Roboter sowie die Simulation verhalten, bei der Variation der Parameter, welche als Befehl an den Roboter übertragbar sind. Dies sind die Geschwindigkeit, die Beschleunigung sowie die Position. Zusätzlich wurde untersucht, wie sich unterschiedliche Massen des Endeffektors auf die Bewegungszeit auswirken.

Zur Bestimmung des Verhaltens wurde ein Startpunkt sowie ein Endpunkt festgelegt. Der Roboter bewegte sich anschließend vom Startpunkt zum Endpunkt und die Zeit für diese Bewegung wurde gemessen. Die Bewegung wurde jeweils 20 Mal wiederholt.

Zur Untersuchung des Verhaltens des Roboters bei variierender Geschwindigkeit wurde die Zeit für die Bewegung für Geschwindigkeiten zwischen $1\frac{rad}{s}$ und $4\frac{rad}{s}$ in Schritten von $0,25\frac{rad}{s}$ gemessen. Die Beschleunigung wurde auf $2\frac{rad}{s^2}$ festgelegt.

Das Verhalten des Roboters bei unterschiedlichen Beschleunigungen wurde anhand der selben Bewegung wie bei der Geschwindigkeitsuntersuchung untersucht. Es wurden Beschleunigungen zwischen $0,5\frac{rad}{s^2}$ und $3\frac{rad}{s^2}$ betrachtet in Schritten von $0,25\frac{rad}{s^2}$. Die Geschwindigkeit wurde auf $2\frac{rad}{s}$ gesetzt.

Der Einfluss der Masse auf die Genauigkeit der Simulation wurde untersucht durch die Durchführung der selben Bewegung wie bei der Geschwindigkeits- und Beschleunigungsuntersuchung. Die Bewegung wurde für drei Massen ausgeführt. Diese sind $0kg$, $1,2kg$ und $3kg$. $0kg$ entspricht dem Roboter ohne Endeffektor, $1,2kg$ entspricht dem Roboter mit Greifer als Endeffektor und $3kg$ entspricht dem Roboter mit dem Schraubenzieher als Endeffektor.

Als letzter Parameter wurde die Position variiert. Zu diesem Zweck wurden insgesamt 5 Positionen zufällig ausgewählt. Der Roboter wurde je Position 20-mal von einer Startposition zur jeweiligen Position bewegt, mit einer Geschwindigkeit von $2\frac{rad}{s}$ und einer Beschleunigung von $2\frac{rad}{s^2}$.

Die Linearachsenbewegung wurde für drei verschiedene Bewegungsdistanzen experimentell untersucht. Hierbei wurden Bewegungsdistanzen gewählt, welche während der Ausführung des Gesamtprozesses zurückgelegt werden. Diese sind $0mm$, $1000mm$ und $2629mm$.

Das Verhalten der Endeffektoren wurde ebenfalls experimentell untersucht. Zu diesem Zweck wurden Einzelprozesse verwendet, die Teil des Gesamtprozesses sind. Für den Greifer wurden fünf Einzelprozesse mit unterschiedlichen Greifpositionen gewählt. Dies bedeutet, dass die Distanz, welche die Finger des Greifers zurücklegen müssen, unterschiedlich ist. Zur Unterscheidung des Schraubenziehers wurden lediglich die Bewegungen Aufheben, Vorschrauben und Festziehen betrachtet. Die Bewegung des Schafts wurde nicht noch einmal gesondert betrachtet, da dessen reales Verhalten bereits in Abbildung 4.13 erfasst wurde.

Abschließend wurde der in Abschnitt 4.1 beschriebene Gesamtprozess untersucht. Dieser wurde zu diesem Zweck 20-mal ausgeführt und die Zeit für die individuellen Teilprozesse sowie den Gesamtprozess gemessen.

5 Ergebnisse der Implementierung

Zur Bewertung der erstellten Simulationsmodelle wurden eine Reihe von Messkampagnen durchgeführt. Das Ziel hierbei war es, das echte Verhalten mit dem simulierten Verhalten des Produktionssystems zu vergleichen, um festzustellen, ob die Simulation eine hinreichend genaue Abbildung der Realität ist.

Zu diesem Zweck wurden die individuellen Bewegungsformen (Gelenkbewegung, Linearachsenbewegung und Endeffektoren) sowie der in Abschnitt 4.1 beschriebene Gesamtprozess zur Produktion eines Teils des Crownmoduls betrachtet.

Die Abweichung der Simulationszeit von der realen Zeit ist hier entsprechend Formel 5.1 als die Differenz der realen Zeit und der Simulationszeit definiert.

$$Abweichung = RealeZeit - Simulationszeit \tag{5.1}$$

5.1 Gelenkbewegung

Die Gelenkbewegung wurde, wie in Abschnitt 4.6 beschrieben, jeweils für verschiedene Geschwindigkeiten, Beschleunigungen, Massen und Punkte betrachtet. Es wurde gemessen, wie sich eine Variation dieser Werte auf die Genauigkeit der Simulation auswirkt.

Die Ergebnisse der Geschwindigkeitsuntersuchung sind in Abbildung 5.1 abgebildet. Die Standardabweichung der jeweiligen Zeiten ist in diesem Fall so klein, dass sie im Diagramm nicht visuell erkennbar ist. Die Standardabweichung der gemessenen Zeit ist für alle Geschwindigkeiten in etwa $5ms$.

Abbildung 5.2 bildet die Abweichung der Simulationsdauer von der realen Dauer für die jeweiligen Geschwindigkeiten ab. Es ist zu erkennen, dass für kleine Geschwindigkeiten die Abweichung sehr klein ist. Sie liegt im Bereich von $-5ms$. Bei hohen Geschwindigkeiten steigt die Abweichung und erreicht ein Plateau bei etwa $115ms$. Die Geschwindigkeit scheint somit bei niedrigen Geschwindigkeiten wenig Einfluss auf die Genauigkeit der Simulation zu haben, während bei hohen Geschwindigkeiten die Genauigkeit sinkt und schließlich einen Maximalwert erreicht. Die Genauigkeit ist am höchsten bei Geschwindigkeiten kleiner gleich $2,25\frac{rad}{s}$. Die Abweichung scheint mit steigender Geschwindigkeit zuzunehmen. Das Plateau nach $3,25\frac{rad}{s}$ erklärt sich dadurch, dass bei diesem Wert die Maximalgeschwindigkeit der Gelenke erreicht ist. Die in Abbildung 5.2 zu sehenden Fehlerbalken ergeben sich aus den Standardabweichungen der Zeitmessung.

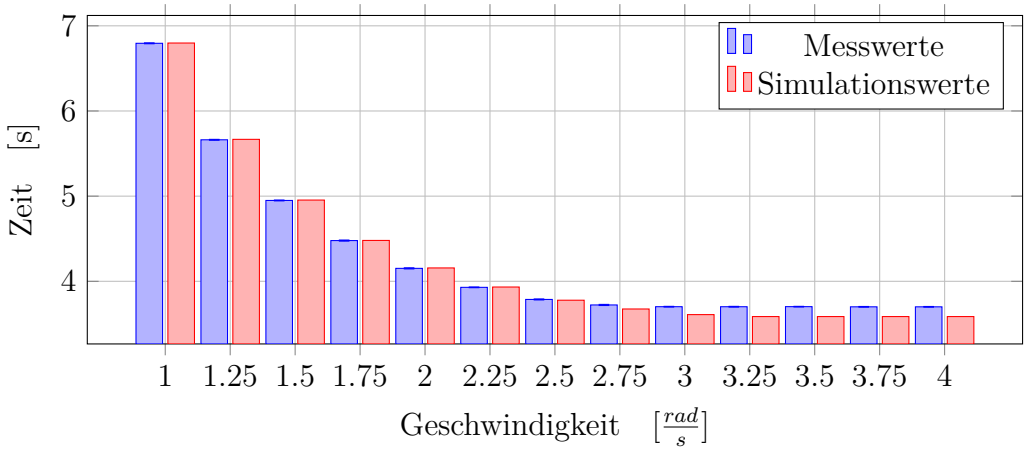


Abbildung 5.1: Zeit zur Bewegung bei variierender Geschwindigkeit

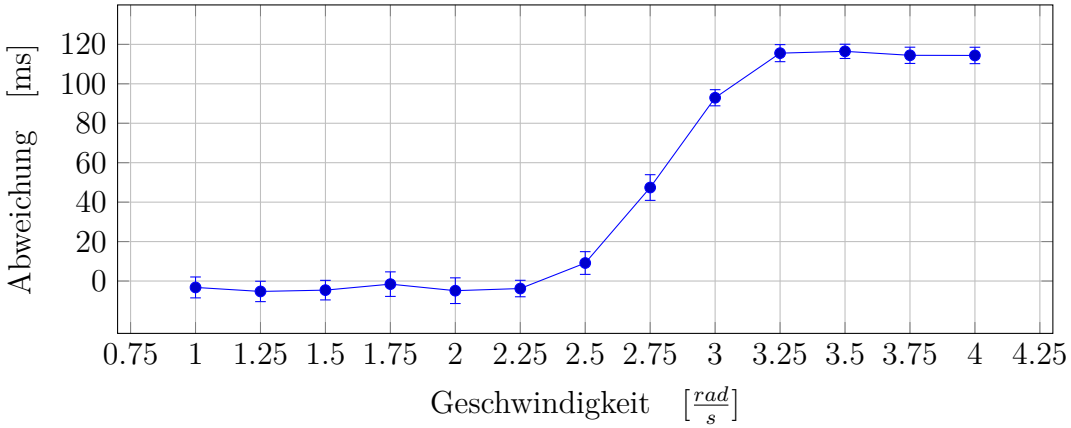


Abbildung 5.2: Abweichung zwischen Simulation und Messwerten bei variierender Geschwindigkeit

Die Ergebnisse der Beschleunigungsuntersuchung sind in Abbildung 5.3 zu sehen. Ähnlich wie bei der Geschwindigkeitsuntersuchung ist die Standardabweichung der Bewegung sehr klein. Sie liegt zwischen $5ms$ und $13ms$.

Entsprechend Formel 5.1 ist in Abbildung 5.4 die Abweichung für die jeweiligen Beschleunigungen dargestellt. Es zeigt sich hier, dass die Abweichung mit Ausnahme einer Anomalie bei $2,75\frac{rad}{s^2}$ sehr konstant ist. Es zeigt sich somit, dass jenseits der Anomalie die Beschleunigung keinen Einfluss auf die Genauigkeit der Simulation hat. Die Genauigkeit der Simulation ist für alle Beschleunigungen außer $2,75\frac{rad}{s^2}$ gleich bei etwa $-5ms$.

Die Ergebnisse der Massenuntersuchung sind in Abbildung 5.5 zu sehen. Es wurde der Roboter ohne Endeffektor, mit Greifer und mit Schraubenzieher betrachtet. Es zeigt sich, dass die Masse keinen erkennbaren Einfluss auf die Bewegungsdauer hat. Diese ist für alle Massen gleich bei $4,15s$.

Die Abweichung der Simulation von der Realität ist in Abbildung 5.6 dargestellt. Auch für die Abweichung zeigt sich kein erkennbarer Einfluss auf die Genauigkeit. Die Ab-

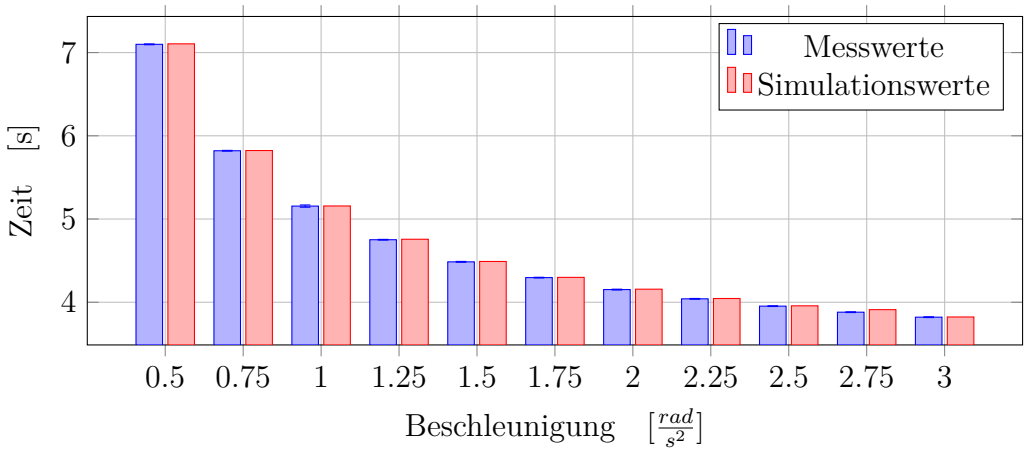


Abbildung 5.3: Zeit zur Bewegung bei variierender Beschleunigung

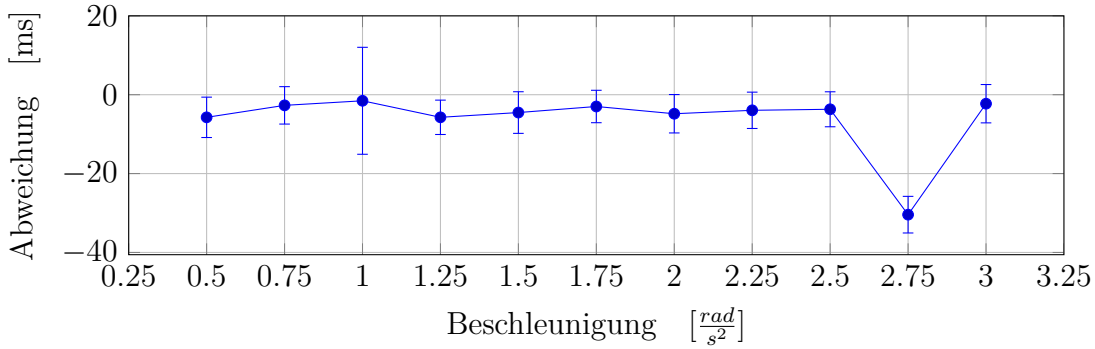


Abbildung 5.4: Abweichung zwischen Simulation und Messwerten bei variierender Beschleunigung

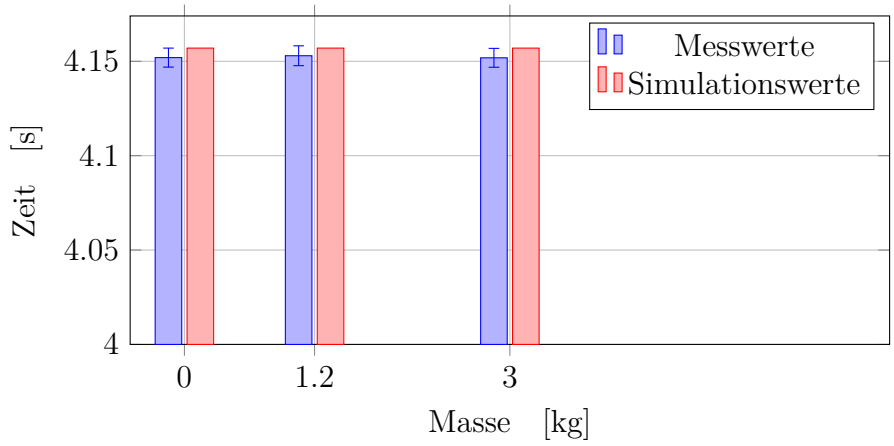


Abbildung 5.5: Zeit zur Bewegung bei variierender Masse

weichung beträgt für alle Massen $-5ms$ mit einer Standardabweichung von $5ms$.

Die Ergebnisse der Bewegung zu verschiedenen Punkten sind in Abbildung 5.7 dargestellt. Auch hier ist das Verhalten je Position sehr konstant mit Standardabweichungen von etwa $5ms$. Die jeweilige Dauer ist abhängig, von den Entfernungen welche die

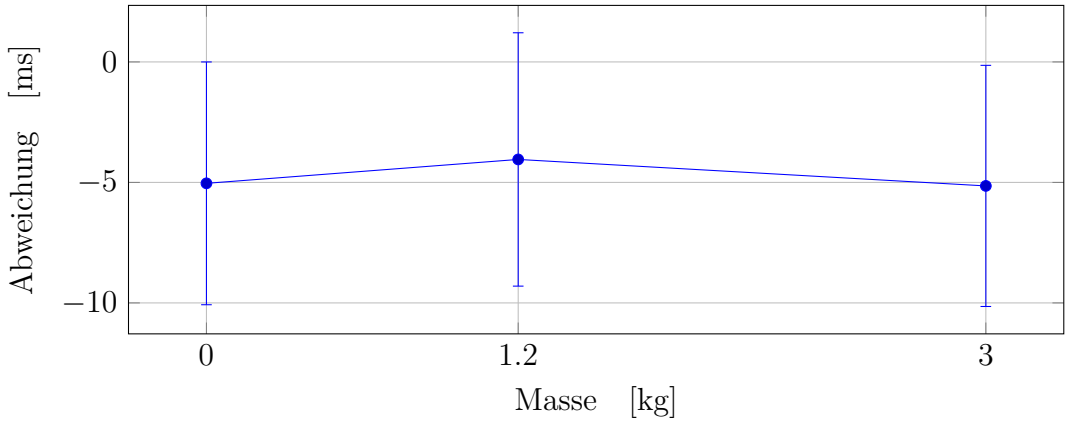


Abbildung 5.6: Abweichung zwischen Simulation und Messwerten bei variierender Masse

jeweiligen Gelenke zurücklegen müssen und ist somit von Position zu Position sehr unterschiedlich.

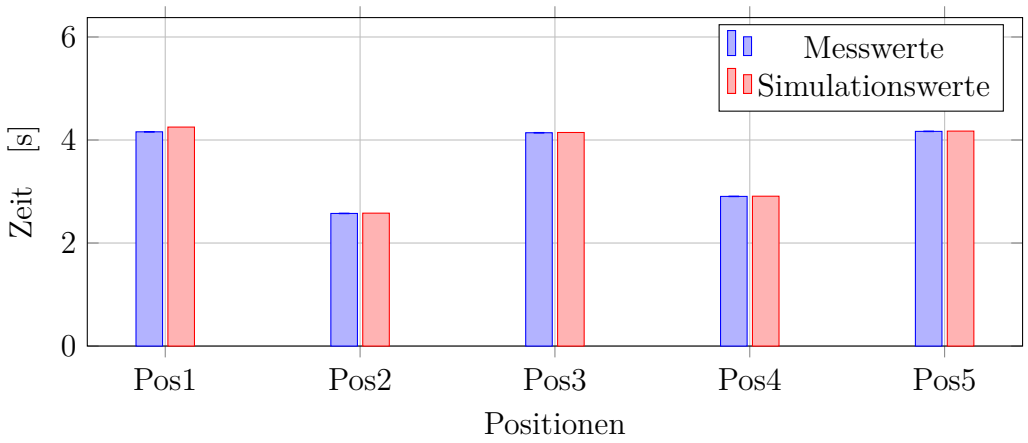


Abbildung 5.7: Zeit zur Bewegung bei variierender Zielposition

Die Genauigkeit der Simulation bei variierender Position ist in Abbildung 5.8 dargestellt. Hier ist zu erkennen, dass für Position 2-5 die Abweichung ähnlich wie bei der Beschleunigungs und Massenuntersuchung $5ms$ beträgt. Position 1 jedoch weicht stark davon ab mit einer Abweichung von $-90ms$. Position 1 unterscheidet sich von den anderen Positionen dadurch, dass bei ihr die Distanz zwischen dem Schwerpunkt des Endeffektors und der Basis des Roboters besonders groß ist. Bei den restlichen Positionen ist diese Distanz wesentlich geringer. Diese Distanz scheint somit einen Einfluss auf die Genauigkeit der Simulation zu haben, wobei große Distanzen eine geringere Genauigkeit aufweisen.

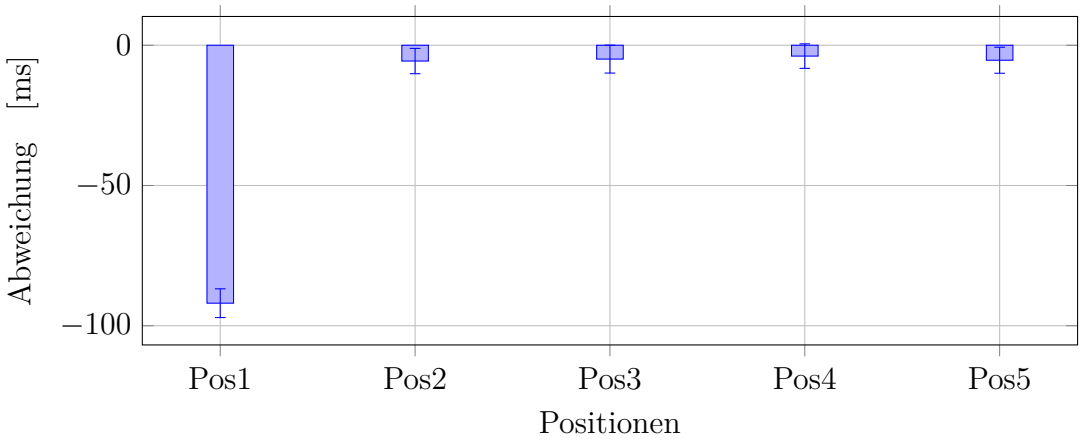


Abbildung 5.8: Abweichung zwischen Simulation und Messwerten bei variierender Zielposition

5.2 Linearachsenbewegung

Wie in Abschnitt 4.6 beschrieben wurde die Bewegung der Linearachse für die Distanzen $0mm$, $1000mm$ und $2629mm$ untersucht.

Auch das Verhalten der Linearachse ist sehr konstant mit einer Standardabweichung von $5ms$ bis $40ms$. Wobei die Standardabweichung bei der Entfernung von $0mm$ am niedrigsten ist und bei $1000mm$ am höchsten. Die Dauer der Bewegung der Linearachse bei $0mm$ ergibt sich wie in Abschnitt 4.3.2 beschrieben, aus den Signal- und Rechenzeiten des Roboters sowie der SPS. Das Verhalten zwischen Zeit und Distanz ist linear, da sich nach einer Beschleunigungsphase die Linearachse die Distanz mit einer konstanten Geschwindigkeit zurücklegt. Die ermittelten Messwerte sind in Abbildung 5.9 abgebildet.

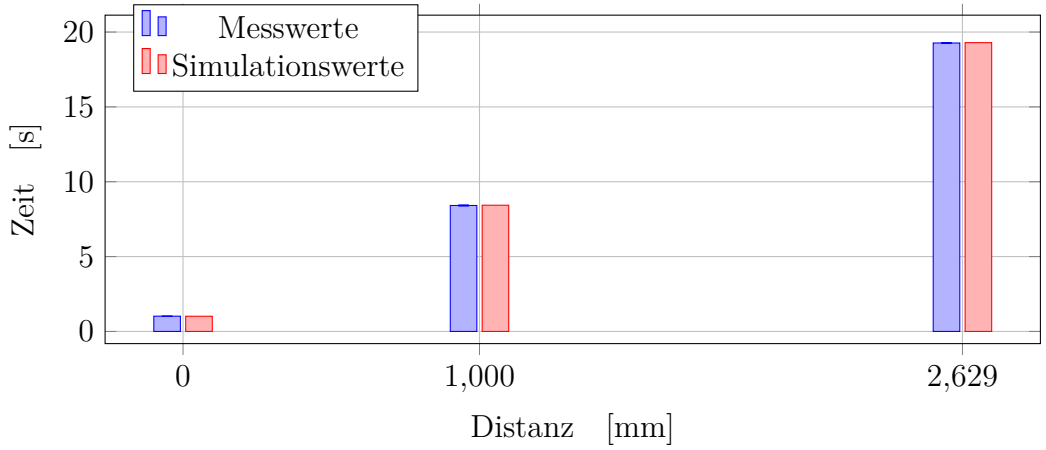


Abbildung 5.9: Zeit zur Bewegung der Linearachse für verschiedene Entfernungen

Die Abweichung der Simulation von der Realität für die Linearachsenbewegung ist in Abbildung 5.10 abgebildet. Es ist zu erkennen, dass bei der Entfernung von $0mm$ die

Abweichung $5ms$ beträgt. Die Simulation ist somit schneller als die Realität. Bei tatsächlicher Bewegung ist die Simulation jedoch langsamer als die Realität. Bei $1000mm$ entsteht eine Abweichung von $-17ms$ und bei $2629mm$ eine Abweichung von $-24ms$. Die Abweichung steigt somit mit der zurückzulegenden Entfernung. In Anbetracht der Dauer der Bewegungen von $8s$ und $19s$ ist die Abweichung prozentual an der Bewegungsdauer jedoch sehr gering.

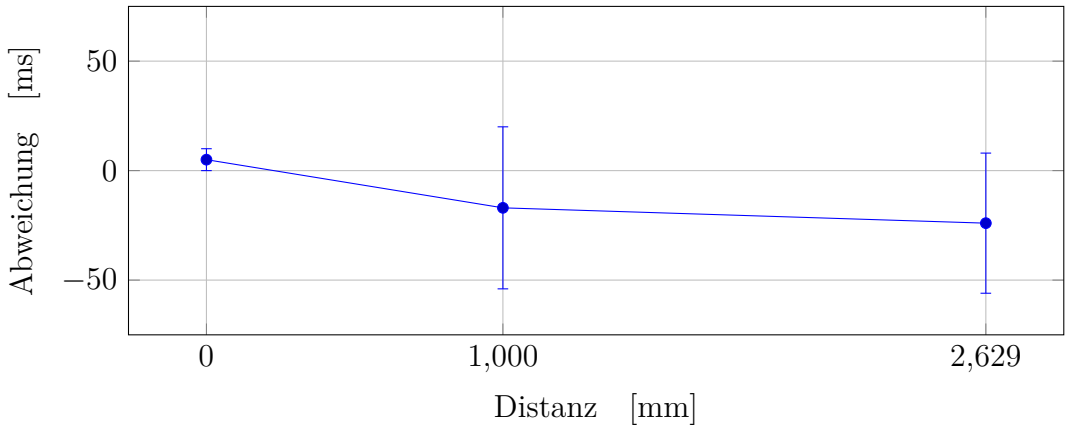


Abbildung 5.10: Abweichung der Dauer der Bewegung der Linearachse zwischen Realität und Simulation

5.3 Endeffektoren

Wie in Abschnitt 4.6 beschrieben, wurde ebenfalls das Verhalten der Endeffektoren experimentell untersucht.

Die Ergebnisse der Greiferbewegung sind in Abbildung 5.11 dargestellt. Auch hier ist erkennbar, dass die Standardabweichung der Bewegungsdauer sehr klein ist. Sie liegt zwischen $14ms$ und $19ms$. Die Abweichung der Simulation von den Messergebnissen bei

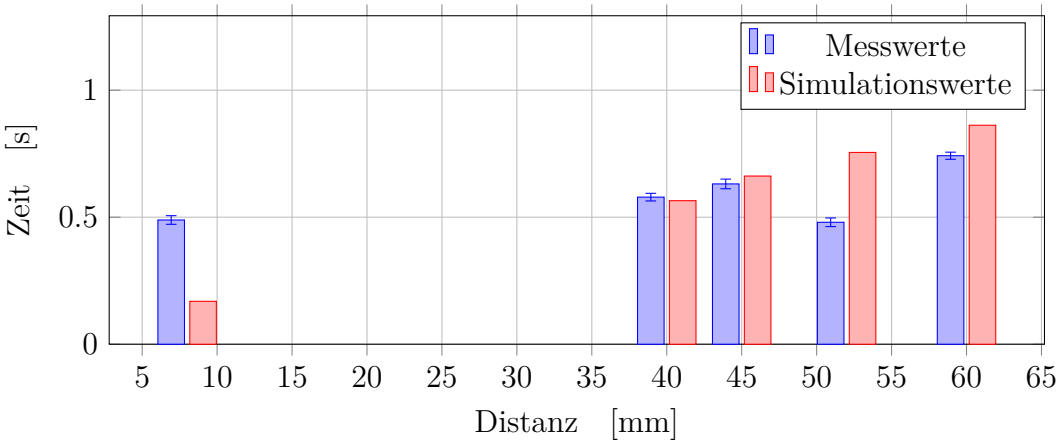


Abbildung 5.11: Dauer der Bewegungs des Greifers für verschiedene Öffnungsweiten

der Greiferbewegung ist in Abbildung 5.12 dargestellt. Hier ist zu erkennen, dass bei einer kleinen Bewegung von $8mm$ die Simulation schneller ist als die Messergebnisse. Bei größeren Entfernungen jedoch ist die Simulation langsamer als die Messergebnisse.

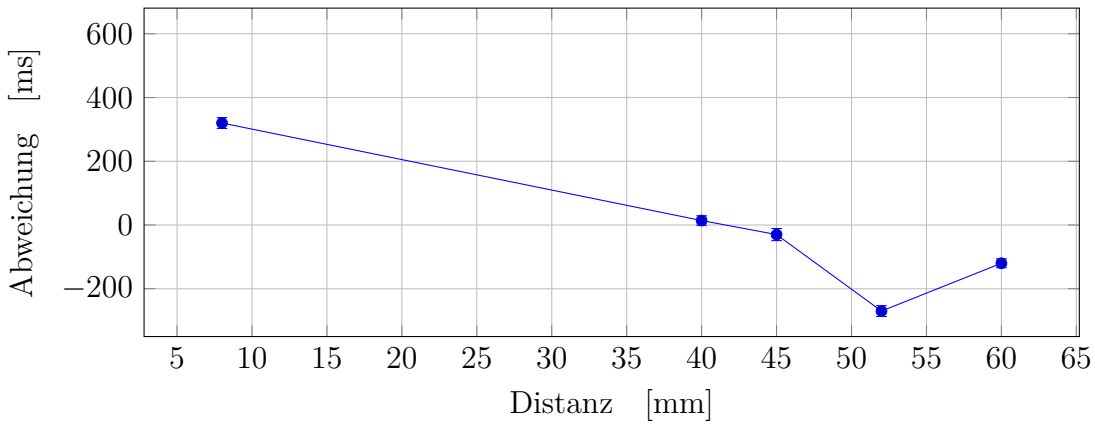


Abbildung 5.12: Abweichung der Bewegung des Greifers zwischen Realität und Simulation

Die erfassten Messwerte sowie die Simulationsdaten der Schraubenzieherbewegung sind in Abbildung 5.13 dargestellt. Es ist zu erkennen, dass die Standardabweichung für die Prozesse Aufheben und Vorschrauben sehr hoch ist. Die Standardabweichung beträgt für diese Prozesse $346ms$ und $445ms$. Die tatsächliche Ausführungsdauer dieser Prozesse ist somit sehr variabel. Dies liegt daran, dass die tatsächliche Dauer dieser Prozesse sehr von äußeren Einflüssen wie der genauen Positionierung der Schrauben und der Bewegung der Bauteile abhängig ist.

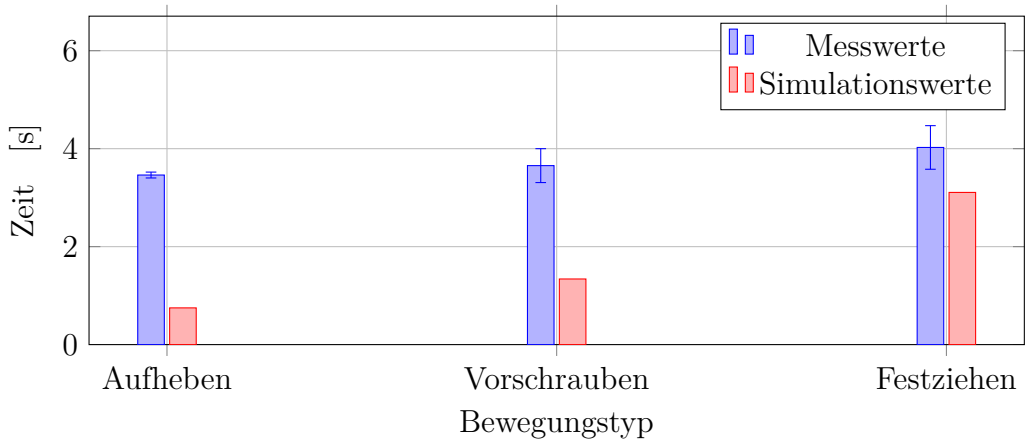


Abbildung 5.13: Dauer der Bewegung des Schraubenzieher je Bewegungstyp

Die Abweichung der Simulation des Schraubenziehers von der Realität ist in Abbildung 5.14 dargestellt. Es lässt sich erkennen, dass die Abweichung für alle Prozesse sehr hoch ist. Der Prozess Aufheben weist mit $2712ms$ die höchste Abweichung auf. Der Prozess Festziehen wiederum weist mit $918ms$ die niedrigste Abweichung auf. Der

Schraubenzieher besitzt somit unter allen simulierten Prozessen die größte Abweichung zwischen Simulation und Realität.

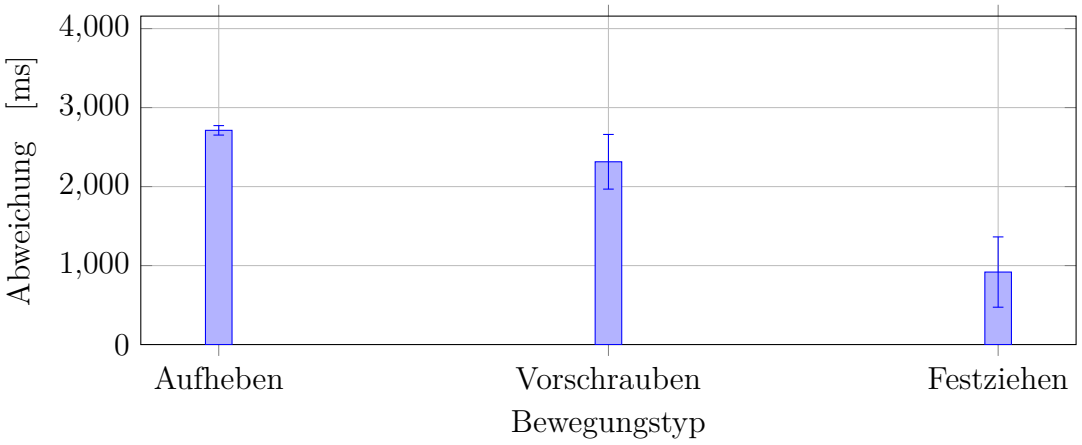


Abbildung 5.14: Abweichung der Dauer der Schraubenzieherbewegung zwischen Simulation und Realität

5.4 Gesamtprozess

Der Gesamtprozess besteht, wie in Abschnitt 4.1 beschrieben, aus dem Zusammenbau eines Teils des Crownmoduls. Zu diesem Zweck verhandeln die einzelnen AAS mithilfe des Ausschreibungsverfahrens miteinander, um den Produktionsprozess abzuschließen. Die Untersuchung des Gesamtprozesses wurde, wie in Abschnitt 4.6 beschrieben, durchgeführt. Über den gesamten Produktionsprozess ergibt sich bei einer realen Gesamtdauer von $293,35s$ eine Abweichung von $36,92s$. Dies entspricht einer Genauigkeit der Simulation von $87,41\%$.

Die Zeiten zur Ausführung der individuellen Produktionsschritte sind in Abbildung 5.15 dargestellt. Diese Zeiten umfassen, sowohl die Durchführung des Ausschreibungsverfahrens als auch die Durchführung der Bewegungen der Ressourcen.

Abbildung 5.16 stellt die Abweichung zwischen Simulation und Realität für die einzelnen Produktionsschritte dar. Es ist zu erkennen, dass die Abweichung für die Schritte VFork, XRodA, XRodB und HFork+DEU in etwa konstant bei $5s$ liegt. Für den Prozess Schrauben ergibt sich eine eine höhere Abweichung von $18s$.

Durch das Herausrechnen der Zeiten, welche zur Ausführung der Bewegung benötigt werden, lässt sich die Dauer, welche allein für digitale Aufgaben verwendet wird, isolieren. Die digitalen Aufgaben sind Signal- und Rechenzeiten. Diese sind in Abbildung 5.17 dargestellt. Für die 4 Schritte, VFork, XRodA, XRodB und HFork+DEU ergibt sich eine konstante Zeit von etwa $14s$. Diese umfasst die originale Ausschreibung für die Ausführung des Produktionsschrittes sowie die Ausschreibung zur Bereitstellung des Einzelteils. Der Schritt Schrauben benötigt in der Realität eine Dauer von $18s$. Dies umfasst die Ausschreibung zur Ausführung des Produktionsschrittes, die Bereitstellung

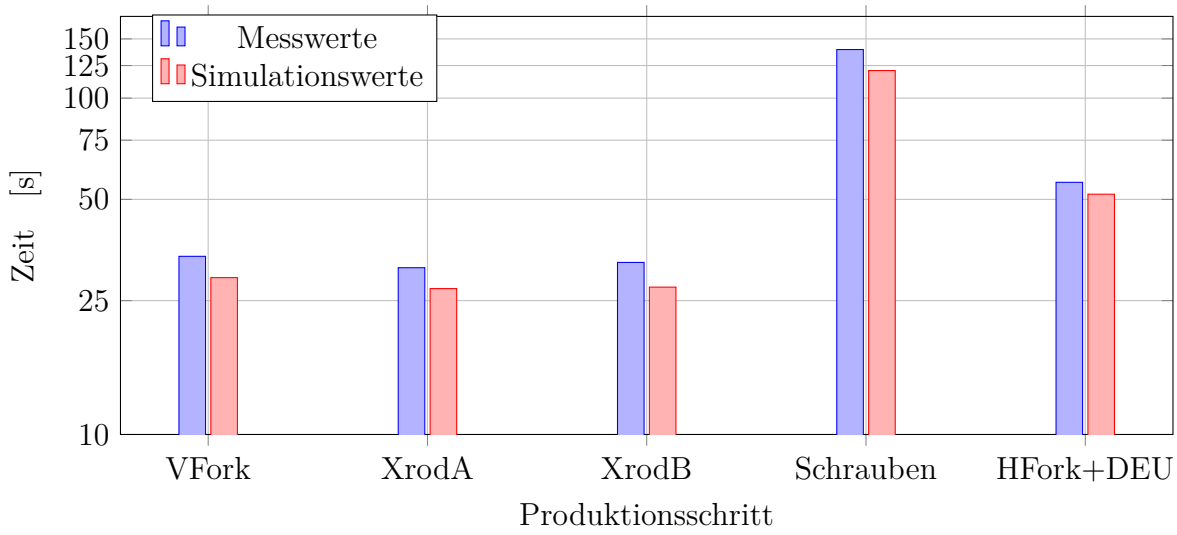


Abbildung 5.15: Dauer der Ausführung der jeweiligen Produktionsschritte

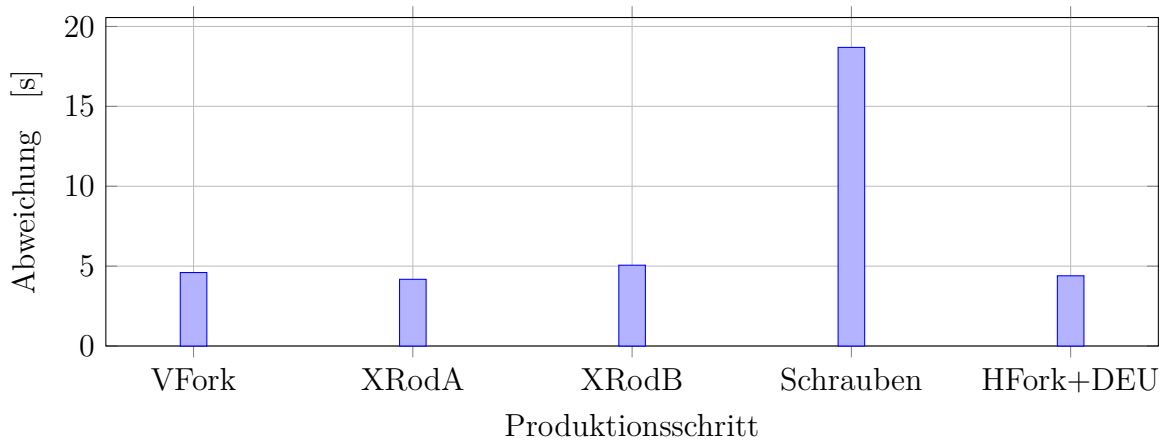


Abbildung 5.16: Abweichung der Dauer der Produktionsschritte zwischen Realität und Simulation

der Einzelteile sowie die Ausschreibung zur Einlagerung des zusammengeschraubten Bauteils.

Die Abweichung zwischen den simulierten Signal- und Rechenzeiten und den gemessenen Werten ist in Abbildung 5.18 dargestellt. Hier ist zu erkennen, dass die Abweichung für die Schritte VFork, XRodA, XRodB und HFork+DEU etwa $4s$ beträgt. Für den Schritt Schrauben ist die Abweichung geringer und beträgt $2,7s$.

Abbildung 5.19 zeigt die Abweichungen aufgrund der Signal- und Rechenzeiten im Vergleich mit den Abweichungen aufgrund der Bewegungszeiten. Es ist zu erkennen, dass während den Prozessen VFork, XRodA, XRodB und HFork+DEU die Abweichungen aufgrund der Signal- und Rechenzeiten größer sind als die der Bewegungsszeiten. Für Schrauben ist das Gegenteil der Fall. Dies liegt daran, dass die Abweichungen der Schraubenzieherbewegung während des Prozess Schrauben sehr groß sind. Die Abweichungen der Bewegungen für die restlichen Bewegungen sind im Verlgeich sehr gering.

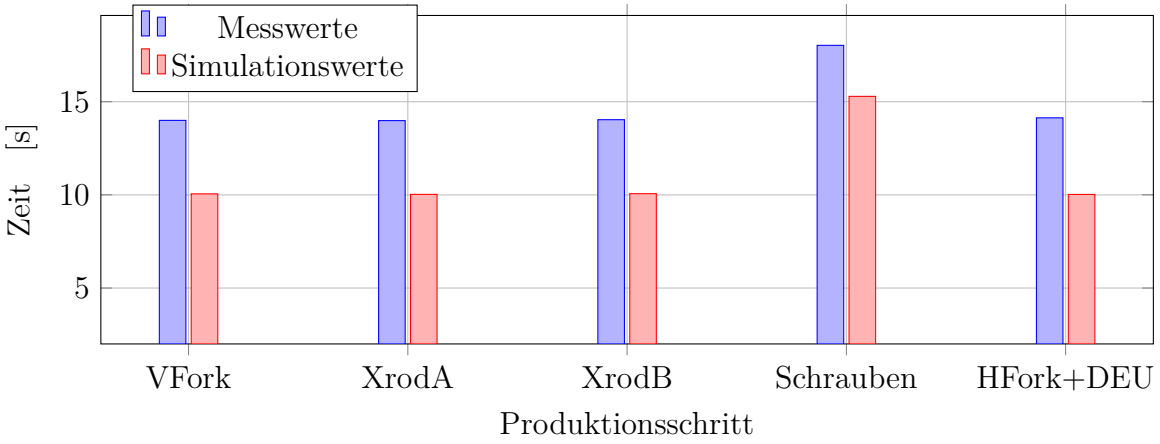


Abbildung 5.17: Dauer der Signal- und Rechenzeiten je Produktionsschritt

Abbildung 5.18: Abweichung der Signal- und Rechenzeit zwischen Realität und Simulation

Die Abweichung zwischen Simulation und Realität für die Bewegung je Position ist in Abbildung 5.20 dargestellt. Es ist zu erkennen, dass die Abweichung für jene Prozesse, welche den Einsatz des Schraubenziehers umfassen, besonders hoch ist. Ebenfalls hervorzuheben ist die Abweichung bei P1. Dies ist auf eine starke Abweichung bei der Bewegung der Gelenke zu einer Position zurückzuführen.

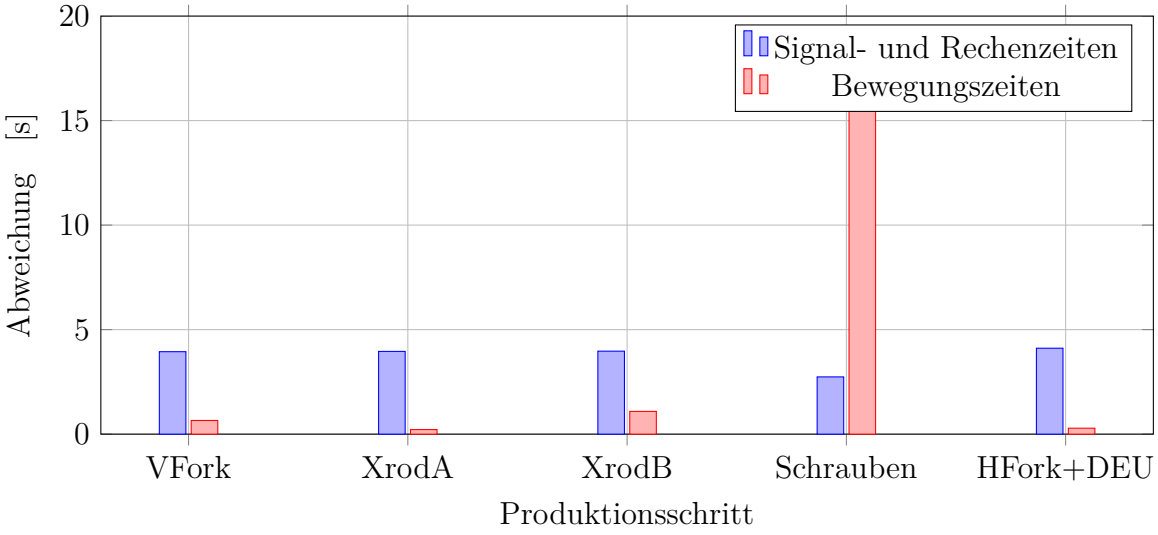


Abbildung 5.19: Abweichung der Signal- und Rechenzeit und Bewegungszeiten zwischen Realität und Simulation

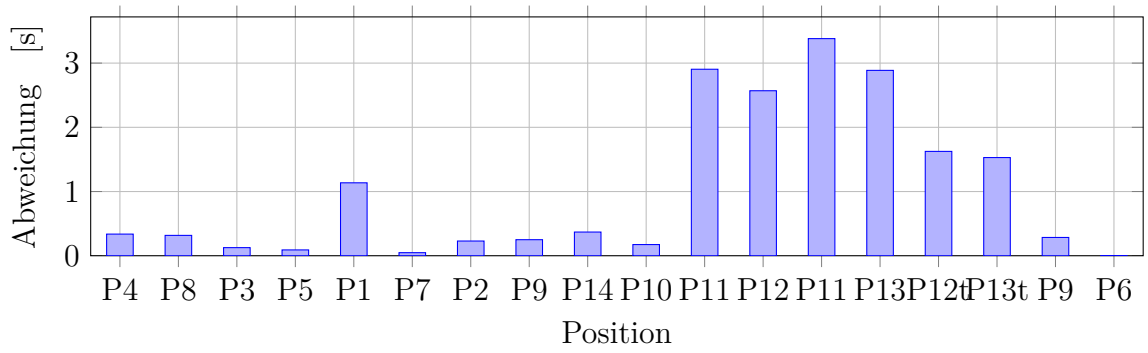


Abbildung 5.20: Abweichung der Bewegungszeiten je Position (nach Tabelle 4.1) zwischen Realität und Simulation

6 Validierung und Diskussion

Im Rahmen der Arbeit wurde ein Produktionssystem modelliert zur Simulation von Produktionsprozessen. Die Genauigkeit dieser Modellierung wurde bestimmt durch den Vergleich mit einem realen Produktionssystem. Wie in Abschnitt 5.4 dargestellt, ergab sich für den Gesamtprozess eine relative Genauigkeit der Simulation von 87,41% mit einer absoluten Abweichung von $36,92s$.

Welche Genauigkeit für eine Simulation ausreicht, ist abhängig vom spezifischen Anwendungsfall dieser Simulation[51]. Zur Bestimmung, welche absolute Abweichung innerhalb der Simulation hinreichend ist, ist es sinnvoll, die jeweiligen Modelle in die in Abschnitt 2.2 vorgestellte ISA 95 Pyramide einzuordnen. Die in dieser Arbeit verwendeten Modelle integrieren dabei mehrere Ebenen der ISA 95 Pyramide. Die Modelle der Assets lassen sich klar der Ebene der intelligenten Geräte sowie der physikalischen Prozesse zuordnen. Die Assets manipulieren und erfassen reale physikalische Prozesse. Ausgehend von dieser Einordnung und den unterschiedlichen, in Abschnitt 2.2 vorgestellten Zeithorizonten der jeweiligen Ebenen, lässt sich feststellen, dass die Modelle der Assets in einem Bereich operieren, in dem Zeithorizonte von Sekunden bis Millisekunden relevant sind. Die Modelle der Verwaltungsschalen wiederum integrieren die Steuerungssysteme und die Fertigungsbetriebssysteme. Die AAS steuern und kontrollieren ihre jeweiligen Assets. Gleichzeitig ist die AAS zuständig für die Zuweisung der einzelnen Prozesse zu Ressourcen. Sie übernimmt gemeinsam mit anderen AAS durch das Ausschreibungsverfahren die Sequenzierung der einzelnen Prozessschritte. Sie umfasst daher sowohl die Fertigungsbetriebssysteme als auch die Steuerungssysteme. Ihre Entscheidungen finden somit in einem Zeithorizont von mehreren Minuten statt. Die Genauigkeit der jeweiligen Modelle wird weiterführend entsprechend dieser unterschiedlichen Zeithorizonte bewertet.

6.1 Diskussion der Modelle der Ressourcen

Die Modelle der Ressourcen befinden sich in einem Bereich, in dem bereits Abweichungen von Millisekunden relevant sein können. Da die kleinste in der Simulation abgebildete Zeitgröße eine Millisekunde beträgt, lassen sich Abweichungen im niedrigen zweistelligen Millisekundenbereich als hinreichend genau betrachten. Wie in Kapitel 5 dargestellt, befindet sich die absolute Abweichung der individuellen Bewegungsprozesse, mit Ausnahme der Bewegungen des Schraubenziehers, im Millisekundenbereich. Der Fehler der Schraubenzieherbewegungen ist jedoch größer als eine Sekunde.

Die Gelenkachsenbewegung macht einen Großteil der Zeit der Gesamtbewegung aus. Es weist in den meisten Fällen eine absolute Abweichung von $-5ms$ auf. Wie in Abschnitt 5.1 gezeigt, kann es bei spezifischen Positionen zu sehr großen Ungenauigkeiten kommen, von bis zu $-90ms$. Eine Gemeinsamkeit dieser Positionen ist, dass bei ihnen der Schwerpunkt des Endeffektors sich besonders weit von der Basis des Roboters entfernt. In Folge der gestiegenen Entfernung von Endeffektormasse zur Roboterbasis wirken bei diesen Positionen höhere Momente auf die einzelnen Gelenke ein. Diese scheinen durch den Regler des Roboters nicht hinreichend kompensiert zu werden, was zu einer Abweichung zwischen der simulierten Bewegungsdauer und der realen Bewegungsdauer für diese Positionen führt. Diese Positionen traten jedoch während des Gesamtprozesses relativ selten auf, da durch die Bewegung der Linearachse es nicht notwendig war, den Roboter in solche Positionen zu bewegen. Dieser Fehler wirkt sich somit nicht sehr stark auf die Genauigkeit der Simulation der Gelenkachsen während des Gesamtprozesses aus.

Für die Gelenkbewegung war ebenfalls zu erkennen, dass es für einen Großteil der Bewegungen einen konstanten Fehler von etwa $-5ms$ gab. Eine mögliche Ursache dieses Fehlers ist, dass Signal- und Rechenzeiten in der Modellierung des Roboters überschätzt wurden. Darauf weist hin, dass die Abweichung von $-5ms$ konstant für fast alle Gelenkachsenbewegungen ist. Die Abweichung befindet sich im einstelligen Millisekundenbereich. Da, wie zuvor beschrieben, der Roboter im Sekunden- bis Millisekundenbereich operiert, ist die Abweichung im einstelligen Millisekundenbereich hinreichend gering, um das Modell als hinreichend genau zu bewerten.

Die Abweichung der Linearachse beträgt $5ms$ bis $-24ms$. Wie bei der Gelenkachsenbewegung sind diese Abweichungen im Vergleich zum relevanten Zeithorizont sehr gering. Die Genauigkeit der Linearachse kann somit als hinreichend bewertet werden. Wie in Abschnitt 5.2 dargestellt, sinkt mit zurückzulegender Entfernung die Genauigkeit der Simulation. Ein möglicher Grund dafür ist, dass bei der Bewegung der Linearachse sich sowohl der Roboter als auch dessen Kabelführungen bewegen. Im Besonderen ist die Bewegung der Kabelführungen von großen Unsicherheiten behaftet. Die Kabelführungen können sich unerwartet verkanten oder aus verkanteten Situationen lösen. Die Kabelführungen bringen dabei nicht genügend Kraft auf, um zu einer Unterbrechung der Bewegung zu führen. Sie können jedoch ruckartige Bewegungen herbeiführen oder die Bewegung des Roboters verlangsamen. Weitere Untersuchungen sind notwendig, um die genaue Quelle des Fehlers zu bestimmen. Aufgrund der Länge der Linearachsenbewegungen sind die Abweichungen der Simulation von der Realität im Verhältnis zur Gesamtdauer sehr gering, wodurch die Bewegung weiterhin eine hohe relative Genauigkeit aufweist.

Die Genauigkeit des Greifers bezogen auf die Einzelprozesse schwankt sehr stark. In manchen Prozessen bewegt sich der Greifer in der Simulation zu langsam, während er sich in anderen wiederum zu schnell bewegt. Abbildung 6.1 zeigt die summierten Beträge der Abweichungen sowie den Betrag der summierten Abweichung je Produktionsschritt. Es ist zu erkennen, dass für die Schritte XRodA und XRodB die Summe der Beträge der Abweichungen und der Betrag der Summe der Abweichungen stark voneinander abweichen. Dies bedeutet, dass für XRodA und XRodB eine hohe Ungenauigkeit

in sowohl die positive als auch negative Richtung existiert. Die Ungenauigkeiten in positiver und negativer Richtung heben sich jedoch auf und tauchen somit nicht in der Betrachtung des Gesamtprozesses auf. Ein möglicher Grund für diese Ungenauigkeit ist eine nicht hinreichende Erfassung der Geometrie der XRod sowie eine unzuverlässige Positionsbestimmung des Greifers. Eine fehlerhafte Darstellung der Breite der Einzelteile könnte die Abweichung erklären, da in diesem Fall die Abweichung des Greifens und Schließens sich gegenseitig aufheben. Eine weitere Erklärung für die Abweichung ist der Umstand, dass die Schritte XRodA und XRodB den Einsatz sehr kleiner Greiferbewegungen umfassen, während die restlichen Prozesse größere Greiferbewegungen einsetzen. Dies ist ein Indiz darauf, dass das Verhalten des Greifers zwischen großen und kleinen Bewegungen sich signifikant voneinander unterscheidet.

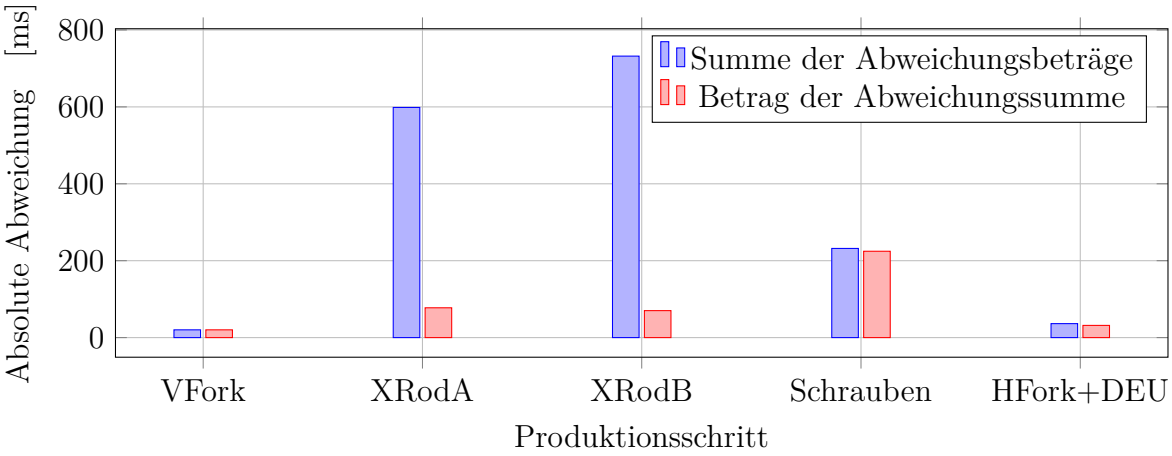


Abbildung 6.1: Absolute Abweichung der Greiferbewegung nach Produktionschritt

Die Abweichungen der Bewegung des Greifers schwanken zwischen $320ms$ und $-270ms$. Mit einer Abweichung im dreistelligen Millisekundenbereich verfügt das Modell des Greifers somit nicht über eine hinreichende Genauigkeit.

Der Schraubenzieher hat die höchste Ungenauigkeit aller Bewegungsformen mit Abweichungen zwischen $918ms$ und $2712ms$. Diese Abweichungen liegen im Sekundenbereich. Das Modell des Schraubenziehers ist daher nicht hinreichend genau. Die Ursachen dieser Ungenauigkeit sind: die zu vereinfachte Abbildung des Schraubenzieherverhaltens sowie eine nicht hinreichende physikalische Modellierung. Die Bewegung des Schafts ist innerhalb des Modells sehr gut abgebildet. Dies ist darauf zurückzuführen, dass hier die technischen Daten des Schraubenziehermodells so ermittelt wurden, dass diese der Bewegung des Schaftes entsprechen. Für die anderen Bewegungen wurde jedoch auf die Daten aus der technischen Beschreibung des Schraubenziehers zurückgegriffen, welche eine nicht hinreichende Beschreibung des Schraubenziehers lieferten. Wie in Abschnitt 5.3 beschrieben, ergab sich für den Prozess Festziehen die geringste Abweichung der drei restlichen Bewegungsformen. Diese Abweichung beträgt jedoch immer noch eine Sekunde. Aufgrund der äußeren Kräfte, die auf den Schraubenzieher während seiner Bewegung wirken, war die Modellierung hier nicht hinreichend. Das Verhalten des Schraubenziehers ist stark beeinflusst durch die Bewegung und Verformung der zusammenzuschraubenden Einzelteile. Dies wurde im Rahmen dieser Arbeit nicht modelliert.

Die exakte Position der Einzelteile und der Schrauben spielt im Unterschied zum Greifer eine wichtige Rolle. Für den Prozess Aufheben ist das Verhalten primär beeinflusst durch sowohl die Geometrie der Schraubenzufuhr als auch die exakte Position der aufzunehmenden Schraube. Während die Geometrie der Schraubenzufuhr annähernd mitbetrachtet wurde, wurde die exakte Position der Schraube vernachlässigt. Die reale Ausführung der Prozesse hat jedoch gezeigt, dass bereits geringfügige Positionsabweichungen der Schraube innerhalb der Schraubenzufuhr dazu führen konnten, dass die Schraube nicht ordentlich aufgenommen wurde. Dies beeinflusste nachfolgend ebenfalls die Prozesse des Einschraubens und Festziehens durch die ungenaue Positionierung der Schraube auf dem Bit. Dieses Verhalten wurde in dieser Arbeit nicht mitbetrachtet, da es eine zu detaillierte Modellierung der Vormontagezelle sowie der Schraubenzufuhr erfordert hätte. Im Besonderen ist die Positionierung der Schrauben in der Schraubenzufuhr sehr komplex und undeterministisch. Diese vereinfachte Betrachtung der Schraubenzufuhr führt jedoch zu hohen Ungenauigkeiten.

Der Prozess des Vorschraubens sieht sich von außen, primär der Einwirkung durch die zusammenzuschraubenden Bauteile ausgesetzt. Diese bringen Kräfte sowohl auf die Schraube als auch auf den Schraubenzieher ein und können zu Abweichungen in der Dauer des Prozesses führen.

Das Festziehen ist den Schraubenzieherprozessen am meisten äußeren Einwirkungen ausgesetzt. Diese Einwirkungen entstehen durch die Bewegung der Bauteile beim Festziehen der Schraube. Wie in Abschnitt 4.1 beschrieben, kommt es beim Festziehen der Schraubverbindung zwischen HFork und DEU zu einer elastischen Verformung der DEU. Dies führt dazu, dass sich die DEU im Laufe des Prozesses stark bewegt. Diese Bewegung sowie die Verformung der DEU wurden in dieser Arbeit nicht mitbetrachtet. Die zusätzlichen Kräfte, welche dadurch auf den Schraubenzieher wirkten, führten zu einer Varianz in der Umdrehungszahl des Schraubenziehers. Diese wurde nicht mitbetrachtet, da dies eine detaillierte physikalische Modellierung der Einzelteile erfordert hätte.

Die Simulation der Pneumatik weist eine Abweichung zwischen $7ms$ und $11ms$ auf. Diese Werte liegen im niedrigen zweistelligen Millisekundenbereich. Das Modell der Pneumatik verfügt somit über eine hinreichende Genauigkeit. Die Pneumatik besitzt von allen Ressourcen die höchste Genauigkeit. Dies ergibt sich daraus, dass sowohl in der Realität als auch in der Simulation während der Bewegung lediglich $2s$ gewartet wird. Die hohe Genauigkeit ergibt sich somit aus dem sehr einfachen Verhalten sowohl im Modell als auch in der Realität. Die verbliebene Abweichung lässt sich durch Signal- und Rechenzeiten erklären, welche hier nicht abgebildet wurden.

6.2 Diskussion der Modelle der Verwaltungsschale

Jenseits der für den Anwendungsfall spezifischen physikalischen Bewegungen wurde ebenfalls das digitale Verhalten der AAS abgebildet. Wie in Abschnitt 5.4 beschrieben, liegt die Abweichung zwischen den simulierten Signal- und Rechenzeiten und den realen

Signal- und Rechenzeiten für die Schritte VFork, XRodA, XRodB und HFork+DEU bei $4s$ und für den Schritt Schrauben bei $3s$.

Die AAS ist in der Ebene der Fertigungsbetriebssysteme und der Steuerungsebene einordnenbar. In diesen Ebenen befinden sich die Zeithorizonte im Minutenbereich. Da sich die Abweichung der AAS im niedrigen einstelligen Sekundenbereich befindet, ist ihre Abweichung hinreichend gering.

Die Konstanz der Abweichung bei VFork, XRodA, XRodB und HFork+DEU deutet darauf hin, dass es sich hier um konstante Signal- und Rechenzeiten innerhalb der AAS handelt, welche nicht abgebildet wurden. Die verwendete AAS verfügt über ein komplexes internes Nachrichtensystem, das im Modell der AAS nicht abgebildet wurde. Die Zeit, welche die Zustandsmaschinen der AAS benötigen, um von einem Zustand zum nächsten zu wechseln, wurde ebenfalls nicht modelliert. Diese Signal- und Rechenzeiten können jedoch sehr abhängig von der Leistung des Rechners sein, auf dem die AAS ausgeführt wird. Im Rahmen der Experimente wurde ein Rechner, wie in Abschnitt 4.1 beschrieben, verwendet. Es ist zu erwarten, dass bei der Verwendung eines anderen Rechners diese Abweichung anders ausfallen könnte.

Gegen die Annahme, dass es sich hier um konstante Signal- und Rechenzeiten handelt, spricht jedoch, dass der Schritt Schrauben lediglich eine Abweichung von $3s$ aufweist. Während die anderen Schritte insgesamt zwei Ausschreibungen enthalten, enthält Schrauben drei Ausschreibungen. Unter der Annahme, dass es sich bei der Abweichung um nicht abgebildete Signal- und Rechenzeiten handelt, wäre es zu erwarten, dass die Abweichungen bei Schritt Schrauben größer sind als bei den anderen. Die Ursache dieses widersprüchlichen Ergebnisses muss weiter untersucht werden. Ein Grund könnte es sein, dass während der Messung des Schrittes Schrauben mehr Rechenleistung zur Verfügung stand, da anders als bei anderen Schritten dieser Schritt eine aktive Partizipation durch den Menschen erforderte. Dieser Umstand ist dadurch gegeben, dass die Schritte VFork, XRodA, XRodB und HFork+DEU alle durch den Roboter alleine ausgeführt und zurückgesetzt werden können. Für den Schritt Schrauben ist es jedoch erforderlich, dass der Mensch die Schrauben aus den Bauteilen entfernt, so dass der Prozess erneut ausgeführt werden kann. Während es somit möglich ist, dass bei den anderen Schritten der Mensch parallel zum Experiment andere Arbeiten am Rechner tätigt, ist dies beim Schritt Schrauben nicht möglich. Es steht somit mehr Rechenleistung für die AAS zur Verfügung. Dies ist eine mögliche Erklärung für das widersprüchliche Ergebnis des Schrittes Schrauben.

Die Abweichung der Signal- und Rechenzeiten macht 50% der Abweichung des Gesamtprozesses aus. In den Produktionsschritten VFork, XRodA, XRodB und HFork+DEU beträgt der Anteil der Signal- und Rechenzeiten an der Gesamtabweichung 80%. Für den Prozess Schrauben sind es lediglich 16%, da hier die Abweichungen des Schraubenziehers überwiegen. Es zeigt sich jedoch, dass für sämtliche Prozesse außer dem Prozess Schrauben die allgemeinen Modelle für den Großteil der Abweichungen verantwortlich sind. Die für den Anwendungsfall spezifischen Modelle machen lediglich einen kleinen Teil der Abweichung aus.

Die in Abschnitt 3 erlaubt es, ein beliebiges auf AAS basierendes Produktionssystem zu modellieren. Durch den Anwendungsfall konnte gezeigt werden, dass das Verhal-

ten aller Systembestandteile abgebildet werden kann. Dies umfasst sowohl das physische Verhalten realer Ressourcen als auch das Verhalten der digitalen Bestandteile wie der AAS und der Kommunikationsinfrastruktur. Es ist somit möglich, den gesamten Produktionsprozess eines beliebigen Anwendungsfalls in der Luftfahrt mithilfe der in dieser Arbeit entwickelten Methodik zu betrachten. Die in Abschnitt 2.8 aufgezeigte Forschungslücke konnte somit im Rahmen dieser Arbeit abgedeckt werden. Die AAS sowie ihre Bestandteile wurden als Modelle in SysML abgebildet und mit einem Modell der Kommunikationsinfrastruktur verknüpft. Es ist möglich, diese Modelle mit Simulationsmodellen beliebiger Ressourcen zu verbinden.

Die in dieser Arbeit vorgestellte Methodik ermöglicht die Untersuchung der Produzierbarkeit von Produkten in der Luftfahrt in frühen Entwicklungsphasen durch die Bereitstellung von Modellen zur Abbildung von AAS. Diese Modelle können eingesetzt werden, um vereinfacht Produktionsprozesse zu modellieren. Das Modell des Produktionssystems kann durch Modelle von Ressourcen ergänzt werden, um die Assets während der Produktion ebenfalls abzubilden. Hierbei ist ein beliebiger Detailgrad in der Modellierung der Assets möglich.

7 Zusammenfassung und Ausblick

Die Luftfahrt steht vor großen Herausforderungen. Um das Ziel der Klimaneutralität zu erreichen, müssen neue und disruptive Technologien angewendet werden. Um die Entwicklungsprozesse in der Luftfahrt zu beschleunigen, ist es notwendig, die Produktion bereits frühzeitig in die Entwicklung miteinzubinden. Zu diesem Zweck bedarf es moderner Modellierungs- und Simulationsmethoden, um das Produktionssystem ganzheitlich abzubilden.

Im Rahmen dieser Arbeit wurde eine Methodik zur ganzheitlichen Simulation eines auf AAS basierten Produktionssystems entwickelt. Hierzu wurden Modelle zur Abbildung von proaktiven AAS entwickelt. Die Kommunikation zwischen diesen wird ermöglicht durch eine Modellierung der Kommunikationsinfrastruktur zwischen den AAS. Die einzelnen AAS und das Modell der Kommunikationsinfrastruktur können zur Simulation des Produktionssystems in ein Gesamtmodell integriert werden. Mithilfe dieser Methodik lassen sich beliebige Anwendungsfälle umsetzen, durch die Modellierung spezifischer Ressourcen und Produkte. Durch die Modellierung des Verhaltens dieser Ressourcen können diese in der Simulation des Produktionssystems ebenfalls berücksichtigt werden.

Mithilfe der Methodik wurde ein Anwendungsfall in der Luftfahrt implementiert. Dies war die Produktion eines Teils des Crownmoduls. Die erstellten Modelle wurden anschließend mit der Produktion in der Realität verglichen. Es konnte dadurch gezeigt werden, dass die jeweiligen Modelle für ihren Zweck hinreichend genaue Abbildungen der Realität sind. Mithilfe der in dieser Arbeit vorgestellten Methodik ist es möglich, ein Produktionssystem in der Luftfahrt hinreichend genau abzubilden.

Die Genauigkeit der Simulation in dieser Arbeit kann weiter verbessert werden durch eine zusätzliche Verfeinerung der jeweiligen Modelle. Insbesondere das Modell der AAS kann durch die Simulation von zusätzlichen Signal- und Rechenzeit verfeinert werden. Dies ist im Besonderen notwendig, um die Methodik auf beliebige Produktionssysteme in der Luftfahrt anwendbar zu machen.

Die für den Anwendungsfall spezifischen Modelle der Endeffektoren des UR10 bieten ebenfalls großen Raum zur Verbesserung. Für den Greifer kann eine Verbesserung durch eine weitere Verfeinerung und Untersuchung des Verhaltens des Greifers sowie der Geometrie der Bauteile erreicht werden. Für den Schraubenzieher ist eine tiefgreifendere Überarbeitung des genutzten Modells notwendig.

Im Rahmen dieser Arbeit wurde lediglich eine geringe Menge an Ressourcen eingesetzt. Durch die Modellierung weiterer Ressourcen lassen sich komplexere Produktionsprozesse abbilden, welche einen besseren Einblick auf das Verhalten von AAS ermöglichen.

Die physische Modellierung des zu produzierenden Produkts wurde in dieser Arbeit nicht betrachtet. Die Einbindung eines physikalischen Modells des Produkts könnte jedoch einen positiven Einfluss auf die Güte der Simulation haben und die Abbildung zusätzlicher Produktionsprozesse ermöglichen.

Eine weitere Möglichkeit der Erweiterung der hier gezeigten Methodik ist die Modellierung zusätzlicher logistischer Aspekte. Im Rahmen dieser Arbeit wurde die Logistik als offene Lagerfläche, welche dem Roboter stets zugänglich war, abgebildet. Für die Simulation einer größeren Arbeitsstation oder sogar einer Fabrik wäre es notwendig, zusätzliche Logistikprozesse abzubilden. Dies könnte über die Modellierung von frei beweglichen Robotern erfolgen. Die Hochskalierung der hier angewandten Methodik auf eine Fabrik könnte ebenfalls die Untersuchung komplexerer Produkte ermöglichen.

Im Rahmen der Hochskalierung wäre es ebenfalls sinnvoll, die Modelle so zu erweitern, um es zu ermöglichen, Assets abzubilden, die aus anderen Assets bestehen. Dies würde es ermöglichen, Produktionssysteme abzubilden, welche aus höherrangigen AAS bestehen, welche in einer Hierarchie unter ihnen angeordnete AAS verwalten.

Das Modell der AAS, welches in dieser Arbeit eingesetzt wurde, ist ebenfalls eine sehr vereinfachte Sicht auf die Informationen, welche in einer AAS abgebildet werden. Es wurde sich hier primär auf das Verhalten der AAS fokussiert. Die Inklusion von weiteren Informationen, die einen Bestandteil der AAS bilden könnten, könnte ebenfalls sinnvoll sein.

Literatur

[1] Constanze Fetting. „The European green deal“. In: *ESDN report* 53 (2020).

[2] Volker Grewe u. a. „Evaluating the climate impact of aviation emission scenarios towards the Paris agreement including COVID-19 effects“. In: *Nature Communications* 12.1 (2021), S. 3841. ISSN: 2041-1723. DOI: 10.1038/s41467-021-24091-y. URL: https://www.nature.com/articles/s41467-021-24091-y.pdf.

[3] Matthias Finger u. a. *Navigating towards the decarbonisation of European aviation*. Bd. 2021/53, November 2021. Robert Schuman Centre for Advanced Studies Policy Briefs. Florence: European University Institute, 2021. ISBN: 9789294661098.

[4] Eckart Frankenberger u. a. „Technology Development and Future Aircraft Design as a Methodical Challenge“. In: *DS 31: Proceedings of ICED 03, the 14th International Conference on Engineering Design, Stockholm*. 2003, S. 491–492.

[5] Y. Asiedu und P. Gu. „Product life cycle cost analysis: State of the art review“. In: *International Journal of Production Research* 36.4 (1998), S. 883–908. ISSN: 0020-7543. DOI: 10.1080/002075498193444.

[6] Robert A. McDonald u. a. „Future aircraft concepts and design methods“. In: *The Aeronautical Journal* 126.1295 (2022), S. 92–124. ISSN: 0001-9240. DOI: 10.1017/aer.2021.110. URL: https://www.cambridge.org/core/services/aop-cambridge-core/content/view/417A8518731154FE1C05E6737A8F12A4/S000192402100110Xa.pdf/div-class-title-future-aircraft-concepts-and-design-methods-div.pdf.

[7] N. Kroll u. a. „DLR project Digital-X: towards virtual aircraft design and flight testing based on high-fidelity methods“. In: *CEAS Aeronautical Journal* 7.1 (2016), S. 3–27. ISSN: 1869-5590. DOI: 10.1007/s13272-015-0179-7. URL: https://link.springer.com/content/pdf/10.1007/s13272-015-0179-7.pdf.

[8] Dennis J. L. Siedlak u. a. „A digital thread approach to support manufacturing-influenced conceptual aircraft design“. In: *Research in Engineering Design* 29.2 (2018), S. 285–308. ISSN: 1435-6066. DOI: 10.1007/s00163-017-0269-0. URL: https://link.springer.com/content/pdf/10.1007/s00163-017-0269-0.pdf.

[9] Hendrik Meyer u. a. „DEVELOPMENT OF A DIGITAL TWIN FOR AVIATION RESEARCH“. In: *Deutscher Luft- und Raumfahrt Kongress*. 2020. URL: https://elib.dlr.de/136848/.

[10] Qiang Zhang u. a. „Digital thread-based modeling of digital twin framework for the aircraft assembly system“. In: *Journal of Manufacturing Systems* 65 (2022), S. 406–420. ISSN: 0278-6125. DOI: 10.1016/j.jmsy.2022.10.004. URL: https://www.sciencedirect.com/science/article/pii/S0278612522001753.

[11] Anouck Chan u. a. „The Aircraft and Its Manufacturing System: From Early Requirements to Global Design“. In: *Advanced Information Systems Engineering*. Hrsg. von Xavier Franch u. a. Lecture Notes in Computer Science. Cham: Springer International Publishing und Imprint Springer, 2022, S. 164–179. ISBN: 978-3-031-07472-1. DOI: 10.1007/978-3-031-07472-1_10. URL: https://link.springer.com/content/pdf/10.1007/978-3-031-07472-1_10.pdf.

[12] M. Weiss u. a. *MaSiMO - Development and Research of Industry 4.0 Components with a Focus on Experimental Applications of Proactive Asset Administration Shells in Data-Driven Maintenance Environments*. 2023. DOI: 10.25967/610125.

[13] Lucas Sakurada u. a. „A Methodology for Integrating Asset Administration Shells and Multi-agent Systems“. In: *2023 IEEE 32nd International Symposium on Industrial Electronics (ISIE)*. Piscataway, NJ: IEEE, 2023, S. 1–6. ISBN: 979-8-3503-9971-4. DOI: 10.1109/ISIE51358.2023.10227964.

[14] Heiner Lasi u. a. „Industrie 4.0“. In: *WIRTSCHAFTSINFORMATIK* 56.4 (2014), S. 261–264. ISSN: 0937-6429. DOI: 10.1007/s11576-014-0424-4.

[15] Birgit Vogel-Heuser. *Handbuch Industrie 4.0 Bd.4: Allgemeine Grundlagen*. Hrsg. von Thomas Bauernhansl und Michael ten Hompel. Berlin, Heidelberg, 2016. DOI: 10.1007/978-3-662-53254-6_12. URL: https://link.springer.com/content/pdf/10.1007/978-3-662-53254-6_12.pdf.

[16] Armin Roth. *Einführung und Umsetzung von Industrie 4.0: Grundlagen, Vorgehensmodell und Use Cases aus der Praxis*. Berlin und Heidelberg: Springer Gabler, 2016. ISBN: 978-3-662-48504-0. DOI: 10.1007/978-3-662-48505-7.

[17] Yang Lu. „Industry 4.0: A survey on technologies, applications and open research issues“. In: *Journal of Industrial Information Integration* 6 (2017), S. 1–10. ISSN: 2452414X. DOI: 10.1016/j.jii.2017.04.005.

[18] M. Stevenson * u. a. „A review of production planning and control: the applicability of key concepts to the make-to-order industry“. In: *International Journal of Production Research* 43.5 (2005), S. 869–898. ISSN: 0020-7543. DOI: 10.1080/00207540420002985 20.

[19] Hong-Chao Zhang. „Manufacturing Process Planning“. In: *Dorf (Hg.) 1994 – Handbook of design*, S. 587–616.

[20] Daniel Alejandro Rossit u. a. „Production planning and scheduling in Cyber-Physical Production Systems: a review“. In: *International Journal of Computer Integrated Manufacturing* 32.4-5 (2019), S. 385–395. ISSN: 0951-192X. DOI: 10.1080/0951192X.2019.1605199.

[21] International Society of Automation. *ISA-95 Standard: Enterprise-Control System Integration*. URL: https://www.isa.org/standards-and-publications/isa-standards/isa-95-standard (besucht am 13. 01. 2025).

[22] Magnus Åkerman. „Implementing Shop Floor IT for Industry 4.0“. Diss. 2018.

[23] Birgit Boss u. a. *Handbuch Industrie 4.0: Band 2: Automatisierung*. Hrsg. von Birgit Vogel-Heuser u. a. Berlin, Heidelberg, 2024. DOI: 10.1007/978-3-662-58528-3. URL: https://link.springer.com/content/pdf/10.1007/978-3-662-58528-3_139.pdf.

[24] Industrial Digital Twin Association. *Specification of the Asset Administration Shell: Part 1: Metamodel*. Frankfurt am Main, 2024-06-01. URL: https://industrialdigitaltwin.org/content-hub/aasspecifications/part1_metamodel (besucht am 15. 12. 2024).

[25] VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik u. a. *VDI/VDE 2193 Blatt 2 Sprache für I4.0-Komponenten - Interaktionsprotokoll für Ausschreibungsverfahren*. Düsseldorf, 2020-01-00. URL: https://katalog.slub-dresden.de/id/211-DE88731093.

[26] Aljosha Köcher u. a. „A reference model for common understanding of capabilities and skills in manufacturing“. In: *at - Automatisierungstechnik* 71.2 (2023), S. 94–104. ISSN: 0178-2312. DOI: 10.1515/auto-2022-0117.

[27] Saadia Dhouib u. a. „Papyrus4Manufacturing: A Model-Based Systems Engineering approach to AAS Digital Twins“. In: *2023 28th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. Piscataway, NJ: IEEE, 2023, S. 1–8. ISBN: 979-8-3503-3991-8. DOI: 10.1109/ETFA54631.2023.10275523.

[28] Enxhi Ferko u. a. „Towards Interoperable Digital Twins: Integrating SysML into AAS with Higher-Order Transformations“. In: *IEEE 21st International Conference on Software Architecture companion*. Piscataway, NJ: IEEE, 2024, S. 342–349. ISBN: 979-8-3503-6625-9. DOI: 10.1109/ICSA-C63560.2024.00063.

[29] INCOSE. *INCOSE systems engineering handbook*. John Wiley & Sons, 2023.

[30] Oliver Alt. *Modellbasierte Systementwicklung mit SysML*. Hanser eLibrary. München: Carl Hanser Fachbuchverlag, 2012. ISBN: 9783446431270. DOI: 10.3139/9783446431270. URL: http://www.hanser-elibrary.com/doi/book/10.3139/9783446431270.

[31] Junda Ma u. a. „Systematic Literature Review of MBSE Tool-Chains“. In: *Applied Sciences* 12.7 (2022), S. 3431. DOI: 10.3390/app12073431. URL: https://www.mdpi.com/2076-3417/12/7/3431/pdf?version=1648642331.

[32] No Magic. *Cameo Simulation Toolkit User Guide: 2022x*. 2022. URL: https://docs.nomagic.com/display/CST2022xR1/ (besucht am 16. 12. 2024).

[33] Eduard Babulak und Ming Wang. „Discrete Event Simulation: State of the Art“. In: *Discrete Event Simulations*. Hrsg. von Aitor Goti. Erscheinungsort nicht ermittelbar: IntechOpen, 2010. ISBN: 978-953-307-115-2. DOI: 10.5772/9894.

[34] Eldin Wee Chuan Lim, Hrsg. *Discrete Event Simulations: Development and Applications*. Erscheinungsort nicht ermittelbar: IntechOpen, 2012. ISBN: 9789535107415. DOI: 66196. URL: https://directory.doabooks.org/handle/20.500.12854/66196.

[35] Ola Batarseh und Leon F. McGinnis. „SysML to discrete-event simulation to analyze electronic assembly systems“. In: *Proceedings of the 2012 Symposium on Theory of Modeling and Simulation - DEVS Integrative M&S Symposium*. TMS/DEVS '12. San Diego, CA, USA: Society for Computer Simulation International, 2012. ISBN: 9781618397867.

[36] Niamat Ullah Ibne Hossain u. a. „Application of systems modeling language (SysML) and discrete event simulation to address patient waiting time issues in healthcare“. In: *Smart Health* 29 (2023), S. 100403. ISSN: 23526483. DOI: 10.1016/j.smhl.2023.100403.

[37] Timothy Sprock und Conrad Bock. „SysML Models for Discrete Event Logistics Systems“. In: *Journal of research of the National Institute of Standards and Technology* 125 (2020), S. 125023. ISSN: 1044-677X. DOI: 10.6028/jres.125.023.

[38] Oliver Schonherr und Oliver Rose. „First steps towards a general SysML model for discrete processes in production systems“. In: *Proceedings of the 2009 Winter Simulation Conference*. Hrsg. von Manuel D. Rossetti. Piscataway, NJ: IEEE, 2009, S. 1711–1718. ISBN: 978-1-4244-5770-0. DOI: 10.1109/WSC.2009.5429164.

[39] Vasilis Siatras u. a. „On the Use of Asset Administration Shell for Modeling and Deploying Production Scheduling Agents within a Multi-Agent System“. In: *Applied Sciences* 13.17 (2023), S. 9540. DOI: 10.3390/app13179540.

[40] Jakub Arm u. a. „Automated Design and Integration of Asset Administration Shells in Components of Industry 4.0“. In: *Sensors (Basel, Switzerland)* 21.6 (2021). DOI: 10.3390/s21062004.

[41] Universal Robots A/S. *User Manual: UR10E E-Series: SW 5.19*. 2024. URL: https://www.universal-robots.com/download/manuals-e-seriesur20ur30/user/ur10e/519/user-manual-ur10e-e-series-sw-519-english-international-en/ (besucht am 23. 12. 2024).

[42] Universal Robots A/S. *User Manual: UR10 E-Series: SW 5.12*. 2022. URL: https://www.universal-robots.com/download/manuals-e-seriesur20ur30/user/ur10e/512/user-manual-ur10e-e-series-sw-512-english-international-en/ (besucht am 26. 12. 2024).

[43] Robotic Inc. *Robotiq 2F-85 & 2F-140 for e-Series Universal Robots: Instruction Manual*. 2018.

[44] OnRobot A/S. *User Manual: for UR Robots with Polyscope 3/5: v6.2.0*. 2024.

[45] Bernd Sauer. *Grundlagen der Berechnung und Gestaltung von Maschinenelementen*. 10. Auflage. Bd. 1. Konstruktionselemente des Maschinenbaus / Bernd Sauer (Hrsg.) Berlin und Heidelberg: Springer Vieweg, 2023. ISBN: 978-3-662-66822-1. DOI: 10.1007/978-3-662-66823-8.

[46] Mateus Molina und Tagline Treichel. „Achieving Interoperability with MBSE and Asset Administration Shells: Integration of MATLAB/Simulink and BaSyx“. In: *2023 IEEE/ACM 11th International Workshop on Software Engineering for Systems-of-Systems and Software Ecosystems*. Piscataway, NJ: IEEE, 2023, S. 1–4. ISBN: 979-8-3503-0174-8. DOI: 10.1109/SESoS59159.2023.00005.

[47] F. Wilking u. a. „SysML 4 Digital Twins – Utilization of System Models for the Design and Operation of Digital Twins“. In: *Proceedings of the Design Society* 2 (2022), S. 1815–1824. DOI: 10.1017/pds.2022.184.

[48] Yassine Ghanjaoui u. a. „A Model-Based Approach for Evaluating and Validating the Sustainability of Production Systems“. In: *ONERA-DLR Aerospace Symposium (ODAS)*. 2023. URL: https://elib.dlr.de/195864/.

[49] United States Securities and Exchange Commission, Hrsg. *Exhibit 10.23: AIRCRAFT LEASE AGREEMENT*. 30.11.2009. URL: https://www.sec.gov/Archives/edgar/data/1520504/000119312513265084/d500205dex1023.htm (besucht am 09. 01. 2025).

[50] Frontline Systems, Inc. *Excel Solver*. URL: https://www.solver.com/excel-solver-online-help (besucht am 27. 12. 2024).

[51] Christian Troost u. a. „How to keep it adequate: A protocol for ensuring validity in agent-based simulation“. In: *Environmental Modelling & Software* 159 (2023), S. 105559. ISSN: 1364-8152. DOI: 10.1016/j.envsoft.2022.105559. URL: https://www.sciencedirect.com/science/article/pii/S1364815222002596.

Abbildungsverzeichnis

Tabellenverzeichnis