Institut für Softwaretechnologie

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit

Konzeption und Realisierung einer Cloud Manufacturing Orchestrations- und Planungsplattform

Daniel Pawlowicz

Studiengang:	Softwaretechnik
Prüfer/in:	Prof. Dr. Stefan Wagner
Betreuer/in:	Jonas Fritzsch, M.Sc. Matthias Strljic, M.Sc.
Beginn am:	15. Juli 2018
Beendet am:	15. Januar 2019

Masterarbeit

Konzeption und Realisierung einer Cloud Manufacturing Orchestrations- und Planungsplattform

Der Fakultät Konstruktions-, Produktions- und Fahrzeugtechnik der Universität Stuttgart zur Erlangung der Würde eines Master of Science Softwaretechnik (M.Sc.) vorgelegte Abhandlung

von

Daniel Pawlowicz

aus Stuttgart

Prüfer:	Prof. Dr. Stefan Wagner
Betreuer:	Jonas Fritzsch, M.Sc. Matthias Strljic, M.Sc.
Immatrikulationsnummer:	3203642
Eingereicht am:	15. Januar 2019

Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen der Universität Stuttgart.

Institut für Softwaretechnologie – Abteilung Software Engineering

Januar 2019

Master's Thesis

Conception and implementation of a Cloud Manufacturing orchestration and planning platform

A Master's Thesis presented to the faculty Konstruktions-, Produktions- und Fahrzeugtechnik, University of Stuttgart,
in partial fulfillment of the requirements for the degree of
Master of Science Software Engineering (M.Sc.).

by

Daniel Pawlowicz

from Stuttgart

First Examiner:	Prof. Dr. Stefan Wagner
Second Examiner:	Jonas Fritzsch, M.Sc. Matthias Strljic, M.Sc.
Student ID:	3203642
Submitted on:	15th January 2019

Institute for Control Engineering of Machine Tools and Manufacturing Units at University of Stuttgart.

Institute of Software Technology – Software Engineering Group

January 2019

Zusammenfassung

Cloud Manufacturing ist ein Paradigma, welches den Ansatz des Cloud Computings auf die Fertigung überträgt. Der Grundgedanke dafür ist, einen allgegenwärtigen und bedarfsgerechten Zugriff auf einen virtualisierten Pool an Ressourcen zu ermöglichen. Eine Plattform soll dabei die Zuordnung dieser Ressourcen zwischen den Teilnehmern in diesem Netzwerk übernehmen. Ein Vorteil davon ist, dass Hersteller und Dienstleister in einer neuen Form miteinander kollaborieren können. Fertigungsaufträge können aufgeteilt und einzelne Aufgaben an spezialisierte Hersteller oder Dienstleister abgegeben werden. Diese Orchestration der auf der Plattform verfügbaren Ressourcen, die als Services angeboten werden, muss geplant und koordiniert werden, um einen Kundenauftrag zu erfüllen.

Herausforderungen für dieses Paradigma sind der Datenschutz sowie die Sicherheit der Daten. Sensible Daten der Teilnehmer, Informationen über das Produkt oder Kenntnisse über Technologien und Prozesse müssen über die Plattform ausgetauscht werden.

Die vorliegende Arbeit befasst sich mit der Ausarbeitung eines Konzepts für eine Plattform, die eine Orchestrierung der Services sowie eine Planung von Aufträgen ermöglichen soll. Dabei sollen sensible Daten und geistiges Eigentum im Kontext des kollaborativen Cloud Manufacturing geschützt werden können. Dafür wurden bereits existierende wissenschaftliche Arbeiten analysiert und ausgewertet. Basierend auf diesen Erkenntnissen wurde ein Konzept für eine Plattform ausgearbeitet, die eine transparente Kollaboration der Teilnehmer des Netzwerks ermöglicht.

Die Plattform wurde mit einer Blockchain umgesetzt, die Teilnehmer veröffentlichen konzeptionierte Smart Contracts, die ihre Ressourcen, Dienstleistungen und Aufträge repräsentieren. Dabei werden Daten nur mit autorisierten Teilnehmern ausgetauscht. Zudem wird für den Austausch keine weitere Instanz benötigt, wie ein Plattform-Betreiber als Vermittler, der die Daten dann ebenfalls einsehen könnte.

Stichwörter: Cloud Manufacturing, Blockchain, Smart Contract

Abstract

Cloud manufacturing is a paradigm that describes the transfer of cloud computing to manufacturing. The fundamental idea is to enable an ubiquitous and needs-oriented access to a virtualized pool of resources. For that matter, a platform is designed to take over the task of allocating these resources in this network. One advantage of this is that manufacturers and service providers can collaborate with each other in a new form. Production orders can be split up and individual tasks can be handed over to specialized manufacturers or service providers. This orchestration of the resources which are offered as services on the platform must be planned and coordinated in order to fulfill a customer order.

Challenges to this paradigm are data protection and data security. Sensitive data of the participants, information about the product or knowledge about technologies and processes must be exchanged via the platform.

For this present paper, a concept for a platform which should enable the orchestration of services and the planning of orders. Sensitive data and intellectual property should be protected in the context of collaborative cloud manufacturing. Therefor, existing scientific papers were analyzed and evaluated. Based on these findings, a concept for a platform was developed that enables transparent collaboration between the participants of the network.

The platform was implemented with a blockchain. The participants publish conceptual smart contracts representing their resources, services and orders. Data is only exchanged with authorized participants and without the need for another entitiy, such as a platform operator as mediator, who could then also view the data.

Keywords: Cloud Manufacturing, Blockchain, Smart Contract

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Akronyme

ASC Auftrag-Smart-Contract 69, 72, 74, 78, 82, 83

CC Cloud Computing 2, 7–9, 13, 19, 31

CM Cloud Manufacturing 1–5, 7–13, 19, 30, 32, 36–38, 43–46, 48, 49, 52–54, 56, 58–61, 63, 67, 69, 75, 77, 79, 81, 82, 85, 87

ERP Enterprise-Resource-Planning 13, 29, 37, 54

EVM Ethereum Virtual Machine 75

GSC Genehmigungs-Smart-Contract 71, 72, 74, 81, 82, 84

HaaS Hardware-as-a-Service 12

HTTP Hypertext Transfer Protocol 38

IaaS Infrastructure as a Service 8

IoT Internet of Things 13, 19, 32, 36, 42, 52, 76

IPFS Interplanetary File System 77, 78

LSC Logistik-Smart-Contract 65, 66, 72, 78, 82

MaaS Manufacturing as a Service 2, 10

NIST National Institute of Standards and Technology 7, 8

OSC Operator-Smart-Contract 67–70, 72, 78, 81–83

OWL Web Ontology Language 27

PaaS Platform as a Service 8

PoS Proof-of-Stake 16

PoW Proof-of-Work 16, 75

PSC Provider-Smart-Contract 66

RSC Ressourcen-Smart-Contract 63–66, 74, 78, 82

SaaS Software as a Service 8

SCM Supply Chain Management 9, 43, 87, 88

VSC Verzeichnis-Smart-Contract 61, 66, 72, 78, 82

XaaS Anything as a Service 9

Kapitel 1

Einleitung

Produzierende Unternehmen stehen neuen Herausforderungen, die einschneidende Änderungen mit sich bringen werden, gegenüber. In erster Linie ist dies den Kunden geschuldet, die kosteneffizient auch kleine Stückzahlen für beispielsweise Prototypen produzieren lassen wollen. Auch können die produzierenden Unternehmen nicht allen Fertigungswünschen der Kunden gleichermaßen begegnen. Die Fertigungsabläufe der produzierenden Unternehmen müssen in der Lage sein, Anforderungen wie Skalierbarkeit und Anpassungsfähigkeit bei hoher Qualität und minimalen Kosten zu gewährleisten, um einen Wettbewerbsvorteil zu erzielen. Das führt dazu, dass Produktionsunternehmen zusammen arbeiten, um komplexe Fertigungen zu realisieren [1].

Das erfordert eine genaue Koordination der Aufgaben der einzelnen Unternehmen, da unterschiedliche Teilnehmer entlang der Produktions- und Lieferkette in dieser verteilten Produktionsumgebung zusammenarbeiten. Die einzelnen Fertigungssysteme in dieser Produktionsumgebung (als kollaboratives Cloud Manufacturing (CM) müssen sich dynamisch auf die Nachfrage anpassen können, um kritische und komplexe Produktionsprozesse unterstützen zu können.

Entlang des Fertigungsprozesses werden Informationen und Wissen sowie das Design des Produktes zwischen den involvierten produzierenden Unternehmen ausgetauscht. Dies erfordert transparente Prozesse, ohne aber den Wettbewerbsvorteil von Kunden und den Produzenten aufzuheben. Somit ist die Koordination der einzelnen Teilnehmer, die

Verwaltung der Produktionsaufträge sowie die Orchestration der erforderlichen Abläufe eine große Herausforderung [2].

1.1 Motivation

Industrie 4.0, auch Smart Manufacturing genannt, ist die Bezeichnung für eine Digitalisierung der Produktion. Dabei können vier Prinzipien unterschieden werden [3]:

- Vernetzung
- Informationstransparenz
- Technische Assistenz
- Dezentrale Entscheidungen

Erst mit dem Reifegrad der IT-Lösungen, den Fertigungstechnologien sowie ein umfassendes Verständnis und der Einsatz von Prozessketten sind Produktionsunternehmen in der Lage Industrie 4.0 umzusetzen. Einer der Kernaspekte der IT für Industrie 4.0 ist das Cloud Manufacturing, was die Anwendung des Paradigmas des Cloud Computing (CC) in der Fertigung meint. Es ermöglicht einen allgegenwärtigen und bedarfsgerechten Zugriff auf die Produktionsumgebung. Damit steht ein virtueller sowie skalierbarer Ressourcenpool zur Verfügung, der die Prozesse der Herstellung unterstützt und der von mehreren Teilnehmern genutzt werden kann [4][5].

Die Möglichkeiten des Einsatzes der Technologie werden derzeit diskutiert. Es ist eines der Trendthemen in der Branche, da es enormes Potenzial für die Fertigungsindustrie birgt. Mit dem Prinzip des CM kann ein umfassendes kollaboratives Netzwerk aus Produktionseinheiten aufgebaut werden. So ist das Manufacturing as a Service (MaaS) als Geschäftsmodell entstanden, welches Kunden ein hohes Maß an personalisierten Produkten bietet [6]. Dies wird durch ein Netzwerk aus produzierenden Unternehmen ermöglicht, welche jeweils durch deren Spezialisierung zu der Herstellung eines Produktes beitragen.

Eines der Probleme beim *Business in der Cloud* ist allerdings nach wie vor der Datenschutz sowie die Sicherheit der Daten [7]. Die Hauptsorge der Unternehmen ist, dass sensible Daten in der Cloud gespeichert werden und somit potenziell jeder beispielsweise auf die Technologien oder Informationen dieses Unternehmens Zugriff hat [8].

1.2 Zielsetzung

Der Fokus der vorliegenden Ausarbeitung liegt auf der Konzeption und der Entwicklung einer Orchestrations- und Planungsplattform für ein kollaboratives Cloud Manufacturing. Zunächst soll ein Überblick gegeben werden, welche Lösungen im Umfeld des CM bereits eingesetzt werden. Dabei wird der Schwerpunkt auf die Fragestellung gelegt, wie die Produktionsabläufe gekapselt werden und wie die Sicherheit sowie der Schutz der Daten gewährleistet werden kann.

Unter diesen Gesichtspunkten soll ein Konzept für eine Plattform ausgearbeitet werden, welches eine kollaborative Herstellungsplanung und -durchführung von Produkten ermöglicht. Dabei soll die folgende Forschungsfrage beantwortet werden:

Forschungsfrage: Wie muss im Kontext des kollaborativen Cloud Manufacturing eine Orchestrations- und Planungsplattform konzipiert sein?
Dabei sollen folgende Aspekte berücksichtigt werden:

- Wie kann der Schutz der Daten sowie des geistigen Eigentums gewährleistet werden?
- Wie kann die Zerlegung von Aufgaben erfolgen?
- Wie können die (zerlegten) Aufgaben den (verteilten) Fertigungsressourcen zugeordnet werden?

1.3 Methodik

Die vorliegende Arbeit ist in drei logische Abschnitte unterteilt. Der erste Teil behandelt die Grundlagen, gibt den Stand der Technik für das CM wieder und stellt verwandte Arbeiten vor. Für die Literaturrecherche wurde das Schneeballsystem wie von Wohlin beschrieben angewendet [9]. Bei dieser Technik werden die Literaturreferenzen auf weiterführende Literatur untersucht. Diese Literatur wird wiederum gesichtet und deren relevanten Literaturreferenzen analysiert. Wird eine bestimmte Quelle häufig zitiert, so kann davon ausgegangen werden, dass es sich um eine für diese Domäne maßgebliche Literatur handelt. Weiterhin können Quellen ermittelt werden, die neuer sind als die Ausgangsquelle, indem recherchiert wird, welche Literatur diese Ausgangsquelle zitiert.

Somit können auch, zeitlich vorwärts gerichtet, mehrere Iterationen erfolgen, um aktuelle Literatur zu recherchieren.

Die Limitierung dieser Technik ist, dass die Gefahr besteht, lediglich einzelne Forschungsströmungen abzudecken. Autoren, die eine andere Perspektive vertreten oder in einer anderen Richtung forschen, können unter Umständen nicht unter Verwendung des reinen Schneeballsystems, basierend auf wenigen Ausgangsquellen, ermittelt werden.

Die folgenden Ausgangsquellen wurden vom Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW) empfohlen und geben einen ersten Überblick über das CM im Allgemeinen sowie mit dem Fokus auf das kollaborative CM. Die Quellen sind:

- Development of a Novel Solution to Enable Integration and Interoperability for Cloud Manufacturing [10]
- Multi-agent-based scheduling in cloud manufacturing with dynamic task arrivals [11]
- Distributed manufacturing network models of smart and agile mini-factories [12]
- Collaborative Cloud Manufacturing: Design of Business Model Innovations Enabled by Cyberphysical Systems in Distributed Manufacturing Systems [13]
- Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives [14]

Ausgehend von diesen Quellen wurde eine rückwärts und vorwärts gerichtete Suche unternommen. Die vorwärts gerichtete Suche wurde mit Google Scholar durchgeführt.

Der zweite Abschnitt beschreibt die Lösungsfindung für ein Konzept einer Plattform für das CM. Für das Konzept wurden unterschiedliche Lösungsansätze aus dem Stand der Technik analysiert. Hierfür wurden verschiedene Kriterien ausgewählt und die Ansätze basierend auf diesen Kriterien bewertet.

Der letzte Abschnitt beschreibt die Entwicklung eines Konzepts für eine CM-Plattform. Dafür wurden die Anforderungen an die Plattform ausgearbeitet und in Frage kommende Technologien ausgewertet. Das entwickelte Konzepte wurde prototypisch implementiert und anhand eines entworfenem Szenarios validiert. Weiterhin werden in diesem Abschnitt mögliche zukünftige Arbeiten diskutiert.

1.4 Gliederung

Die vorliegende Arbeit ist in neun Kapitel gegliedert. Nach der Einleitung in **Kapitel eins** werden die für das weitere Verständnis relevanten Grundlagen in **Kapitel zwei** vorgestellt.

Kapitel drei gibt den aus der Literaturrecherche ausgearbeiteten Stand der Technik wieder. In **Kapitel vier** wird eine Analyse über bereits bestehende Konzepte für das CM beschrieben. **Kapitel fünf** stellt die Entwicklung eines Konzepts für eine CM-Plattform vor. Das **Kapitel sechs** erläutert das Vorgehen bei der Implementierung des ausgearbeiteten Konzepts. In **Kapitel sieben** wird ein Szenario vorgestellt, mit dem die Implementierung validiert worden ist und die Auswertung der Validierung. In **Kapitel acht** stellt verwandte Arbeiten vor. Der Schluss in **Kapitel neun** fasst die Kernelemente dieser Arbeit zusammen und soll einen Ausblick auf mögliche weitere Arbeiten in diesem Bereich geben.

Kapitel 2

Grundlagen

Das nachfolgende Kapitel behandelt die Grundlagen, die für den weiteren Verlauf dieser Ausarbeitung relevant sind. Als wichtige Kerntechnologie für das CM wird CC vorgestellt, anschließend wird CM als Konzept erläutert. Für den Lösungsansatz soll die Blockchain Technologie als Ausgangsbasis dienen, deren Konzept zum Abschluss vorgestellt wird.

2.1 Cloud Computing

CC ist ein Modell, dass die Bereitstellung von verschiedenen virtuellen IT-Ressourcen, wie Server oder Anwendungen, bei Bedarf ermöglicht [15]. Dafür werden Rechen- und Speicherressourcen von Anbietern als virtualisierter Pool bereitgestellt. Somit haben Nutzer eine effiziente und dynamische Möglichkeit, diese bei Bedarf in Anspruch zu nehmen. Die Abrechnung und Zahlung für die genutzten Services erfolgen je nach in Anspruch genommener Leistung. Der Vorteil ist dabei, dass die Nutzer selbst keine Anschaffungen für IT-Ressourcen tätigen müssen. Sie haben so die Möglichkeit, ohne Investitionen in Hardware oder Software Ressourcen zu nutzen und je nach Bedarf abzurechnen.

Das National Institute of Standards and Technology (NIST) veröffentlichte 2011 für das CC eine Definition [16]. Demnach weist das CC fünf Charakteristika auf: Der *On-demand Self Service* (Bedarfsgerechter Selbst-Service) soll die Bereitstellung der IT-

Ressourcen automatisch und ohne Interaktion mit dem Anbieter erlauben. Der *Broad Network Access* ermöglicht den Nutzern den Zugang über das Internet mithilfe von Standard-Mechanismen. Das *Resource Pooling* bietet den Nutzern einen gemeinsamen Pool an Ressourcen, die von mehreren Nutzern in Anspruch genommen werden können. *Rapid Elasticity* bedeutet, dass die Services schnell zur Verfügung gestellt oder auch entfernt werden können. Der *Measured Service* bietet eine transparente Überwachung der Nutzung der Ressourcen.

In Bezug auf die Bereitstellung der Services werden von NIST vier Modelle unterschieden. Die *Private Cloud* ermöglicht den exklusiven Zugang für genau eine Organisation. Der Service einer *Public Cloud* kann von mehreren Nutzern unterschiedlicher Organisationen oder der Öffentlichkeit gemeinsam genutzt werden. Bei der *Community Cloud* handelt es sich um ein Bereitstellungsmodell, bei dem bestimmte Interessensgruppen Zugang zu derselben Cloud-Umgebung haben. Eine *Hybrid Cloud* vereint zwei oder mehrere unterschiedliche Bereitstellungsmodelle.

Weiterhin werden drei grundlegende Service-Modelle unterschieden. Das *Infrastructure as a Service (IaaS)* als Modell bietet dem Nutzer den Zugriff auf IT-Ressourcen wie Server oder Netzwerke über das Internet. Auf dieser Infrastruktur können vom Nutzer virtuelle Maschinen eingerichtet sowie Betriebssysteme installiert werden. Weiterhin können Datenbanken und Unternehmensanwendungen bereitgestellt und genutzt werden. Bei dem Modell *Platform as a Service (PaaS)* kann der Nutzer auf ein bereits eingerichtetes System zugreifen. Anwendungen können installiert und direkt genutzt werden. Bei dem Modell *Software as a Service (SaaS)* können die Nutzer über das Internet Anwendungen, wie beispielsweise Microsoft Office 365, verwenden. Diese sind vom Provider bereits installiert und der Nutzer muss nicht die darunterliegenden Layer (Schichten), wie das Betriebssystem oder notwendige Datenbanken, einrichten. Er kann sich beispielsweise über ein Frontend einloggen und hat Zugriff auf die Anwendung.

2.2 Cloud Manufacturing

Das CM ist ein Paradigma, welches die Übertragung der Idee von CC auf die Fertigung beschreibt [17]. Die Nutzung der Cloud ermöglicht beim CM einen allgegenwärtigen und bedarfsgerechten Netzwerkzugriff, es wird somit ein virtualisierter, gemeinsamer und konfigurierbarer Ressourcenpool unterstützt. So sollen die Herstellungsprozesse und das

gesamte Supply Chain Management (SCM) davon profitieren. Auch soll die Minimierung von Interaktionen der Teilnehmer den Verwaltungsaufwand senken und Ressourcen unmittelbar nach der Nutzung auf Basis von *Pay-as-you-go* wieder freigeben werden [18]. Pay-as-you-go bedeutet, dass man erst dann für eine Leistung bezahlen muss, wenn man diese auch in Anspruch genommen hat.

Die Einbeziehung von CC als zugrunde liegende Technologie ist einer der Hauptunterschiede zwischen CM und anderen IT-unterstützten Paradigmen für die Fertigung. Es eröffnet damit die Möglichkeit, Aktivitäten zur Fertigung oder Verarbeitung von Produkten als Dienstleistung beziehungsweise Service in einer verteilten Umgebung anzubieten. Diese Aktivitäten werden im Bereich des CM als *Anything as a Service (XaaS)* bezeichnet. Dies soll das Konzept von „Integration von verteilten Ressourcen“ als auch das Konzept der „Verteilung von integrierten Ressourcen“ widerspiegeln [19]. Damit kann der Kunde als Auftraggeber eines Produktes sowie der Produzent gezielt strategisches Outsourcing und On-Demand Dienste nutzen beziehungsweise anbieten [4]. So können beispielsweise Bereiche wie Produktdesign, Fertigung oder auch Tests durch andere Teilnehmer der Cloud-Umgebung ausgeführt werden.

2.2.1 Prinzip einer Cloud Manufacturing Plattform

Das Zusammenkommen von Angebot und Nachfrage dieser Dienstleistungen soll durch eine CM-Service-Plattform ermöglicht werden. Die Plattform übernimmt eine intelligente Zuordnung der Aufgaben der Kunden zu den produzierenden Teilnehmern, welche dann die Fertigung als Service übernehmen [19].

Abbildung 2.1 zeigt das Prinzip einer CM-Plattform, welche die Anbieter mit den Kunden zusammenbringt. Dabei stellen die Anbieter als Provider auf der Plattform ihre Fertigungsressourcen und -kapazitäten als Input zur Verfügung. Dem gegenüber steht der Kunde, der seine Fertigungswünsche als Anfrage an ebendiese sendet. Das Gefüge von Input und Anfragen wird auf der CM-Plattform gesammelt und verwaltet, der Kunde erhält Zugriff auf den virtuellen Ressourcenpool. Die Kundenanfragen werden über die CM-Plattform analysiert und den Ressourcen zugeordnet, der Provider erhält dann den Auftrag [20].

Die produzierenden Teilnehmer werden als Partner nach den Möglichkeiten der Services und der Fähigkeiten den Anforderungen des Kunden beziehungsweise des Kun-

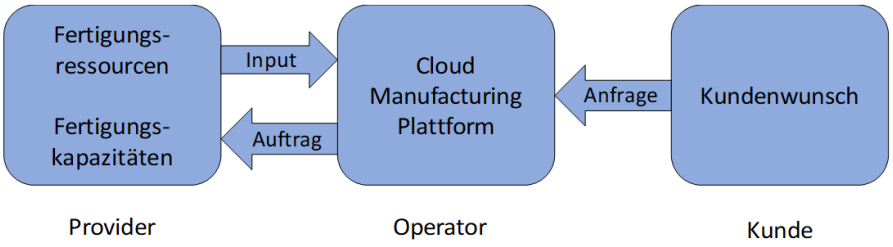

Abbildung 2.1: Prinzip einer CM-Plattform in Anlehnung an Ning et al. [20]

denauftrages entsprechend ausgewählt. Die Besonderheit am CM ist dabei, dass diese zusammen als gleichberechtigte Partner miteinander interagieren und zusammenarbeiten um einen Auftrag zu erfüllen. Die CM-Service-Plattform übernimmt dafür die Orchestration der Dienstleistungen für das MaaS und gestaltet den Produktionsprozess dynamisch [21].

2.2.2 Bereitstellungsmodelle

Im CM können in Bezug auf die Breitstellung von Services vier verschiedene Modelle unterschieden werden [17].

Public
Bei diesem Modell werden die Produktionsressourcen und -kapazitäten eines Unternehmens öffentlich zur Verfügung gestellt. Der Anbieter kann ungenutzte Ressourcen gegen eine Gebühr anbieten und Kunden können die Ressourcen in Anspruch nehmen, die sie für einen bestimmten Zeitraum benötigen.

Private
Die Services werden innerhalb eines Unternehmens zur Verfügung gestellt. So können beispielsweise global verteilte Abteilungen einen gemeinsamen Service nutzen, um einen Kostenvorteil zu erzielen, da Ressourcen so gebündelt werden können. Die Sicherheit bei diesem Modell ist hoch, da nur unternehmensintern auf die Ressourcen zugegriffen werden kann.

Hybrid
Dieses Modell ist ein Zusammenschluss aus mehreren unterschiedlichen Typen. So kann beispielsweise eine Kombination aus einer Public Cloud sowie einer Private Cloud angeboten werden. Somit kann sichergestellt werden, dass sensible Daten nur innerhalb der Private Cloud ausgetauscht werden, während bestimmte Services über Schnittstellen, über die Grenzen der Private Cloud hinaus, öffentlich zur Verfügung gestellt werden können.

Community
Bei diesem Modell werden Services über zwei oder mehrere Organisationsgrenzen hinweg miteinander geteilt. Die teilnehmenden Organisationen verfolgen bei dieser Zusammenarbeit ähnliche oder gemeinsame Interessen und Ziele. So können bestimmte komplexe Fertigungsaufgaben, die beispielsweise hohe Sicherheitsanforderungen haben, von mehreren produzierenden Unternehmen realisiert werden.

2.2.3 Teilnehmer des Cloud Manufacturing

Die Teilnehmer im CM können drei verschiedenen Rollen zugeordnet werden [17]. Ein Teilnehmer kann auch mehrere Rollen einnehmen und kann beispielsweise als Provider sowie als Consumer im Netzwerk auftreten.

Provider
Die Provider verfügen über die Fertigungsressourcen sowie -fähigkeiten und erbringen die Dienstleistung. Sie sind am gesamten Lebenszyklus des Herstellungsprozesses beteiligt. Diese Produktionsressourcen werden durch den Provider virtualisiert und als Service zur Verfügung gestellt.

Consumer
Die Consumer treten als Nachfrager im Netzwerk auf. Sie nehmen die Dienstleistungen der Provider je nach Bedürfnis in Anspruch.

Nach Wu et al. werden die Teilnehmer in vier Gruppen eingeteilt [22]. Abbildung 2.2 visualisiert diese Teilnehmer.

Operator
Der Operator ist verantwortlich für den Betrieb und das Management des CM-Systems. Er betreibt die Plattform und vermittelt die Ressourcen und Services der Provider an die Kunden, die die entsprechenden Fertigungswünsche haben. Sie befassen sich mit der Verwaltung des CM-Systems und stellen die Technologien und Services für die Planungen der Kundenaufträge der CM-Plattform zur Verfügung.

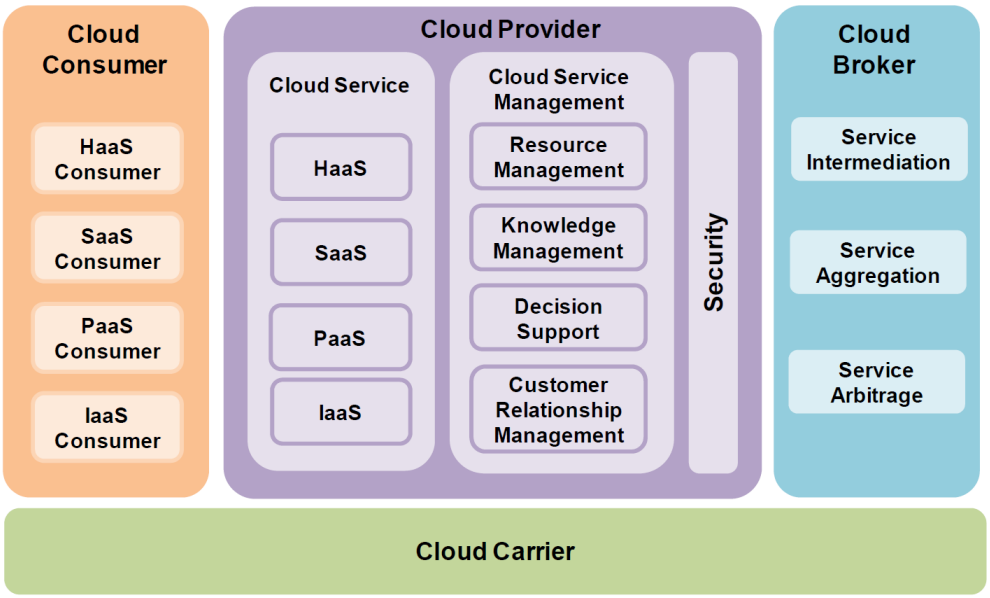

Abbildung 2.2: CM Teilnehmer nach Wu et al. [22]

Der *Cloud Consumer* nutzt die Services, die über die Cloud Plattform angeboten werden. Der *Cloud Provider* stellt die verschiedenen Services zur Nutzung bereit. Der *Cloud Broker* verwaltet die vom Cloud Provider zur Verfügung gestellten Services und vermittelt zwischen dem Cloud Consumer und dem Cloud Provider. Der *Cloud Carrier* ist der Vermittler zwischen den drei Teilnehmern und stellt die Konnektivität zwischen den Cloud Consumern, den Cloud Providern und den Cloud Brokern her.

Die Zusammenarbeit im CM erfolgt über die Anfrage des Cloud Consumers nach einen Service zur Nutzung. Die Bereitstellung der Services wird in vier Arten unterschieden:

Hardware-as-a-Service (HaaS)
Im Kontext von CM werden damit Maschinen und Werkzeuge bezeichnet. Diese werden

je nach Bedarf nur für die Nutzung gemietet. Ein Consumer kann so beispielsweise einen 3D-Drucker für die Herstellung eines Produktes nutzen und muss diesen nicht kaufen.

Software-as-a-Service (SaaS) im CM
Das können Softwareanwendungen wie Enterprise-Resource-Planning (ERP) Systeme oder spezielle CAD/CAM-Programme sein, die von Designer oder Entwicklern nach Bedarf genutzt werden. Eine Lizenz für die Programme muss für die Nutzung nicht gekauft werden.

Platform-as-a-Service (PaaS) im CM
Im CM kann eine Plattform mit einer speziellen Umgebung, wie beispielsweise ein High-Speed-Client, die Herstellungskosten senken und die Entwicklungszeit reduzieren. Die dafür benötigte Hardware kann so nur bei Bedarf, wie beispielsweise Tests eines fertigen Designs, genutzt werden und bedarf keiner teuren Anschaffung.

Infrastruktur-as-a-Service (IaaS) im CM
Das sind wie im CC Computerressourcen wie Server oder Speicherplatz. Spezielle Anforderungen, wie High-Performance Berechnungen können auf der Basis von *pay-as-you-go* genutzt werden.

2.2.4 Schlüsseltechnologien für das Cloud Manufacturing

Das CM wird erst ermöglicht durch den fortgeschrittenen Reifegrad von IT-Lösungen und verschiedenen Technologien [23]. Diese sind das bereits vorgestellte CC, der Umgang von Big Data, als große und komplexe Datenmengen, sowie die Digitalisierung der Produktionsstätten [4] [21].

Internet of Things (IoT), beziehungsweise das Internet der Dinge, ist die Vernetzung der Maschinen damit eine Maschine-zu-Maschine oder Mensch-zu-Maschine Kommunikation ermöglicht wird. Dafür werden die Maschinen mit Sensoren und der Möglichkeit zur Kommunikation ausgestattet. Somit sind sie in ständiger Verbindung mit ihrer Umwelt und können relevante Daten über beispielsweise ihren Zustand zur Verfügung stellen [24].

Big Data Analysen sind für die Echtzeitauswertung von Maschinen- und Sensordaten essenziell. Anhand der Ergebnisse der Analysen kann der Status einer Produktionsressource sowie deren Kapazität abgerufen werden. Weiterhin können Vorhersagen darüber getroffen werden, wie die Auslastung von Maschinen oder ganzer Produktionsstätten zu bestimmten Zeiträumen sein könnte. Auch in Bezug auf das Produktdesign können Big Data Analysen herangezogen werden, um das Produkt beispielsweise anhand von bereits vorhandenem Wissen zu optimieren [14].

2.3 Die Blockchain Technologie

Im Wesentlichen ist eine Blockchain ein öffentliches Hauptbuch (Ledger) und beinhaltet dauerhaft Informationen zu jeder durchgeführten Transaktion, die über die Blockchain durchgeführt worden ist. Das System für die Transaktionen basiert nicht auf Vertrauen, sondern auf einem kryptografischen Nachweis. Dadurch wird eine zentralisierte Instanz, wie eine Bank bei Geldtransaktionen, nicht mehr benötigt und die Teilnehmer können direkt Transaktionen durchführen [25].

2.3.1 Funktionsweise

Eine Blockchain kann als eine Kette von Hashes von digitalen Signaturen angesehen werden. Als Hash wird eine Zahl mit einen eindeutigen Wert bezeichnet, der eine Zeichenkette darstellt. Jede Transaktion wird in einen Block geschrieben, validiert und mit dem Vorgänger-Block verbunden. Der Block enthält den kryptografischen Hashwert des Vorgänger-Blocks sowie einen Zeitstempel der Erstellungszeit. Diese werden auf allen Knoten, den teilnehmenden Computern, kopiert und sind somit identisch, öffentlich und für jeden einsehbar [26]. Der erste Block hat keinen vorangegangenen Block und wird als Genesis-Block bezeichnet. Ist ein Block hinzugefügt worden, kann dieser nicht mehr geändert oder entfernt werden, was die Integrität der Blöcke und Transaktionen gewährleistet. Mit dieser Transaktionshistorie entsteht eine chronologische Reihenfolge [27].

Ein weiterer Bestandteil eines Blocks ist der Merkle-Baum. Dieser ist eine digitale Signatur aller Transaktionen, die in einem Block enthalten sind. Wenn ein Block angehängt werden soll, wird dieser zunächst mit einem Konsensmechanismus validiert, der auch die Gültigkeit der Transaktionen im Block prüft. Das erfolgt über das sogenannte Mining,

bei dem bestimmte Teilnehmer des Netzwerkes, als Miner bezeichnet, die Transaktionen verifizieren. Auch wird hiermit sichergestellt, dass im gesamten Netzwerk lediglich ein Block angehängt wird, der zum letzten Block im Ledger wird [28].

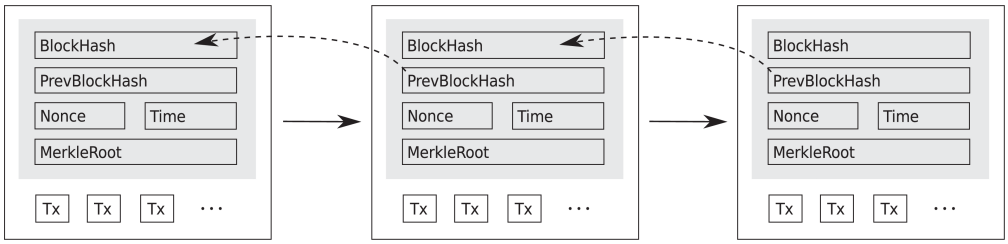

Abbildung 2.3: Vereinfachte Darstellung der Blockchain nach Tschorsch & Schauermann [28]

Abbildung 2.3 zeigt eine vereinfachte Darstellung der Blockchain. Zu sehen ist der BlockHash, der vom PrevBlockHash im nachfolgendem Block referenziert wird. Nonce ist eine Pseudozufallszahl und wird für das Mining verwendet, während das Time Feld den Zeitstempel der Erstellungszeit repräsentiert. Der MerkleRoot ist die digitale Signatur aller Transaktionen als Hashwert in dem jeweiligen Block und wird aus dem Merkle-Baum berechnet. Die Transaktionen werden als Felder mit der Bezeichnung *Tx* in der Darstellung aufgeführt [28].

Eine Blockchain kann entweder öffentlich oder privat zugänglich sein. Bei einer privaten Blockchain können nur bestimmte Teilnehmer, die einen Zugang haben, auf die Blockchain zugreifen. Bitcoin mit seinem dezentralen Krypto-Währungssystem ist ein Beispiel für eine öffentliche Blockchain. Bei einer öffentlichen Blockchain kann jedermann Teilnehmer dieser werden und Transaktionen vornehmen [29].

Die Konzeption der Blockchain als dezentrales Netz aus Knoten ist ein weiteres Merkmal und bietet durch diese Dezentralisierung ein fehlertolerantes Netzwerk. Fällt ein Knoten aus, ist das Gesamtsystem nicht beeinträchtigt. Kommt ein neuer Knoten hinzu, wird die Blockchain auf diesen neuen Knoten repliziert und er nimmt als Teilnehmer am Netzwerk teil [30].

2.3.2 Konsensmechanismus

Bei einer zentralisierten Architektur ist eine Instanz für die Richtigkeit der Daten zuständig. Bei einer dezentralen Architektur, wie es bei der Blockchain der Fall ist, sind

die Daten auf verschiedene Knoten repliziert. Diese Daten müssen nach jeder Transaktion auf allen Knoten identisch sein, dafür muss lediglich ein valider Block an das Ende der Blockchain angehängt werden [27]. Die Teilnehmer der Blockchain müssen über diesen Zustand übereinstimmen – beziehungsweise zu einem Konsens kommen. Demnach werden für den Erhalt der Blockchain und somit ihrer Daten Konsensmechanismen eingesetzt [26].

Proof-of-Work (PoW)
Bei diesem Konsensverfahren werden mathematische Aufgaben durch die Miner bearbeitet. Eine solche Aufgabe kann beispielsweise das Finden eines Ergebnisses mit einer bestimmten Eigenschaft sein, die aus einer Hashfunktion resultiert. Der Grad der Schwierigkeit für die Berechnung wird entsprechend angepasst, sodass die Miner in regelmäßigen Abständen eine Lösung für die gestellte Aufgabe finden können. Der Miner, der die Lösung zuerst berechnen konnte, erhält eine Vergütung für die investierte Arbeit [30].

Proof-of-Stake (PoS)
Bei diesem Verfahren wird die Entscheidung darüber gefällt, welcher Miner den nächsten Block anhängen kann, indem dieser anhand der Anteile an der Blockchain-Währung ausgewählt wird. Je höher der Anteil ist und je länger dieser Anteil bereits gehalten wird, desto höher fällt die Gewichtung aus. Über ein Zufallsprinzip wird dann entschieden, wer den nächsten Block validiert und anhängt, wobei eine höhere Gewichtung die Chance erhöht ausgewählt zu werden. Ein Vorteil hierbei ist, dass bei diesem Auswahlverfahren das Interesse der Teilnehmer am Netzwerk berücksichtigt wird. Das Prinzip geht von der These aus, dass je mehr und länger ein Teilnehmer Anteile am Netzwerk hält, desto mehr Interesse hat er dieses zu unterstützen. Ein weiterer Vorteil ist, dass im Gegensatz zum PoW keine Rechenleistung aufgewendet werden muss und das Verfahren somit ökonomischer ist [30].

2.3.3 Smart Contracts

Ein Smart Contract ist ein Computerprogramm und basiert auf einem computergesteuerten Transaktionsprotokoll. Wenn bestimmte Bedingungen zutreffen, werden die im Smart Contract hinterlegten Vereinbarungen zwischen den Parteien durchgesetzt. So ein

Programm wird automatisch ausgeführt, ohne dass eine weitere Instanz benötigt wird. Das heißt, dass zwei Parteien direkt und ohne menschliche Interaktion Transaktionen durchführen können. Die Vorteile von einem derartigen automatischen Mechanismus sind unter anderem Kosten- sowie Zeitersparnisse, da ein Smart Contract die Bedingungen einem Algorithmus folgend auswertet und eine weitere Prüfung oder eine Aktion von einem Menschen nicht mehr notwendig ist [26].

Da die Smart Contracts auf der Blockchain auf jedem Knoten repliziert werden, entsteht auch eine hohe Transparenz. Sie sind immutable, daher nach der Veröffentlichung unveränderlich und erhalten eine eindeutige Adresse. Das Programm eines Smart Contracts, der auf einer Blockchain veröffentlicht wurde, kann von jedem Teilnehmer eingesehen werden. Das Programm der Smart Contracts wird automatisch ausgeführt, sobald ein bestimmtes und im Programm festgelegtes Ereignis eintritt. So können alle geschäftlichen Transaktionen oder vertraglichen Vereinbarungen automatisiert durch Smart Contracts abgewickelt werden [29].

Kapitel 3

Stand der Technik

Dieses Kapitel besteht aus einen Literaturüberblick und gibt den Stand der Technik für das CM wieder. Weiterhin werden Ansätze aus der Literatur vorgestellt, die den Einsatz der Blockchain Technologie im Kontext von Industrie 4.0 thematisieren.

3.1 Schlüsselfunktionen für das Cloud Manufacturing

Für das CM können neben Schlüsseltechnologien (Kapitel 2.2.4), wie dem CC, auch Schlüsselfunktionen beschrieben werden. Diese können nach bestimmten Kriterien klassifiziert werden und ermöglichen das Identifizieren von Funktionen, die für das CM benötigt werden. Lu et al. schlagen eine Taxonomie vor, welche auf einen objektorientierten Ansatz beruht. Abbildung 3.1 stellt eine hierarchische Struktur von Schlüsselfunktionen für das CM dar, die als Grundlage für eine Plattform verwendet werden können [19]. Dabei werden Funktionen für Fertigungsressourcen und Funktionen für Fertigungsdienstleistungen unterschieden.

Funktionen für Fertigungsressourcen

Physical Resource Perception (Wahrnehmung der physischen Ressourcen): Um die verteilten Ressourcen für die Fertigung im IoT dynamisch zu organisieren und zur Verfügung zu stellen, müssen diese wahrgenommen und zentral verwaltet werden. Dies kann über

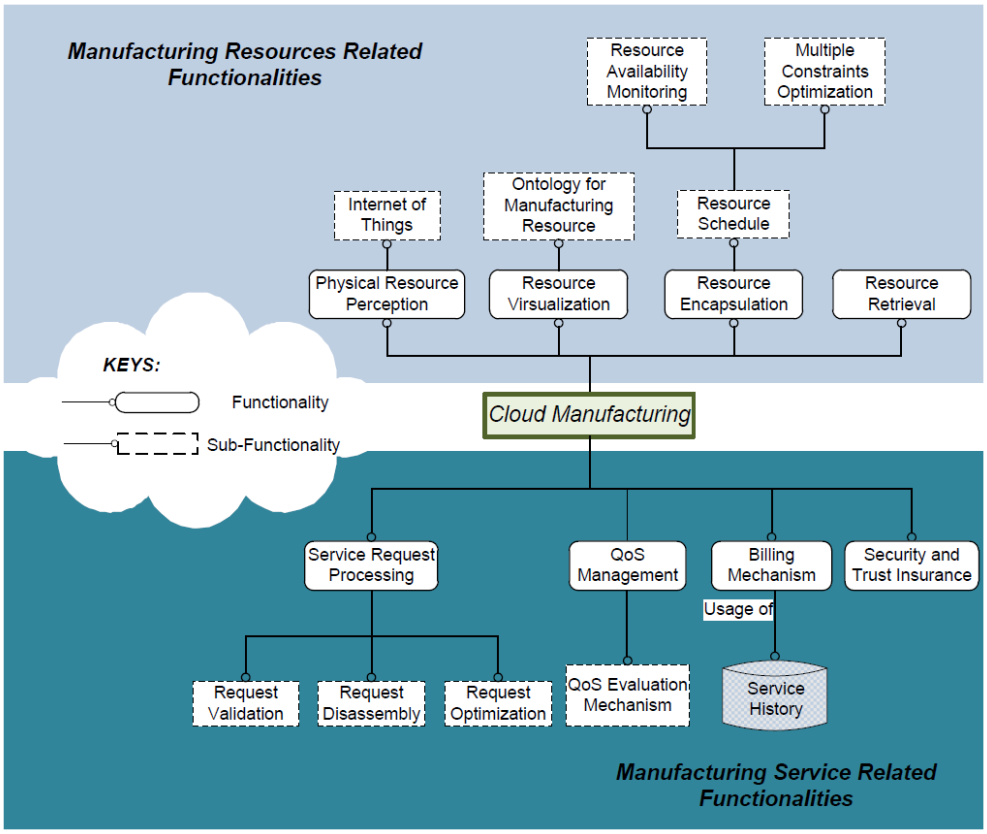

Abbildung 3.1: Hierarchische Struktur von Schlüsselfunktionen für das CM nach Lu et al. [19]

Technologien wie RFID oder GPS erfolgen, um die Informationen über die Ressourcen zur Fertigung zu sammeln und an das CM-System zu übertragen [19].

Resource Virtualization (Virtualisierung der Ressourcen): Diese Funktionalität meint die Abstraktion der physischen Ressourcen als virtuelle Maschinen und Werkstätten. Dabei sollen die Fertigungsressourcen mittels Ontologie beschrieben werden. Von dieser Beschreibung ist die Effizienz der Service- sowie der Ressourcenbereitstellung maßgeblich abhängig [19].

Resource Encapsulation (Kapselung der Ressourcen): Meistens muss eine Fertigungsaufgabe zerlegt werden um von verschiedenen Fertigungsressourcen bearbeitet werden zu können. Nach der Zerlegung werden die resultierenden Teilaufgaben über einen Optimierungsprozess an die optimalen Fertigungsressourcen übermittelt [19].

Resource Retrieval (Beschaffung der Ressourcen): Um eine Fertigungsressource abrufen zu können muss diese klar durch eine Beschreibung definiert sein. Über die Anfrage kann dann eine geeignete Ressource bestimmt werden [19].

Funktionen für Fertigungsdienstleistungen

Service Request Processing (Bearbeitung von Serviceanfragen): Über standardisierte Formate wie CAD oder STEP soll eine Serviceanfrage mit weiteren Anforderungen wie den Kosten- oder Zeitraumaspekt muss die Plattform die Realisierbarkeit validieren können. Fehler oder eine zusätzliche manuelle Prüfung sollen frühzeitig erkannt werden, damit nicht unnötige Ressourcen verbraucht werden [19] [31].

Quality of Service Management: Diese Kern-Funktionalität zielt darauf ab Kunden das Vertrauen in den Service zu geben. Dabei muss einerseits das Zustandekommen des Vertrages und andererseits die Überwachung der Fertigung betrachtet werden.

Billing Mechanism (Abrechnungsmechanismus): Beim CM soll das Preismodell Pay-as-you-go als Abrechnung gelten. Wenn aber mehrere Dienstleister zu einem Service beziehungsweise Produkt beitragen, ist der Abrechnungsprozess wesentlich komplexer. Für diesen Schritt können Preisberechnungen mit künstlicher Intelligenz und basierend auf historischen Werten ran gezogen werden [17] [19].

Security and Trust (Sicherheit und Vertrauen): Durch Komplikationen im Datenschutz und der -sicherheit unternehmen viele Kunden und Anwender nur sehr zögernd den Schritt in die Cloud. Beim CM werden sehr sensible und vertrauliche Daten in der Cloud verarbeitet. Das sind Kerntechnologien, Konstruktionsdateien und personenbezogene sowie unternehmensrelevante Daten [32] [19].

3.2 Herstellungsprozess im Cloud Manufacturing

Der Prozess der Produktherstellung ist im CM hingegen zum traditionellen Vorgehen abweichend [33]. Insbesondere ist die Aufteilung der einzelnen Aufgaben zur Herstellung eines konkreten Produktes komplexer. Auch die Verteilung der einzelnen Aufgaben auf die verschiedenen Hersteller in einem distributiven Hersteller-Netzwerk sind wesentlich umfangreicher [34].

Der Kernaspekt von CM und MaaS ist, Kunden Fertigungsressourcen als Dienst anzubieten und bereitzustellen, welche über das Internet abgerufen und genutzt werden kann. Abbildung 3.2 zeigt den Fertigungsprozess als Workflow nach Tao [33]. Dieser Service Lifecycle wird in vier Phasen eingeteilt:

- Service generation
- Service pre-application
- Service application
- Service after-application

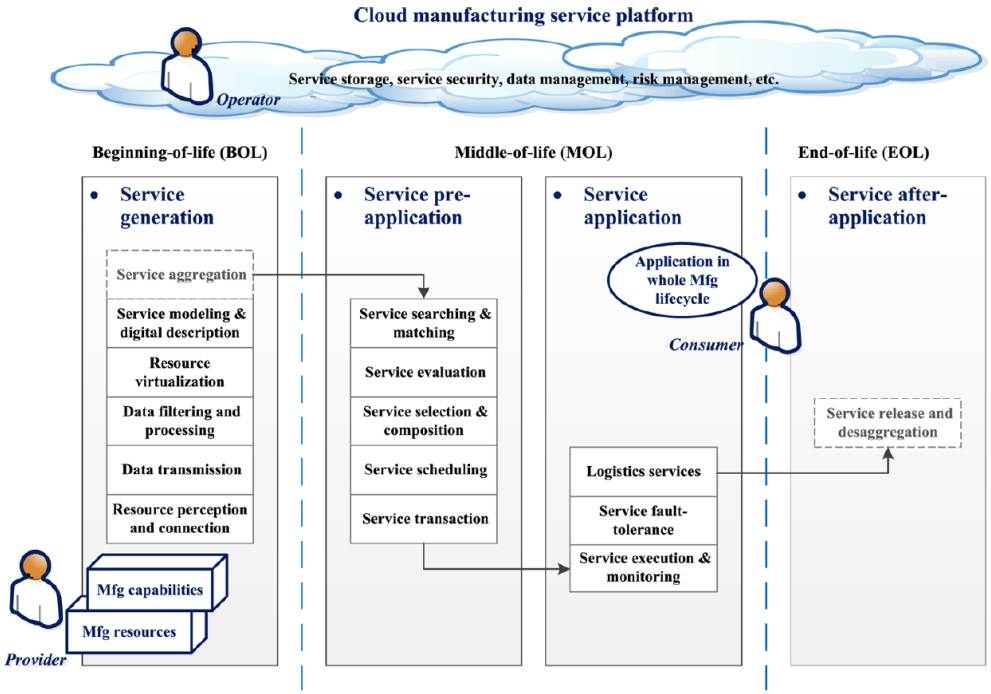

Abbildung 3.2: Workflow in Cloud Manufacturing nach Tao [33]

Service-generation

Ressourcenwahrnehmung und -bindung (Resource perception and connection): Die Ressourcenwahrnehmung wird hier mit agentenbasierten Technologien zur verteilten Steuerung und Kommunikation in virtuelle Ressourcen umgewandelt. Dabei können die eigentlich passiven Maschinen als aktive Teilnehmer in der Umgebung agieren. Mittels

Sensoren können die Maschinen diese Echtzeitdaten an die Umgebung signalisieren und so ihren Zustand mitteilen [33].

Datenübertragung (Data transmission): Zunächst müssen die Daten in ein einheitliches Format beziehungsweise Sprache transformiert werden. Dabei werden die notwendigen Anfragen von Kunden ein Datenformat konvertiert welches vom System ausgewertet werden kann. Lu schlägt hierfür eine ontologische Sprache wie RDF (Resource Description Framework) vor um eine höhere Interpretierbarkeit der Maschinen zu erreichen [33].

Datenfilterung und -verarbeitung (Data filtering and processing): Dieser Schritt parst und analysiert die Kundenangaben und legt diese im System ab. Dabei wird eine Serviceanfrage erzeugt, welche für das Fertigungsprojekt alle Spezifikationen wie Produkteigenschaften, Kostenerwartungen sowie Anforderungen an die Logistik beinhaltet. In diesem Schritt erfolgt auch die Validierung beziehungsweise die Analyse, ob das Produkt hergestellt werden kann [33].

Lu et al. diskutieren für die Filterung und Verarbeitung der Daten ein mögliches Vorgehen [35]. Damit wird ein Produkt, welches vom Kunden angefordert wird auf die Merkmale runter gebrochen und kann anhand der Möglichkeiten der Fertigungsressourcen verknüpft werden.

Abbildung 3.3 zeigt die hierarchische Darstellung eines Kundenprojektes. Dabei wird die Kundenanfrage an vordefinierte Merkmale überführt. Das Projekt enthält Angaben wie den Eigentümer, der die Anfrage gestellt hat und Produktinformationen wie Menge und das Material aus dem das Produkt bestehen soll. Die Ebene Produkt besteht aus den einzelnen Artikeln. Diese können selbst Baugruppen sein und aus mehreren Artikeln bestehen. Dabei kann jeder Artikel des Produktes auf vordefinierte Features runter gebrochen werden [35].

Ressourcenvirtualisierung: Wie bereits beschrieben werden hier die Fertigungsressourcen sowie -fähigkeiten virtualisiert. Dieser Schritt ist sehr wichtig um eine genaue Beschreibung sowie Spezifikation der Ressource zu erhalten und diese für das CM zur Verfügung zu stellen. So müssen die Eigenschaften einer Ressource sehr feingranular beschrieben werden und auch mögliche auftretende Probleme aufgeführt werden. Das sind auch die Unterscheidungsmerkmale zwischen den einzelnen Teilnehmern im CM-Netzwerk. Denn anhand dieser Beschreibungen können dann die zerlegten Aufgaben eindeutiger zu den infrage kommenden Fertigungsressourcen zugeordnet werden [35].

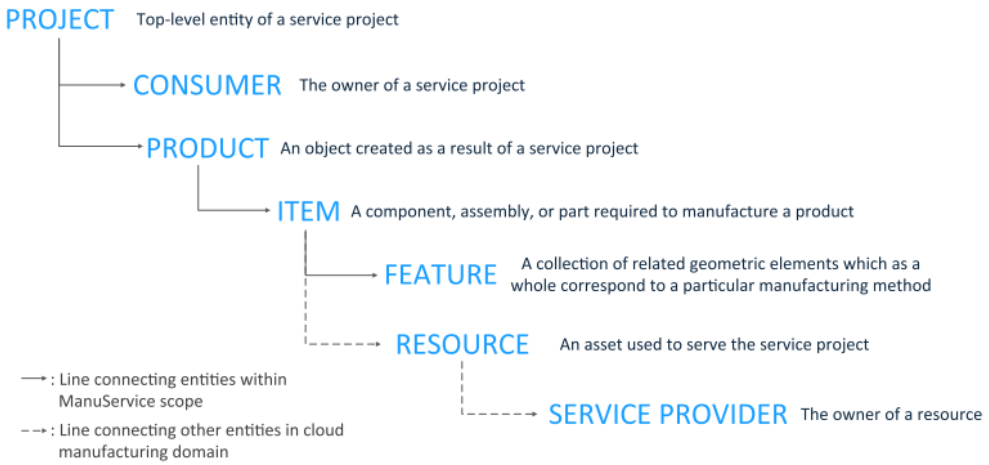

Abbildung 3.3: Hierarchische Produktspezifikationen eines Projektes nach Lu et al. [35]

Servicemodellierung und digitale Beschreibung (Service Modeling and Digital Description): Auf der Modellierung der Services und der digitalen Beschreibung basiert die Realisierung der Produktfertigung. Hier werden die Leistungen beschrieben und beinhalten Informationen wie Logistik oder Wirtschaftlichkeit. Damit werden die verschiedenen virtuellen Ressourcen zur Fertigung zur Verfügung gestellt [35].

Service pre-application

Servicesuche und -matching (Service Searching and Matching): Bei der Servicesuche und dem -matching wird für die Serviceanforderung der optimale Dienst gesucht. Dabei wird für jede Teilaufgabe der beste Dienst ausgewählt, um die Serviceanfrage zu erfüllen. Dabei muss die Suche und das Matching mit unterschiedlichen Kundenanforderungen umgehen können. Das betrifft insbesondere auch die Qualität der Serviceanfragen selbst. Für die Anfragen muss ein unterschiedlicher Stand von Wissen, den Fähigkeiten sowie Möglichkeiten vorausgesetzt werden [33]. Die Gesamtaufgabe wird dann schrittweise in Teilaufgaben zerlegt, bis eine geeignete Fertigungsressource zugeordnet werden kann. Das kann auf Grundlage von Abgleichen von gespeicherten historischen Aufgaben und der dazugehörenden Matchings erfolgen. Liegt nach der Zerlegung noch kein geeignetes Ergebnis vor, kann eine Ausschreibung erstellt werden. Diese wird abgelegt und für zukünftige Servicesuchen genutzt [34] [36].

Service Evaluation: In diesem Schritt wird der Quality-of-Service (QoS) evaluiert. Das soll maßgeblich auf dem Feedback von Kunden basieren die eine erledigte Serviceanfrage und -bearbeitung bewerten. Dieser Schritt soll dabei eine Garantie für Kunden liefern und drückt dabei die Leistung, die Verfügbarkeit, die Sicherheit sowie Zuverlässigkeit aus [33].

Serviceauswahl und -zusammensetzung (Service Selection and Composition): Für das Mapping stehen drei Verfahren zur Auswahl:

- *One-to-one:* Dieses Mapping besteht aus einem einfachen Prozess, um Übereinstimmungen zu finden. Nach dem Vergleich erfolgt die Zuordnung von einer Serviceanforderung zu einer Fertigungsdienstleistung. In der Regel wird dabei die semantische Distanz verwendet um die Ähnlichkeit der Serviceanforderung und den verfügbaren Dienstleistungen zu messen.
- *Many-to-one:* Dabei werden mehrere Ressourcen kombiniert um eine (zusammengesetzte) Ressourcenform zu erstellen. Das können auch verschiedene Dienstleister sein, die eine komplexe Aufgabe für den Kunden erfüllen. Die Zuordnung der Teilaufgaben wird dann durch einen Optimierungsalgorithmus vorgenommen der die komplexen Serviceanfragen und -anforderungen optimal zu ordnet. Die Nutzung von verschiedenen Ressourcen ist für den Kunden nicht sichtbar.
- *One-to-many:* Bei diesem Verfahren erscheint dem Kunden seine Anfrage als einzigartig. Die Fertigungsressource wird allerdings von mehreren Kunden geteilt. Diese Tatsache ist für den Kunden nicht sichtbar.

Da die Ressourcennutzung und -auslastung sehr dynamisch ist, muss die Serviceauswahl sich an den tatsächlichen Kapazitäten orientieren. Damit muss die Verfügbarkeit der Ressource also die bereits angesprochene Ressourcenfähigkeit bei der Zusammensetzung des Services berücksichtigt werden.

Serviceplanung (Service Scheduling): Die Planung der verschiedenen (Teil-)Aufgaben wird als wesentlicher Bestandteil der CM-Umgebung angesehen [33]. Zusätzlich kommt beim CM bezüglich CC noch die Komponente der Logistik dazu, die einer komplexeren Planung der einzelnen Herstellungsschritte bedarf [37].

Service Transaktionen: Das Ziel beim CM ist es, einen geeigneten Service zu ermitteln, der eine Kundenaufgabe mit Rücksichtnahme auf die Rahmenbedingungen des Kunden zu einer hohen Qualität und niedrigem Preis bietet. Hierfür müssen die Zusammenhänge

zwischen dem Informations-, dem Logistik- und dem Zahlungsfluss koordiniert werden [33].

3.3 Architekturmodelle

CM ist ein serviceorientiertes Modell, welches physikalisch gesehen die verteilten Ressourcen und Fähigkeiten der Teilnehmer über einen zentralisierten Cloud-Service zu Verfügung stellt. Somit sind alle Teilnehmer in der Lage diese Ressourcen und Fähigkeiten über diesen Service abzurufen und zu nutzen. Damit Kunden im CM die Cloud Services nutzen können, müssen diese gekapselt und zentral verwaltet werden. Für diesen Einsatz wird eine Plattform benötigt die eine entsprechende Servicesuche, eine intelligente Zuordnung und Ausführung durchführt [38].

Eines der am häufigsten zitierten Ansätze für so eine Plattform ist die von Xu [4]. Abbildung 3.4 zeigt eine Architektur nach Xu das vier Schichten aufweist und drei Domänen zugeordnet werden kann.

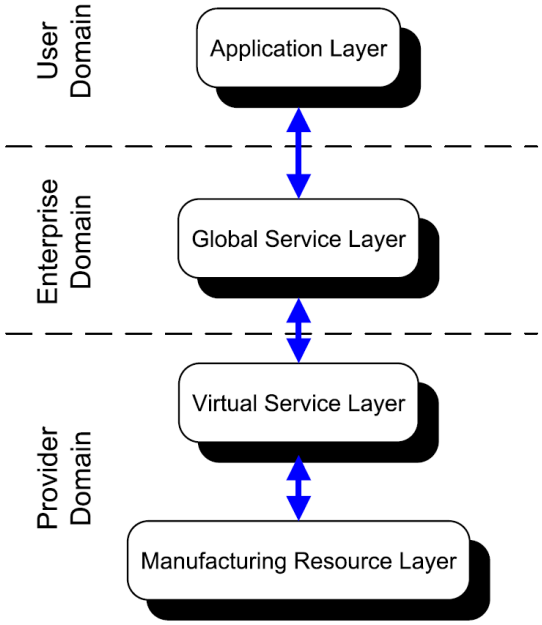

Abbildung 3.4: Architektur mit vier Schichten für das CM Paradigma nach Xu [4]

Die *Provider Domain* besteht aus den *Manufacturing Resource Layer* sowie dem *Virtual Service Layer*. Die *Enterprise Domain* besteht aus dem *Global Service Layer* und die *User Domain* aus dem *Application Layer*. Nachfolgend werden die einzelnen Layer (Schichten) vorgestellt und weitere Ansätze anderer Autoren diskutiert.

Manufacturing Resource Layer (Schicht für die Fertigungsressourcen): Diese Schicht umfasst die Ressourcen, die während des Lebenszyklus der Produktentwicklung benötigt werden. Diese Fertigungsressourcen werden in physische Fertigungsressourcen und die Fertigungskapazitäten unterteilt. Die physischen Fertigungsressourcen umfassen Informationen, Personal und Software als auch Hardwareressourcen und das Material. Die Fertigungskapazitäten sind die immateriellen Ressourcen des produzierenden Unternehmens. Sie stellen die Fähigkeit, wie beispielsweise das Produktdesign oder die Produktion dar, diese Aufgaben zu übernehmen und abzuarbeiten.

Zhang et al. stellen ein Modell für die Beschreibung von Fähigkeiten als Ontologien vor [39]. Ressourcen werden dabei in Fertigungsressourcen sowie Fertigungsfähigkeiten unterteilt. Diese werden weiter klassifiziert in beispielsweise Soft-Ressourcen, wie Software oder Wissen. Über ein Regelwerk werden die Fähigkeiten katalogisiert und das Ontologie-Modell beschreibt diese semantisch um Übereinstimmungen zu finden.

Zhao führt den Ansatz für die Fertigungsressourcen fort und teilt diese in die dynamische Fähigkeit zur Produktion sowie die statischen Fähigkeiten und Funktionen der Maschine. Dabei sollen diese Fähigkeiten als Ontologien in der Web Ontology Language (OWL) beschrieben werden [40].

Abbildung 3.5 stellt die Fertigungsressourcen nach Zhao dar. Die Fähigkeit der Produktion bezieht sich auf die Maschinen und beschreibt den Anlagenzustand (Equipment state), die Aufgabenbelastung (Task load), den Produktionszyklus (Production cycle), die Prozessplanung (Process schedule) und die Prozessqualität (Process quality). Die Fähigkeiten der Funktionen der Maschinen werden in die Genauigkeitsfähigkeit (Accuracy capability), die Formfähigkeit (Shape capability), die Handwerkstypfähigkeit (Craft type capability), die Prozessgrößenfähigkeit (Process size capability) und die Werkstücktypfähigkeit (Workpiece type capability) unterteilt [40].

Virtual Service Layer (Virtualisierungsschicht): Die Virtualisierungsschicht soll die Fertigungsressourcen identifizieren, virtualisieren und als Service für das CM zur Verfügung stellen [4]. Wie von Lu et al. vorgeschlagen, können mehrere Technologien an-

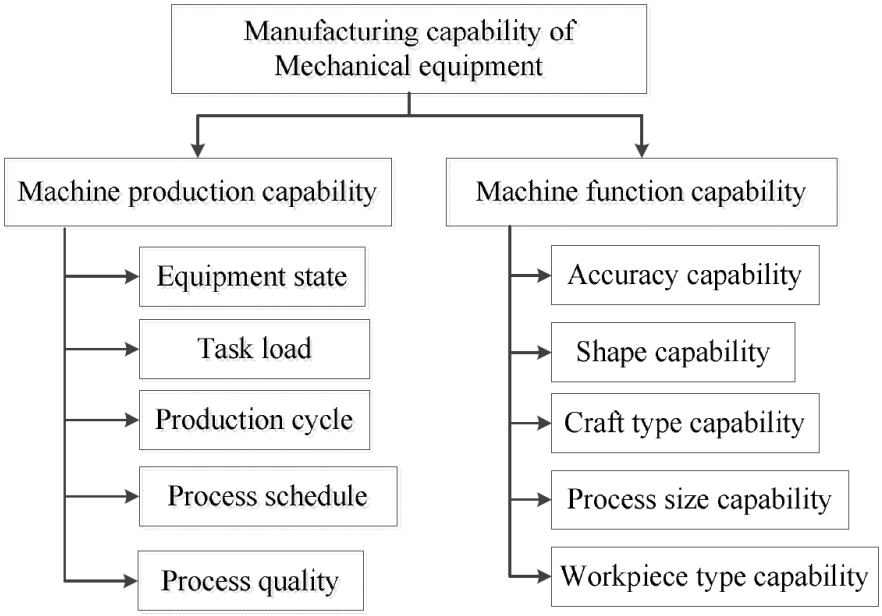

Abbildung 3.5: Einteilung der Fertigungsressourcen nach Zhao [40]

gewendet werden, die für die Identifikation der Fertigungsressourcen eingesetzt werden können. So kann die Identifikation über RFID oder GPS erfolgen [19].

Global Service Schicht: Auf dieser Schicht werden die Bereitstellungstechnologien wie beispielsweise PaaS verortet [4]. Dieses Layer ist verantwortlich für den Betrieb der Aktivitäten der Plattform und beinhaltet die Dienste die dafür notwendig sind. Das können beispielsweise die Zuordnung der Aufgaben zu den Fertigungsressourcen und deren Überwachung sein.

Application Layer (Anwendungsschicht): Die Anwendungsschicht ist die Schnittstelle der verschiedenen Anwender und den Ressourcen der CM Umgebung zur Fertigung. Diese Schicht stellt die Funktionen beispielsweise interaktiv mittels Terminals zur Verfügung [4].

Adamson erweitert dieses vier Schichtenmodell von Xu um drei weitere. Dabei stützt er sich auf einen Ansatz von Huang [17] [41]. Diese sieben Schichten bestehen aus der *Ressourcenschicht (Ressource layer)*, der *Wahrnehmungsschicht (Perception layer)*, der *Virtualisierungsschicht (Virtualisation layer)*, der *Cloud Service Schicht* – auch als *Core*

Middleware bezeichnet, der *Application layer (Anwendungsschicht)*, der *Schnittstellenschicht (Interface Layer)* sowie dem *Supporting Layer*.

Die Wahrnehmungsschicht kapselt dabei die Erfassung von physischen Fertigungsressourcen noch einmal in einer zusätzlichen Schicht ab. Es übernimmt auch die Anbindung der erfassten Ressourcen mit den dazugehörenden Informationen an das CM Netzwerk.

Die Cloud Service Schicht übernimmt die Verwaltung der Systeme, Ressourcen und Aufgaben. Weiterhin ist sie verantwortlich für die Planung sowie Überwachung und den Abrechnungen.

Die Schnittstellenschicht kapselt die Anwendungsschicht von der Interaktion mit den Anwendern ab. Diese stellt einen Service zur manuellen Interaktion wie das Durchsuchen der verfügbaren Dienste zur Verfügung.

Die Supporting Schicht kann auch als Wissensschicht bezeichnet werden, da hier die notwendigen Informationen gekapselt sind, die beispielsweise das Prozesswissen betreffen. Weiterhin werden hier Strategien für die Sicherheit von CM-Systemen und die Kommunikationsumgebung verortet [41].

Helo et al. beschreiben die Einschränkung aktueller IT-Lösungen in den Bereichen verteilte Fertigung in einer Lieferkette und schlagen einen Cloud-basierten Ansatz vor [42]. Die zentralisierte Softwareanwendung, basiert auf einer mehrschichtigen Architektur und bedarf laut Helo et al. keiner aufwendigen IT-Integration der Teilnehmer.

Der User Interface Layer bietet für die Anwender eine flexible webbasierte Oberfläche. Diese kann über ein Dashboard an die Bedürfnisse der Anwender angepasst werden. Der Business Logic Layer empfängt die Anfragen der Anwender und verarbeitet diese. Diese Daten werden in ein Format konvertiert aus denen die Informationen über das Produkt ausgelesen werden können. Der Daten Layer sammelt die Daten aus den unterschiedlichen Quellen und legt diese in einer Cloud Infrastruktur ab. Der Andon Monitor dient der Überwachung der Produktion und kann einen Echtzeit Status der Maschinen ausgeben. Der ERP-Konnektor fungiert im System als Informationsgateway. Dabei werden die Aufträge aus dem ERP als Prozessanweisungen an das Steuerungsautomationssystem der Fertigungsanlage gesendet [42].

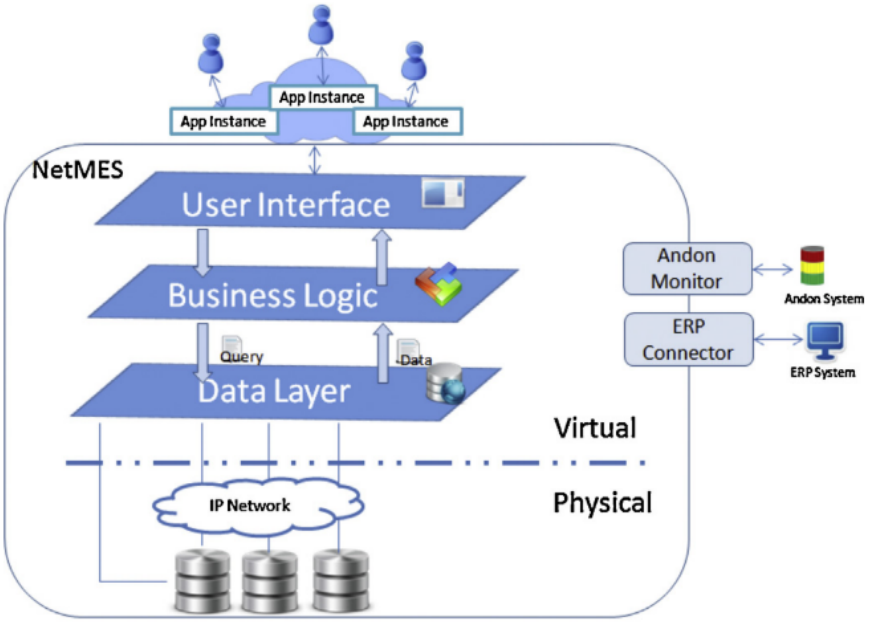

Abbildung 3.6: Cloud-basierte Plattform für CM – NetMES nach Helo et al. [42]

3.4 Kollaboratives Cloud Manufacturing

CM ermöglicht durch die distributive Natur der verteilten Fertigungsressourcen eine Kollaboration der verschiedenen Teilnehmer. So können die Teilnehmer auf Ressourcen und Kapazitäten zurückgreifen ohne das tiefgreifende Investitionen notwendig sind. Diese Agilität der Produktion erhöht allerdings drastisch den Koordinationsbedarf zwischen den Teilnehmern [13].

Adamson et al. stellen mehrere Szenarios vor, die erst durch das CM möglich werden. So können bereits in der Designphase eines Produktes mehrere Fertigungsunternehmen mitwirken, um für den Kunden diesen Vorgang mehr dynamisch zu gestalten und zu individualisieren. Weiterhin ist der Herstellungsvorgang wesentlich dynamischer, da Fertigungsaufgaben für das Produkt verteilt werden können. Je nach beispielsweise Verfügbarkeit oder Kosten können die Fertigungsressourcen ausgewählt werden. Eine Kollaboration bei der Fertigung kann aus einer globalen Sicht nachhaltig sein, da so Ressourcen effektiv genutzt werden können. Abbildung 3.7 zeigt das Konzept für eine verteilte und dynamische Planung in einer kollaborativen CM Umgebung.

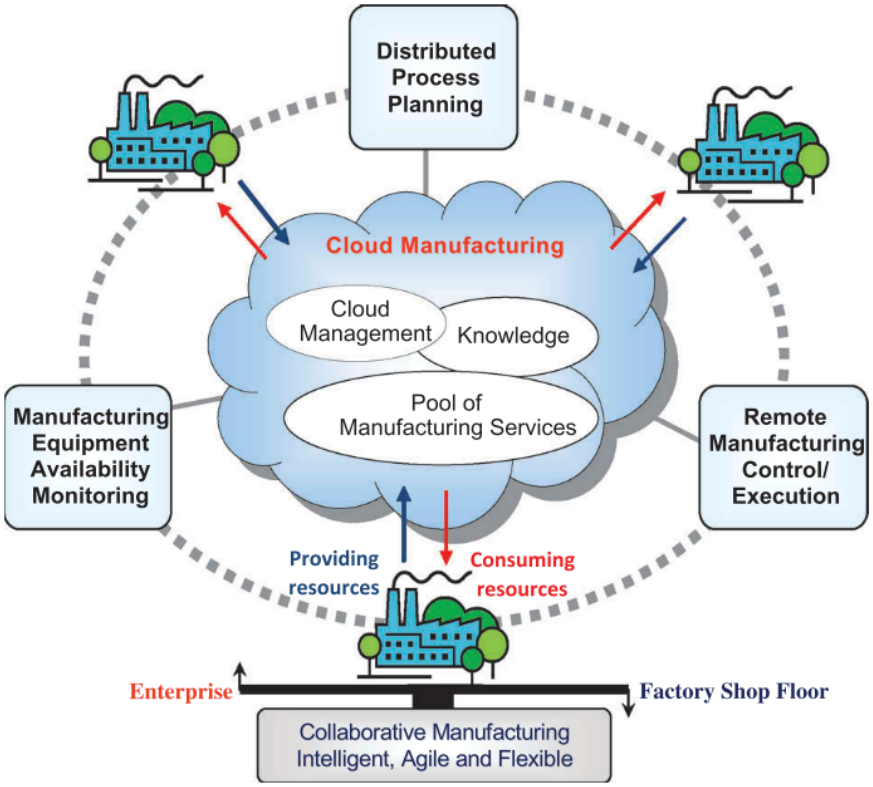

Abbildung 3.7: Kollaboratives Cloud Manufacturing nach Adamson et al. [17]

Dadurch das die Fertigung von Produkten beziehungsweise Komponenten an unterschiedlichen Standorten und in verschiedenen Unternehmen erfolgen kann, müssen auch die Information zu dem Produkt verteilt werden. Dieser Informationsaustausch erfordert weiterhin eine konsistente Interpretation der Produktinformationen. Dabei spielen aber auch Faktoren wie Vertrauen und ein einheitliches Datenformat eine große Rolle.

Einen Ansatz für eine kollaborative CM-Serviceplattform liefern Valilai und Houshmand [43]. Diese Plattform verfolgt einen serviceorientierten Ansatz und basiert auf der Grundlage des CC-Paradigmas. Die Zusammenarbeit zwischen mehreren global verteilten Fertigungsunternehmen soll erleichtert werden, indem standardisierte Formate wie STEP verwendet werden [43].

3.5 Datenschutz und -sicherheit im Cloud Manufacturing

Daten und Informationen von Unternehmen beinhalten in den meisten Fällen sensible Angaben über Unternehmenswissen und geistiges Eigentum. Der Schutz dieser Daten ist eines der Themengebiete die derzeit am häufigsten im Bereich des CM diskutiert werden [44]. Wu et al. betont die Notwendigkeit von Vertrauensmodellen zwischen den einzelnen Teilnehmern des CM Systems und führt die Sicherheit als kritischen Aspekt für den Erfolg von CM an [45]. Auch Gao et al. stellen ein vertrauenswürdiges Netzwerk zwischen den Konsumenten eines Service und Dienstleistern vor [46]. Die Systemsicherheit wird dabei ebenfalls als kritischer Faktor für die erfolgreiche Nutzung eines CM-Systems vorgestellt.

Einen innovativen Ansatz, um Produktdesigndateien zu schützen, stellen Cai et al. vor [47]. Es wird eine partielle Verschlüsselung vorgeschlagen, um ein CAD-Modell in verschiedenen Granularitäten für verschiedene Benutzer mit unterschiedlichen Zugriffsrechten darzustellen. CAD-Modelle können aus mehreren Features (Merkmalen) bestehen, die sich auf bestimmte Bereiche im Modell beziehen und Eigenschaften und Informationen beinhalten. So können die Features eines CAD-Modells flexibel verschlüsselt werden, wobei der Zusammenhang der Features zueinander gemäß den Anforderungen an das Gesamtmodell bestehen bleibt. Dies ermöglicht eine Zusammenarbeit mehrerer Teilnehmer an einem CAD-Modell, ohne dass das gesamte Modell offenbart wird.

Christidis und Devetsikiotis stellen die Anwendung der Blockchain für das IoT vor [26]. Die Blockchain soll dabei als verteiltes Peer-to-Peer-Netzwerk dienen, ohne dass ein vertrauenswürdiger Vermittler zwischen den einzelnen Teilnehmern benötigt wird. Teilnehmer die sich nicht kennen und sich nicht vertrauen können auf diese transparente und für jeden Teilnehmer nachvollziehbare Weise miteinander interagieren.

3.6 Blockchain Technologie im Kontext von Industrie 4.0

Christidis und Devetsikiotis schlagen die Blockchain Technologie und Smart Contracts für die gemeinsame Nutzung von Services und Ressourcen vor [26]. Dabei soll der Einsatz von Geräten, die mit der Blockchain mittels Smart Contracts interagieren, vereinfacht, automatisiert und nachvollziehbar gestaltet werden. So kann beispielsweise die Verteilung von Updates für Geräte über eine Blockchain abgewickelt werden. Die Geräte

können mittels eines Smart Contracts über ein Update benachrichtigt werden. Für den Nutzer des Gerätes kann so eine transparente Kommunikation des Gerätes zum Hersteller dokumentiert werden. Ebenfalls kann die Verteilung des Updates vereinfacht werden, da das Update auch von Geräten abgerufen werden kann, die es bereits heruntergeladen haben.

Li et al. stellen für den Transfer von Daten ein Konzept mit einer Blockchain vor [48]. Dieser Ansatz verfolgt die Idee, dass die Daten für jeden Teilnehmer zur Verfügung gestellt werden.

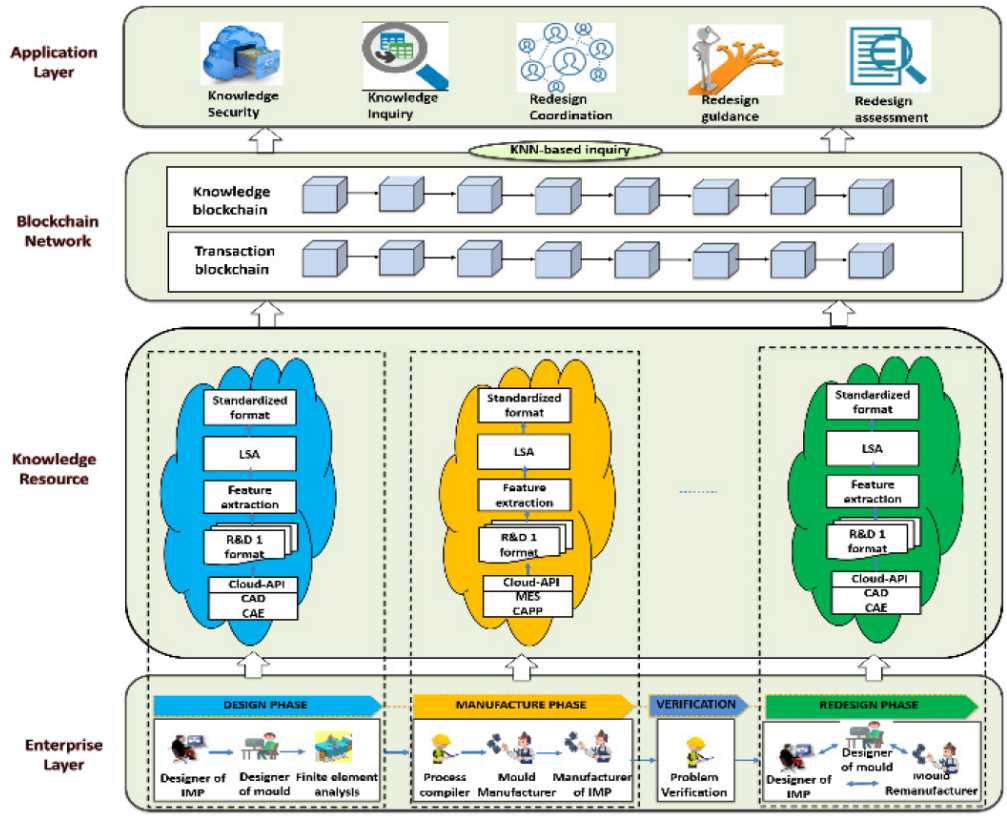

Abbildung 3.8: Knowledge-sharing Plattform nach Li et al. [48]

Abbildung 3.8 zeigt die Knowledge-sharing Plattform nach Li et al.. Bei diesem Konzept werden alle Daten in einer Blockchain abgelegt und gespeichert, die als Wissensdatenbank abgerufen werden kann. Über das *Enterprise Layer* übermitteln die Teilnehmer Ihre Daten an eine private Cloud-Umgebung. Hier werden die Daten in ein standardisiertes Format transformiert. Eine private Blockchain wird eingesetzt, die aus der Cloud-

Umgebung als *Knowledge-Resource*, die Daten für die weitere Verarbeitung empfängt. In der *Application Layer* können die Teilnehmer des Netzwerkes auf die aufgearbeiteten Daten zugreifen.

Angrish et al. stellen mit FabRec einen dezentralen Ansatz vor um Informationen für die Fertigung zwischen verschiedenen Unternehmen auszutauschen [49]. Dafür wird die Blockchain Technologie mit Smart Contracts verwendet. Damit sollen einerseits die Fähigkeiten der Ressourcen den Teilnehmern mitgeteilt werden und andererseits sollen die Transaktionen transparent und dauerhaft aufgezeichnet werden.

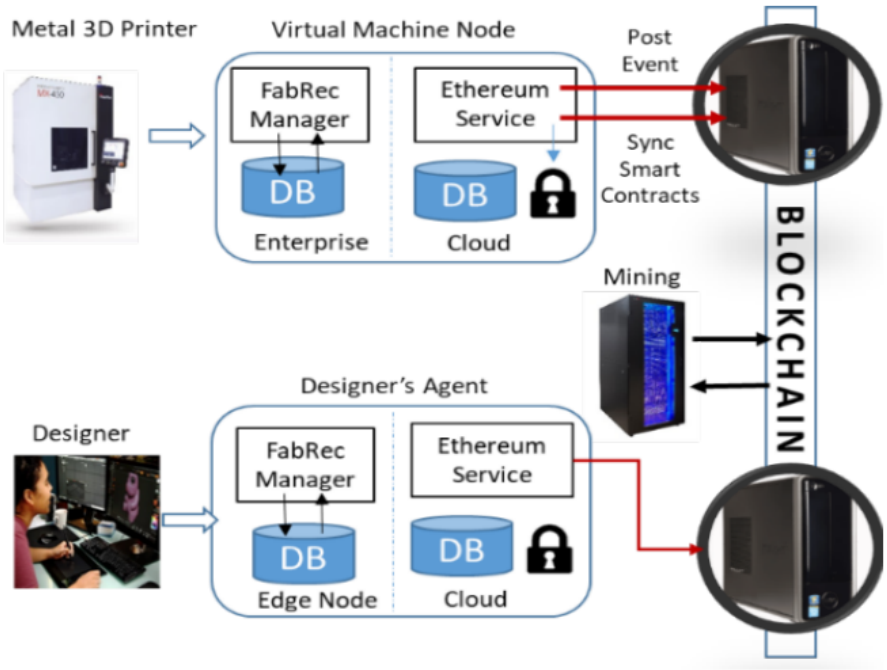

Abbildung 3.9: Austausch von Daten in FabRec nach Angrish et al. [49]

Abbildung 3.9 zeigt den Austausch in FabRec mittels der Blockchain. Das System besteht aus vier Softwarekomponenten in jeweils einem Zuständigkeitsbereich. Der Virtual Machine Node beinhaltet die Komponente des digitalen Zwillings der Maschine. Es ermöglicht die Maschinenkommunikation zu der Fertigungsmaschine. Weiterhin beinhaltet es den Ethereum Client der die Interaktion mit der Blockchain ermöglicht. Der Designer's Agent ermöglicht die Interaktion des Designers mit dem Ethereum Client der wiederum mit der Blockchain interagieren kann. Der Prozess des Mining soll die Authentizität der Ereignisse überprüfen.

Einen weiteren Ansatz liefern Bahga et al. mit Blockchain platform for industrial internet of things (BPIIoT) [50]. Dabei werden die Kernaspekte *consumer-to-machine* (Mensch-zu-Maschine) und *machine-to-machine* (Maschine-zu-Maschine) Transaktionen hervorgehoben, die mittels des BPIIoT Systems ermöglicht werden. Weiterhin sollen so mittels Single-Board Computer auch ältere Fertigungseinrichtungen in das Fertigungsnetzwerk eingebunden werden können. Über Smart Contracts können dann Aufträge aber auch Wartungs- und Diagnoseaufgaben automatisiert werden.

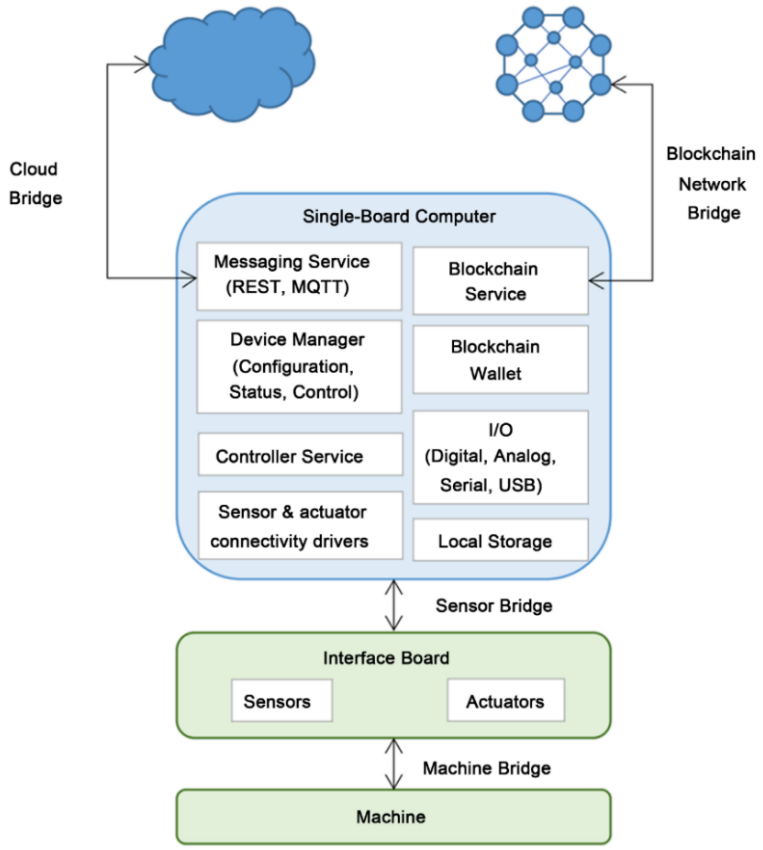

Abbildung 3.10: Blockchain platform for industrial internet of things (BPIIoT) nach Bahga et al. [50]

Abbildung 3.10 visualisiert den Ansatz. Hier werden über die Cloud Daten über den Betrieb der Maschine ausgetauscht. So können beispielsweise die Sensoren der Maschine beziehungsweise des Single-Board Computers ausgelesen werden. Mittels der Blockchain und Smart Contracts erfolgt die Kommunikation bezüglich der Aufträge. Bahga et al.

werfen allerdings die Frage des Datenschutzes auf, da Daten auf der Blockchain abgelegt werden und somit von jedem Teilnehmer eingesehen werden können.

3.7 Fazit

Im Bereich CM existieren bereits Konzepte und Lösungsansätze für eine Plattform, die Zuordnungen von Aufgaben zu den Ressourcen ermöglichen. Die meisten Ansätze setzen dabei auf eine zentrale und agentenbasierte Lösung. Als offene Punkte und zukünftige Arbeiten wird bei diesen Ansätzen die Frage des Datenschutzes und der -sicherheit angegeben.

Weiterhin wurden für das IoT und die Industrie 4.0, Lösungsvorschläge recherchiert, die den Einsatz der Blockchain Technologie vorstellen. Diese können verschiedene spezifische Probleme adressieren. Durch die Natur der Blockchain, dass Daten unveränderlich und unwiderruflich aufgezeichnet und über das Netzwerk repliziert werden, kann so ein Herkunftsnachweis am (digitalen) Eigentum erreicht werden. Weiterhin existiert kein *Single Point of Failure*, was den Ausfall des Gesamtsystems meint wenn eine einzelne Komponente ausfällt, da die Knoten verteilt über alle Teilnehmer identisch sind.

Im folgenden Kapitel sollen daher vier Konzepte analysiert werden. Dabei handelt es sich um zentrale Lösungsansätze sowie dezentrale die mit der Blockchain-Technologie umgesetzt worden sind.

Kapitel 4

Analyse

Wie in Kapitel 3 gezeigt wurde, gibt es bereits unterschiedliche Lösungsansätze und Modelle für eine CM-Plattform. Nachfolgend werden diese erläutert und die Kernelemente der jeweiligen Konzepte herausgearbeitet. Eine SWOT-Analyse soll dann die Schwächen und Stärken als interne Faktoren sowie Chancen und Risiken als externe Faktoren übersichtlich darstellen. Die Abkürzung SWOT steht für Strengths (Stärken), Weaknesses (Schwächen), Opportunities (Chancen) und Threats (Risiken).

4.1 NetMES

Helo et al. stellen einen Cloud-basierten Prototypen für das CM vor [42]. Die NetMES *(manufacturing execution systems)* Plattform ist eine webbasierte Lösung, die für die Kontrolle und Steuerung von Maschinen entwickelt wurde. Die Integration soll dabei über eine Verbindung zu den ERP Systemen der Unternehmen erfolgen. Über das System können die Ressourcen in Echtzeit verfolgt werden, auch über Unternehmensgrenzen hinweg. So kann die Überwachung der Produktion sowohl für den Kunden als auch für den Lieferanten transparent gestaltet werden. Bezüglich der fehlenden Kontrolle über die Speicherung der Daten in der Cloud-Umgebung geben die Autoren an, dass die Datensicherung und der Datenschutz noch ungeklärte Faktoren sind. Tabelle 4.1 zeigt das Ergebnis der SWOT Analyse für die NetMES Plattform.

Innensicht (Internal Factors)	
Stärken (Strengths) (+)	Schwächen (Weaknesses) (-)
▪ Transparenz für den Kunden hinsichtlich des Status seines Auftrags ▪ Zugangskontrolle mit Authentifizierung basierend auf Rollen	▪ Kollaboration wird auf Produktionsebene beschrieben, nicht für die globale Teilung und Planung von Kundenaufträgen und Aufgaben ▪ Single Point of Failure, wenn die zentrale Plattform ausfällt, ist das CM nicht möglich
Außensicht (External Factors)	
Chancen (Opportunities) (+)	Risiken (Threats) (-)
▪ Verschlüsselung der Daten kann Schutz für die Speicherung in der Cloud sicherstellen	▪ Fehlende Kontrolle über die Datenspeicherung in der Cloud ▪ Fehlendes Vertrauen der Benutzer in ein Produkt beziehungsweise eine Plattform

Tabelle 4.1: SWOT Analyse für NetMES nach Helo et al. [42]

4.2 CPMC

Liu et al. stellen einen Cloud-basierten Prototypen für die *Cyber-physical manufacturing cloud* (CPMC): *open-architecture of cloud-based manufacturing systems* vor [51]. Diese Plattform besteht aus vier Schichten (Ressource Layer, Ressource Virtualization Layer, Core Cloud Layer, Application Layer), die untereinander über Hypertext Transfer Protocol (HTTP) kommunizieren. Die Ressourcen für die Fertigung werden über Webservices verfügbar und bekannt gemacht werden. Die Plattform sieht allerdings keine Zerlegung und Planung von Aufgaben vor. Kunden, die einen Produktionsauftrag zuweisen wollen, wählen direkt einen Webservice aus, der dann die Herstellung übernimmt. Tabelle 4.2 zeigt die SWOT Analyse für die Plattform von Liu et al..

Innensicht (Internal Factors)	
Stärken (Strengths) (+)	Schwächen (Weaknesses) (-)
▪ Schnelles An- und Abmelden von Maschinen ▪ Dynamische Konfiguration der Fertigungsressourcen ▪ Kunden können den Status für ihren Auftrag einsehen	▪ Keine Zerlegung und Planung von Aufgaben ▪ Single Point of Failure, da es sich um eine zentrale Plattform handelt
Außensicht (External Factors)	
Chancen (Opportunities) (+)	Risiken (Threats) (-)
▪ Der Kunde kann auf die Ressourcen zugreifen, wenn die Ressourcen den Zugriff autorisieren ▪ Big Data Analysen der Fertigung um Ressourcen dynamisch zu konfigurieren	▪ Kunde und Hersteller müssen dem Cloud-Provider vertrauen ▪ Ein Konzept für Datenschutz wird nicht aufgezeigt

Tabelle 4.2: SWOT Analyse für CPMC [51]

4.3 FabRec

Angrish et al. stellen FabRec als eine Plattform vor, die einen Austausch von Informationen in einem vertrauenswürdigen Netzwerk ermöglicht [49]. Die Plattform basiert auf der Blockchain-Technologie, die transparent Informationen für die Fertigung für alle Teilnehmer zur Verfügung stellt. Mithilfe von Smart Contracts können Aufgaben automatisiert Ressourcen zugeordnet werden. Tabelle 4.3 zeigt die SWOT Analyse für FabRec. Die Autoren führen an, dass der Austausch der Produktionsdaten über die Blockchain für die Unternehmen einen Verlust von Kontrolle über diese Daten bedeutet. Weiterhin vertrauen Unternehmen zumeist nur in ihre eigenen Prozesse, da sie diese selbst verantworten und kontrollieren können. Als weiteren positiven Punkt geben die Autoren an, dass es

für Kunden wichtig ist, die Identität des Herstellers zu kennen. Dies wird erreicht, da bei den Produktionsressourcen die Provider angegeben und somit für die Kunden ersichtlich sind.

Innensicht (Internal Factors)	
Stärken (Strengths) (+)	Schwächen (Weaknesses) (-)
▪ Identität des Anbieters der Produktionsressource ist bekannt ▪ Fähigkeiten der Ressourcen sind allen Teilnehmern bekannt ▪ Keine zentrale Autorität für den Betrieb der Plattform	▪ Dauer für die Verteilung der Daten auf der Blockchain ▪ Keine Zerlegung und Planung von Aufgaben ▪ Aufgaben werden von Kunden direkt Fertigungsressourcen zugeordnet
Außensicht (External Factors)	
Chancen (Opportunities) (+)	Risiken (Threats) (-)
▪ Durch Dezentralisierung entstandene Transparenz für jeden Teilnehmer ▪ Ein Register Smart Contract reguliert die Teilnahme am Netzwerk	▪ Unternehmen vertrauen nur auf eigene Prozesse ▪ Austausch von Produktionsdaten ist sensibel

Tabelle 4.3: SWOT Analyse für FabRec [49]

4.4 BIIot

Bahga et al. stellen ebenfalls eine Plattform, Blockchain Platform for IoT (BIIoT), vor, die auf der Blockchain-Technologie basiert [50]. Die Ressourcen werden den Teilnehmern dabei über einen Cloud-basierten Ansatz bekannt gegeben. Sensible Daten, wie Kundenaufträge, werden über die Blockchain verteilt und die Teilnehmer können darauf zugreifen. Über Smart Contracts sollen die Aufträge automatisiert zugeteilt werden

können und Zahlungen bei Lieferung der Produkte erfolgen. Wenn ein Kunde einen Auftrag über die Blockchain vergibt, erfolgt keine Zerlegung der Aufträge. Der Kunde sendet einen Auftrag beziehungsweise eine Aufgabe direkt an eine Maschine. Wenn ein Auftrag von einer Maschine nicht komplett erledigt werden kann, werden die restlichen Aufgaben mittels der Maschine-zu-Maschine-Kommunikation an andere Maschinen übermittelt. Tabelle 4.4 zeigt die SWOT Analyse von BIIoT. Die Autoren diskutieren die möglichen Bedenken der Teilnehmer der Plattform, unter anderem betrifft das den Datenschutz. Da die Daten auf der Blockchain für jeden Teilnehmer einsehbar sind, kann auch so potenziell auf jeden Eintrag zugegriffen werden.

Innensicht (Internal Factors)	
Stärken (Strengths) (+)	Schwächen (Weaknesses) (-)
▪ Auftragsverwaltung ist dezentral und vertrauenswürdig – ohne Kontrolle durch eine zentrale Autorität ▪ Transparenz des Herstellungsprozesses durch Produktionsaufzeichnungen auf der Blockchain	▪ Teilnehmer im Netzwerk können direkt auf die Fertigungsressourcen zugreifen ▪ Keine Zerlegung und Planung von Aufgaben durch die Plattform
Außensicht (External Factors)	
Chancen (Opportunities) (+)	Risiken (Threats) (-)
▪ Bewertungssystem für Hersteller beziehungsweise Ressourcen können Kunden bei der Auswahl helfen ▪ Smart Contracts können Kunden bei der Auswahl der passenden Produktionsressource helfen	▪ Datenschutzbedenken der Teilnehmer, da die Daten der Blockchain von jedem Teilnehmer eingesehen werden kann ▪ Informationen über Maschinen in einer Cloud-Umgebung

Tabelle 4.4: SWOT Analyse für BPIIoT [50]

4.5 Bewertungskriterien

Die oben aufgeführten Plattformen verfügen über unterschiedliche Funktionsumfänge. Diese sollen nun in ihre Bestandteile aufgeteilt, gruppiert und vergleichbar gemacht werden. Mithilfe von Harvey Balls als Ideogramme sollen die resultierenden qualitativen Daten anschaulich dargestellt werden. Die Bewertungsskala ist dabei für alle Eigenschaften gleich gewichtet. Diese ist eine vierstufige Ordinalskala bei der die Elemente *Nicht erfüllt*, *Teilweise erfüllt*, *Erfüllt* und *Unbekannt* verwendet werden. Nachfolgend werden die Bewertungskriterien aufgelistet und erläutert. Weiterhin wird aufgeführt, wie die Eigenschaften der Skala zugeordnet werden. Anschließend erfolgt die Bewertung der Konzepte anhand der Kriterien.

Datenschutz
Gerade im IoT ist die Privatsphäre und somit der Datenschutz ein oft diskutiertes Thema [52]. Im Bereich der Kollaboration müssen Daten ausgetauscht und somit auch preisgegeben werden. Dabei handelt es sich oft um sensible Daten, die auch Firmengeheimnisse sein können. Ein unachtsamer Umgang mit solchen Daten kann wirtschaftliche Folgen für ein Unternehmen nach sich ziehen [53]. Deshalb ist es wichtig, dass Teilnehmer in einer kollaborativen Umgebung anonym auftreten können und nur notwendige Daten teilen [52]. Somit beschreibt diese Eigenschaft, ob die Anforderungen an den Datenschutz, insbesondere für sensible Daten, erfüllt werden. So sollen beispielsweise nur befugte Teilnehmer auf die Daten zugreifen können.

Die Bewertung *Nicht erfüllt* wird vergeben, wenn zwar der Datenschutz beschrieben ist, aber keine Mechanismen für den Datenschutz ermittelt werden konnten oder diese potenziell ungeeignet sind. *Teilweise erfüllt* im Bereich Datenschutz bedeutet, dass Mechanismen beschrieben werden, aber auch gleichzeitig Limitierungen ersichtlich sind. *Erfüllt* wird als Bewertung vergeben, wenn der Datenschutz sicher gestellt ist und mit dem Konzept die Teilnehmer im Netzwerk potenziell anonym auftreten können. Sind keine Mechanismen für den Datenschutz angegeben und weiterhin der Punkt nicht erläutert wurde, wird es mit *Unbekannt* markiert.

Datensicherheit
In einer Cloud-Umgebung ist die Sicherstellung der Datensicherheit weitaus komplizierter als in einem traditionellen Informationssystem [54]. Der Eigentümer der Daten die

in eine Cloud-Umgebung hochgeladen werden, hat nur eine eingeschränkte Kontrolle darüber, wo die Daten abgelegt werden. Auch werden die Daten in dieser Umgebung auf mehreren System abgelegt. Weiterhin geht mit der Datensicherheit die Datenintegrität einher. Das bedeutet, dass nur autorisierte Nutzer die Daten löschen oder modifizieren können. Ein weiterer Punkt der Datensicherheit ist die Vertraulichkeit der Daten und somit auch der Schutz des geistigen Eigentums [55]. Indem sensible Daten in Cloud-Umgebungen hochgeladen werden, können Angreifer diese Daten abgreifen und das geistige Eigentum für sich beanspruchen. Diese Eigenschaft beleuchtet, wie die Daten weiterverwendet werden. Damit soll insbesondere der Schutz des geistigen Eigentums adressiert werden.

Als *Nicht erfüllt* zählt, dass die Datensicherheit in keiner Weise umgesetzt und es als ungelöstes Problem aufgezählt wird. Die Bewertung *Teilweise erfüllt* wird vergeben, wenn die Datensicherheit von den Autoren als Risikofaktor beschrieben wird und mögliche Konzepte beschrieben werden, die das Problem adressieren. *Erfüllt* bedeutet, dass Konzepte für die Datensicherheit existieren und die Datenintegrität sowie der Schutz des geistigen Eigentums sichergestellt werden kann. *Unbekannt* wird vergeben, wenn dieser Punkt im Konzept nicht besprochen wird.

Ausfallsicherheit

Beim SCM ist die Verfügbarkeit aller involvierten Teilnehmer und deren Ressourcen von großer Bedeutung [55]. Ausfälle die während des gesamten Lebenszyklus des Herstellungsprozesses auftreten können, beeinträchtigen die gesamte Lieferkette. In der Cloud-Umgebung kann die Verfügbarkeit von Fehlern im Netzwerk verursacht werden oder auch den Ausfall von Hardware oder eines ganzen Cloud-Dienstes [54]. Beim CM ist die Verfügbarkeit der Plattform essentiell für ein kollaboratives Herstellen und die Kommunikation zwischen den Teilnehmern [2] Für die Analyse gibt diese Eigenschaft an, ob das (Gesamt-)System verfügbar ist, wenn einzelne Komponenten ausfallen.

Insbesondere wenn das System über einen Single Point of Failure verfügt ohne ein Konzept um eine Ausfallsicherheit zu garantieren, wird die Bewertung *Nicht erfüllt* vergeben. *Teilweise erfüllt* wird vergeben, wenn Risiken aufgezeigt werden und potenzielle Mechanismen existieren, um die Verfügbarkeit von einzelnen Komponenten zu gewährleisten. Die Bewertung *Erfüllt* wird vergeben, wenn das gesamte System ausfallsicher ist und somit alle Funktionen verfügbar sind auch wenn einzelne Komponenten ausfallen.

Wenn die Ausfallsicherheit aus dem Konzept nicht ersichtlich ist, so wird *Unbekannt* vergeben.

Transparenz

Transparenz im CM bedeutet in erster Linie, dass der Status von Ressourcen für alle Teilnehmer eingesehen werden kann [21]. Das führt zu einer effizienteren Planung und Durchführung von Aufgaben. Weiterhin bedeutet Transparenz für den Kunden, Kosten für Aktivitäten, wie der Nutzung einer Ressource, einsehen zu können [4].

Ein System erhält die Bewertung *Nicht erfüllt*, wenn weder der Status und die Verfügbarkeit von Ressourcen eingesehen werden kann, als auch keine Transparenz für die Kosten ersichtlich ist. *Teilweise erfüllt* erhält ein System als Bewertung, wenn ein Konzept aufgezeigt wird, wie die Verfügbarkeit von Ressourcen auf der Plattform für die Teilnehmer ersichtlich ist. Weiterhin wird bewertet ob die Kosten für die Nutzung von Ressourcen transparent dargestellt werden. Die Bewertung *Erfüllt* erhält ein System, wenn die Verfügbarkeit von Ressourcen und die Kosten für die Nutzung eingesehen werden kann. Als *Unbekannt* zählt, wenn die Transparenz für die dargestellten Punkte nicht ermittelt werden konnte.

Kollaboration

Wie bereits in Kapitel 3.4 dargestellt, ist Kollaboration zwischen den Fertigungsunternehmen ein wichtiger Wettbewerbsvorteil. Die Möglichkeit einen Kundenauftrag auf mehrere kleinere Aufträge zu teilen und diese an unterschiedliche Produktionsressourcen kann einen Kosten- und Zeitvorteil für den Kunden als auch für die Fertigungsunternehmen sicherstellen [19]. Es wird analysiert, ob die einzelnen Teilaufgaben geplant und an die Ressourcen verteilt werden können.

Ist diese Möglichkeit nicht gegeben wird die Bewertung *Nicht erfüllt* vergeben. *Teilweise erfüllt* wird als Bewertung vergeben, wenn eine manuelle Interaktion für die Teilung und Planung der Aufgaben vorgenommen werden muss, oder die Zuordnung der Fertigungsressourcen nicht auf der Plattform erfolgt. *Erfüllt* bedeutet, dass der Kunde einen Kundenauftrag auf der Plattform veröffentlicht und dieser vollautomatisch zerteilt, geplant und an die entsprechenden Fertigungsressourcen zugeordnet wird. *Unbekannt* wird vergeben, wenn Kollaboration im Konzept zwar angesprochen werden, aber keine Mechanismen für diese aufgezeigt werden.

4.6 Vergleich

Tabelle 4.5 stellt die Plattformen mit den Bewertungen der Eigenschaften vor. Nachfolgend werden die Bewertungen für die einzelnen Plattformen und den jeweiligen Eigenschaften begründet.

	Datenschutz	Datensicherheit	Ausfallsicherheit	Transparenz	Kollaboration
NetMES	◑	○	◑	◑	◑
CPMC	–	○	◑	●	○
FabRec	○	●	●	●	○
BIIoT	○	◑	◑	●	◑

○ Nicht erfüllt ◑ Teilweise erfüllt ● Erfüllt – Unbekannt

Tabelle 4.5: Visualisierung der Ergebnisse mittels Harvey Balls

Auswertung für NetMES

Für NetMES wurde für das Kriterium Datenschutz die Berwertung *Teilweise erfüllt* vergeben. Das Risiko wird dargestellt und als zukünftige Arbeiten aufgeführt und potenzielle Mechanismen werden für den Datenschutz aufgeführt, allerdings ist kein Mechanismus anhand der Veröffentlichung ersichtlich. Das Kriterium Datensicherheit wird mit *Nicht erfüllt* bewertet. Die Daten werden in einer zentralen Datenbank abgelegt und somit hat zumindest der Provider der CM-Plattform den uneingeschränkten Zugang darauf. Das Bewertungskriterium Ausfallsicherheit wurde ebenfalls mit *Teilweise erfüllt* bewertet. Zwar kann in einer Cloud-Umgebung eine hohe Ausfallsicherheit für die Daten gewährleistet werden, allerdings kann das gesamte System ausfallen, beispielsweise wenn der Provider die CM-Plattform nicht mehr anbietet. Für die Transparenz wird die Bewertung *Teilweise erfüllt* vergeben. Der Status der Produktion ist vom Kunden einsehbar, sowie der Status der Maschinen für die produzierenden Unternehmen. Allerdings ist nicht ersichtlich wie die Auftragszuordnung zu den Produktionsressourcen erfolgt. Eine

Kollaboration wird auf Ebene der Produktion vorgestellt. Wie die CM-Plattform eine Planung von Aufgaben vornimmt, ist nicht ersichtlich. Deshalb wird für dieses Kriterium *Teilweise erfüllt* als Bewertung vergeben.

Auswertung für CPMC

Für CPMC wird für den Datenschutz die Bewertung *Unbekannt* vergeben, da kein Konzept erläutert wird, wie sensible Daten des Kunden geschützt werden können. Das Kriterium Datensicherheit wird mit *Nicht erfüllt* bewertet. Das Konzept erörtert zwar den Schutz der Ressource vor unautorisierten Zugriffen, allerdings nicht was den Datenschutz an sich anbelangt. Wie bei NetMES wird aus den selben Gründen für die Ausfallsicherheit die Bewertung *Teilweise erfüllt* vergeben. Für die Transparenz wird das Kriterium *Erfüllt* vergeben. Das Konzept sieht eine offene Architektur vor und die Beschreibung der Services kann von den Teilnehmern eingesehen werden. Weiterhin kann der Status der Kundenaufträge abgerufen werden. Das Kriterium für Kollaboration wird mit *Nicht erfüllt* bewertet. Aus dem Konzept wird ersichtlich, dass Kunden die Produktionsressourcen direkt auswählen können, ohne dass der Auftrag durch die Plattform zerteilt und für mehrere Produktionsressourcen geplant werden kann.

Auswertung für FabRec

Für FabRec wird für das Bewertungskriterium Datenschutz *Nicht erfüllt* vergeben. Die Autoren führen an, dass für die Transparenz die Identität der Teilnehmer bekannt sein muss. Allerdings wird kein Konzept erläutert, wie der Schutz dieser Identität gewährleistet werden kann. Das Kriterium Datensicherheit wird mit *Erfüllt* bewertet. Die Integrität der Daten die auf der Blockchain abgelegt sind, ist durch die Technologie selbst gewährleistet. Weiterhin wird ein Mechanismus erläutert über den sich Teilnehmer zunächst autorisieren müssen, um am Netzwerk teilnehmen zu können. Ebenso wird die Ausfallsicherheit mit *Erfüllt* bewertet. Das Gesamtsystem ist funktionsfähig, selbst wenn Teile ausfallen, da alle Daten auf der BLockchain als Kopien verteilt vorliegen. Angemerkt werden muss allerdings, dass das Register des Netzwerkes nicht weiter detailliert erläutert wird, wer die verwaltende Instanz ist. Somit ist ungeklärt wie neue Teilnehmer sich am Netzwerk anmelden können, wenn die Registrierung nicht mehr durchgeführt werden kann, beispielsweise wenn die Instanz ausfällt. Die Transparenz wird mit *Erfüllt* bewertet, da jeder Teilnehmer alle Daten einsehen kann. Für die Kollaboration wird die Bewertung *Nicht erfüllt* vergeben, da das Konzept keinen Mechanismus aufweisen kann,

das Aufgaben zerlegen und planen kann. Die Auswahl der Produktionsressourcen erfolgt direkt durch den Kunden und wird nicht von der Plattform abgewickelt.

Auswertung für BIIoT
Das Kriterium Datenschutz für BIIoT wird mit *Nicht erfüllt* bewertet. Einerseits werden Daten der Maschinen in eine Cloud-Umgebung abgelegt. Es wird nicht weiter aufgeführt, wer Zugriff darauf hat und ob es sich bei der Cloud-Umgebung um eine weitere Instanz handelt. Andererseits werden Informationen über Kundenaufträge und Fertigungsressourcen auf der Blockchain abgelegt und sind von jedem Teilnehmer einsehbar. Für die Datensicherheit wird die Bewertung *Teilweise erfüllt* erfüllt. Die Integrität der Daten, die über die Blockchain verteilt werden, kann durch die Technologie sichergestellt werden. Allerdings wird nicht weiter erläutert, wie die Datensicherheit für die Daten in der Cloud-Umgebung gewährleistet werden kann. Die Ausfallsicherheit wird mit *Teilweise erfüllt* bewertet, da auch hier nicht bekannt ist, wie sich das Gesamtsystem verhält, wenn die Cloud-Umgebung nicht verfügbar ist. Das Kriterium Transparenz wird mit *Erfüllt* bewertet. Es werden Mechanismen aufgezeigt, die sicherstellen, dass der Herstellungsprozess für die Teilnehmer eingesehen werden kann. Für das Bewertungskriterium Kollaboration wird *Teilweise erfüllt* als Bewertung vergeben. So wird dargestellt, dass die Fertigungsressourcen selbst Aufträge an andere Ressourcen weiter geben können. Eine Zerteilung oder Planung von der Plattform wird nicht angeboten und der Kunde muss die Fertigungsressourcen für seinen Auftrag selber auswählen.

4.7 Fazit aus der Analyse

In diesem Kapitel wurden vier Ansätze analysiert. Dabei wurde eine SWOT-Analyse der Ansätze durchgeführt und auf Basis der Veröffentlichungen wurden die Stärken, Schwächen, Chancen sowie Risiken im Kontext der zu konzeptionierenden Plattform extrahiert und aufgelistet. Es wurden zwei Ansätze die einen zentralen Ansatz für eine CM-Plattform verfolgen ausgewählt. Weiterhin wurden zwei Ansätze mit einen dezentralen Ansatz vorgestellt.

Der Punkt Datenschutz wird bei allen Konzepten als Punkt für weitere Arbeiten aufgeführt. Lediglich NetMES stellt einen Mechanismus vor der den Schutz der Daten gewährleisten kann. Die Datensicherheit kann mittels der Analyse über die Blockchain

erfolgen, da die Integrität der Daten bereits durch die Technologie selbst gewährleistet werden kann. Für die Ausfallsicherheit konnte die Analyse ebenfalls den Vorteil der Blockchain aufzeigen. Dadurch, dass alle Daten auf allen Knoten bei den Teilnehmern repliziert wird, hat das System in diesem Punkt keinen Single Point of Failure. Alle Konzepte verfolgen die Möglichkeit den Teilnehmern eine transparente Zusammenarbeit zwischen Kunden und Providern zu gewährleisten. In diesem Bereich bietet die Blockchain ebenfalls den Vorteil, dass sämtliche Transaktionen für jeden Teilnehmer sichtbar sind. Der Punkt der Kollaboration beim Produktionsprozess wird lediglich von zwei Ansätzen verfolgt. Dieser bietet, wie bereits im Kapitel 3.4 dargestellt, für das CM ein hohes Potenzial. Eine Orchestration der als Services angebotenen Produktionsressourcen kann nicht von den Plattformen erfolgen, da eine Teilung der Aufgaben und deren Planung beziehungsweise Zuordnung zu den Ressourcen vom Kunden vorgenommen werden muss.

Kapitel 5

Konzept

Dieses Kapitel stellt das Konzept für die CM-Plattform vor. Zunächst werden die Anforderungen an die Plattform vorgestellt. Anschließend wird die Auswahl der Technologie für die Plattform erläutert. Weiterhin wird die Systemumgebung, sowie die für die CM-Plattform relevanten Funktionen besprochen. Abschließend werden die Smart Contracts erläutert, die benötigt werden.

5.1 Anforderungen

Aus den Kapiteln 3 und 4, können funktionale und nicht-funktionale Anforderungen an eine Plattform für das CM abgeleitet werden.

Abbildung 5.1 zeigt ein Anwendungsfalldiagramm für das Systemverhalten aus der Anwendersicht für die CM-Plattform. Das Diagramm dient als Entwurf für die Anforderungen an die Plattform, die nachfolgend aufgelistet werden:

- Kunden müssen Aufträge auf die Plattform einstellen können
- Aufträge müssen über Meta-Informationen beschrieben werden können
- Kunden müssen Aufträge Ressourcen zuweisen können
- Anhänge wie CAD-Dateien sollen auf einen Dateisystem abgelegt werden können

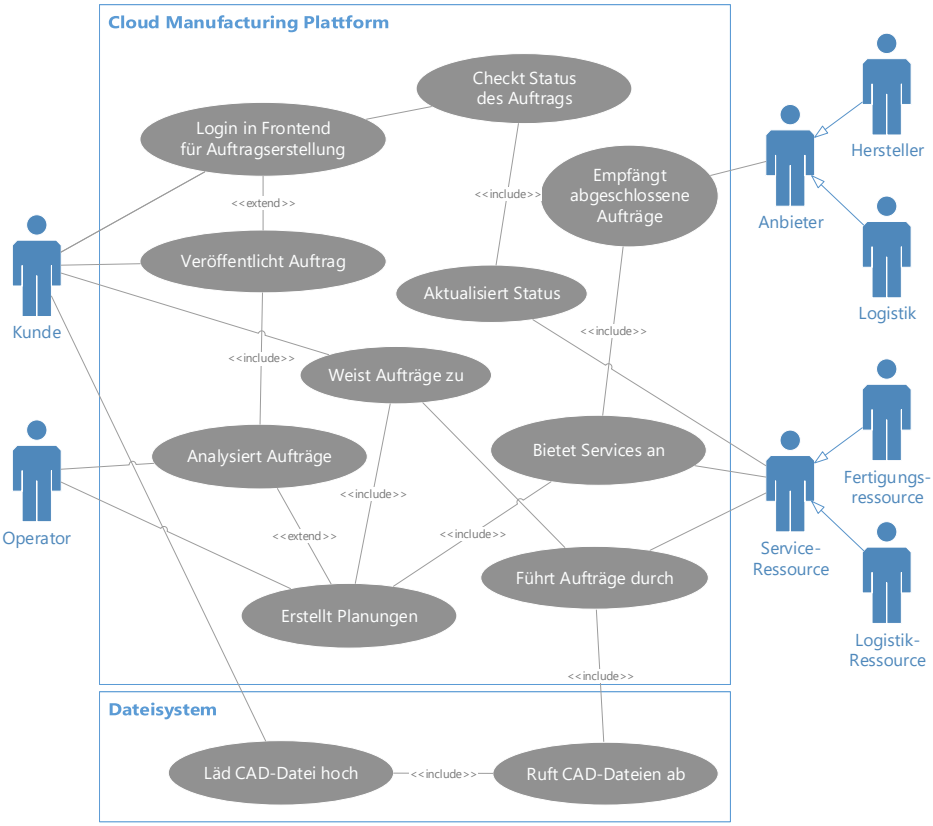

Abbildung 5.1: Anwendungsfalldiagramm CM-Plattform. Eigene Darstellung

- Die CAD-Dateien sollen nur autorisierten Teilnehmern zur Verfügung gestellt werden können
- Kunden müssen den Status ihrer Aufträge einsehen können
- Für Fertigungsressourcen müssen die Fähigkeiten als Services bereitgestellt werden können
- Für Logistik-Ressourcen müssen die Fähigkeiten als Services bereitgestellt werden können
- Die Kapazitäten und der Status der Fertigungsressourcen müssen bekannt sein
- Die als Service bereitgestellten Fertigungsressourcen und Logistik-Dienstleistungen müssen analysiert werden können

- Die Fertigungsressourcen müssen Aufträge entgegen nehmen können
- Die Fertigungsressourcen sollen Anhänge wie CAD-Dateien abrufen können
- Die Fertigungsressourcen müssen mit einen Hersteller als Service-Anbieter verknüpft werden können
- Die Logistik-Ressourcen müssen mit der Logistik als Service-Anbieter verknüpft werden können
- Die Service-Anbieter sollen nach Abschluss eines Auftrags Informationen über diesen erhalten können
- Operatoren müssen Analysen und Planungen der Kundenaufträge vornehmen können
- Operatoren sollen Kunden eine Planung ihrer Aufträge vorschlagen können
- Die Plattform soll für jeden Teilnehmer transparent sein
- Daten sollen transparent und nachvollziehbar übermittelt werden können
- Sensible Daten sollen nicht von jeden Teilnehmer eingesehen werden können

Weiterhin ergeben sich die folgenden nicht-funktionalen-Anforderungen:

- Aufträge sollen schnell Ressourcen zugeordnet werden können
- Aufträge sollen fehlerfrei zugeordnet werden können

5.2 Auswahl der Technologie für die Plattform

Im Kapitel 4.6 wurden vier verschiedene Ansätze vorgestellt und verglichen. Zwei basieren auf einer zentralen und zwei auf einer dezentralen Architektur mit der Blockchain Technologie. Nachfolgend werden die Kriterien aus der Analyse aufgeführt und die Entscheidungsfindung für das weitere Vorgehen erläutert.

Datenschutz

Der Datenschutz spielt für viele Unternehmen eine große Rolle im Wettbewerb [8]. Demnach wollen die Teilnehmer auf einer CM-Plattform möglichst anonym auftreten. Mit dem Einsatz einer Blockchain kann die Anonymität für die Teilnehmer gewährleistet werden. Lu und Xu schlagen für den Datenschutz vor, dass lediglich ein Minimum an Daten zu den Providern übermittelt werden sollen [56] Reyna et al. beschreiben Mechanismen wie Teilnehmer anonym im IoT Transaktionen vornehmen können und schlagen dafür die Verwendung einer Blockchain vor [57].

Datensicherheit

Die Datensicherheit ist ein zentraler Punkt dieser Arbeit. Lu und Xu führen an, dass nur autorisierten Teilnehmern komplette Angaben zum Produkt mitgeteilt werden sollen [56]. Aus der Analyse ist ersichtlich, wie Architekturen mit dem Einsatz der Blockchain eine hohe Datensicherheit gewährleisten können. Der Einsatz einer Blockchain kann die Integrität der Daten sicherstellen. Werden die Daten auf einen zentralen System hochgeladen, so haben die Teilnehmer keine Kontrolle mehr über diese Daten. Auch können kryptografische Verfahren nur autorisierten Teilnehmern den Zugriff auf bestimmte Daten ermöglichen.

Ausfallsicherheit

Da die Blockchain dezentral ist und somit alle Daten auf allen Knoten repliziert werden, ist so eine Ausfallsicherheit gewährleistet. Das Gesamtsystem funktioniert, selbst wenn große Teile der teilnehmenden Knoten ausfallen. Ein Ausfall einer zentralen Plattform für das CM würde auch den Ausfall des Gesamtsystems bedeuten.

Transparenz

Der Einsatz der Blockchain bietet eine hohe Transparenz, da alle Transaktionen dauerhaft und für jeden Teilnehmer einsehbar abgelegt werden. Eine Fertigungsressource kann auch den Status der Bearbeitung eines Auftrags auf der Blockchain veröffentlichen und somit den Kunden über den Fortschritt informieren. Weiterhin können die Services die über die Blockchain angeboten werden, von jeden Teilnehmer eingesehen werden. Es ist für jeden Teilnehmer nachvollziehbar wie die Services aufgebaut sind und welche Daten diese benötigen und verarbeiten.

Kollaboration
Um eine Kollaboration zu ermöglichen, müssen Kundenaufträge gegebenenfalls geteilt, geplant und den verschiedenen Fertigungsressourcen zugewiesen werden. Auf einer zentralen Plattform für das CM ist dafür der Operator verantwortlich. Planungen der Aufgaben für die Fertigungsressourcen erfolgen zentral um diese als Services zu orchestrieren [21] In einer dezentralen Architektur ist ein Operator nicht notwendig für den Betrieb einer CM-Plattform. Allerdings können Operatoren als Teilnehmer auftreten und einen Service für die Planung anbieten.

Somit bietet sich für alle vorgestellten Kriterien der Einsatz einer Blockchain an. Die Teilnehmer können miteinander interagieren, ohne die ihre Identität zu offenbaren und das auf eine transparente Weise. Weiterhin kann mit kryptografischen Verfahren der Datenschutz sowie die Datensicherheit ermöglicht werden.

5.3 Systemumgebung

Die Systemumgebung beinhaltet die Gruppen Stakeholder sowie Komponenten. Abbildung 5.2 zeigt die Systemumgebung mit den verschiedenen Aktoren und Komponenten und den Relationen zu einander. Nachfolgend sollen diese erläutert werden.

5.3.1 Stakeholder

Kunde
Kunden erstellen Aufträge für die Fertigung. Diese Aufträge bestehen aus:

- Meta-Informationen, die das Produkt beschreiben und weiteren Informationen über beispielsweise Kosten und Lieferung
- CAD-Dateien, die das Produkt repräsentieren

Ist ein Auftrag als Smart Contract auf der Blockchain veröffentlicht worden, können Kunden den Status des Auftrages einsehen. Wird eine Verschlüsselung der CAD-Datei vom Kunden gewünscht, so kann zusätzlich ein Smart Contract veröffentlicht werden, der gewährleisten soll, dass die Datei nur von autorisierten Fertigungsressourcen abgerufen werden kann. Dieser beinhaltet den Zugriff auf die CAD-Datei, die auf einen Dateisystem abgelegt wird. Nach der vollständigen Bearbeitung und Auslieferung des Auftrages sowie

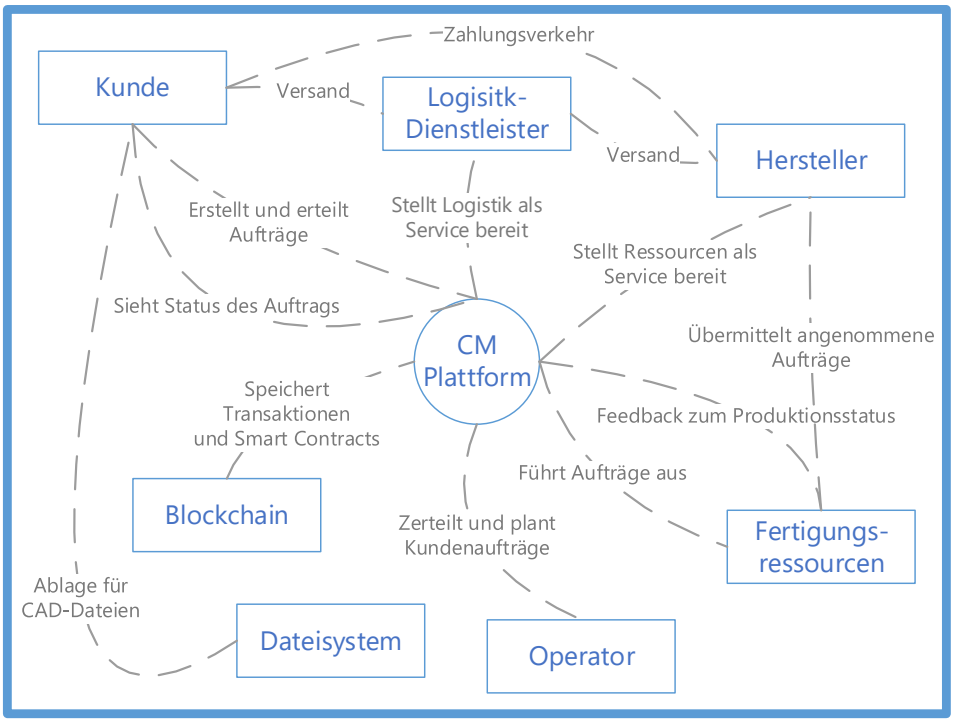

Abbildung 5.2: Kontextdiagramm mit Stakeholder und Komponenten. Eigene Darstellung

der Zahlung, wird der Smart Contract der den Kundenauftrag repräsentierte beendet. Das beendet ebenfalls einen vorhandenen und dazugehörenden Smart Contract, der für den Zugriff auf die CAD-Datei benötigt wurde.

Hersteller

Hersteller erstellen Informationen über die Fertigungsressourcen und deren Kapazitäten. Diese werden dann als Service auf der CM-Plattform zur Verfügung gestellt. Der Hersteller erhält Benachrichtigungen über Kundenaufträge, wenn seine Ressourcen einen Auftrag zugewiesen bekommen haben. Werden mehrere seiner Fertigungsressource für ein einzelnes Kundenprojekt genutzt, erhält er gebündelt diese Informationen. Hierfür übermitteln die Ressourcen an ein Smart Contract des Herstellers alle notwendigen Informationen. Diese können durch das ERP-System des Herstellers weiterverarbeitet werden, um eine Rechnung für die Nutzung der in Anspruch genommenen Ressourcen zu erstellen.

Operator
Operatoren agieren im Netzwerk als Dienstleister, die eine Zerlegung und Planung von Kundenaufträgen durchführen. Hierfür veröffentlichen sie Smart Contracts die diese Logik beinhalten. Bei der Zerlegung werden die Meta-Informationen des Auftrages verwendet. Diese Teilung der Aufträge erfolgt unter Berücksichtigung der im Netzwerk verfügbaren Informationen über die Fertigungsressourcen. Weiterhin können die derzeitigen Auslastungen der Fertigungsressourcen bei der Planung berücksichtigt werden. Ist die Planung auf mehrere räumlich verteilte Fertigungsressourcen aufgeteilt worden, werden zusätzlich für die Transporte zwischen den Produktionsstätten Smart Contracts der Logistik berücksichtigt. Hersteller oder Kunden können ebenfalls als Operatoren im Netzwerk auftreten.

5.3.2 Komponenten

Fertigungsressourcen
Die Ressourcen sind selbst Knotenpunkte im Netzwerk und können über einen eigenen Account angesprochen werden. Sie erhalten Aufträge, die gegebenenfalls Teilaufträge sein können. Diese werden abgearbeitet und das Produkt an den Kunden oder zur weiteren Verarbeitung versendet. Werden mehrere Ressourcen eines einzelnen Herstellers verwendet, so werden mittels eines Smart Contract alle Informationen gebündelt zu diesem Auftrag an den Hersteller mitgeteilt. Während der Bearbeitung bzw. nach Abschluss können die Ressourcen den Status eines Auftrags ändern. Der Status wird als Transaktion in der Blockchain angehängt.

Blockchain
Auf der Blockchain werden die Aufträge als Smart Contracts veröffentlicht. Die Meta-Informationen eines Auftrages werden als Transaktionen angehängt. Zusätzliche Dateien wie CAD-Pläne werden auf ein Dateisystem abgelegt und mit einen Hash-Wert referenziert. Weiterhin werden die angebotenen Services, wie beispielsweise die Fertigungsressourcen, als Smart Contracts auf der Blockchain veröffentlicht. Nach Erfüllung des Auftrags und sobald das Endprodukt beim Kunden beziehungsweise Auftraggeber eingetroffen ist, kann auch die Zahlung über die Blockchain erfolgen.

Dateisystem
Das Dateisystem dient als Ablage für große CAD-Dateien, die Pläne für die Konstruktion beinhalten. Diese werden durch einen Hash-Wert referenziert und können von autorisierten Benutzern abgerufen werden.

5.3.3 Systeminteraktion

Die Stakeholder interagieren mit der CM-Plattform, indem sie Services wie Fertigungsressourcen, Logistik-Dienstleistungen, sowie Planungen von Kundenaufträgen als Smart Contracts auf der Blockchain veröffentlichen. Weiterhin werden ebenfalls Kundenaufträge als Smart Contracts auf der Blockchain veröffentlicht. Nachfolgend soll anhand der Fertigungsressourcen und der Auftragserteilung dargestellt werden, wie die Systeminteraktion erfolgt. Die jeweiligen Smart Contracts werden näher im nachfolgenden Kapitel erläutert.

Bereitstellung der Ressourcen
Der Hersteller erstellt für seine Ressourcen jeweils einen Smart Contract. Jeder Smart Contract eines solchen Typs wird als *Ressourcen Smart Contract* bezeichnet. Er beinhaltet die Fähigkeiten der Fertigungsressource, welche Materialien bearbeitet werden können und sonstige Spezifikationen, die für eine Planung und Zuordnung für einen Auftrag betrachtet werden können. Weiterhin werden die Kapazitäten und die Kosten für die Beanspruchung der Ressource hinterlegt. Diese Smart Contracts werden auf der Blockchain veröffentlicht und sind so lange aktiv bis sie vom Hersteller beendet werden. Wenn eine Änderung vorgenommen werden muss, wird der jeweilige Smart Contract beendet und ein neuer Smart Contract mit den aktualisierten Informationen wird veröffentlicht.

Auftragserteilung
Der Kundenauftrag wird ebenfalls als Smart Contract veröffentlicht. Dieser enthält alle relevanten Informationen als Meta-Informationen, um eine Zuordnung zu einen Service zu ermöglichen. Die CAD-Datei wird auf einen Dateisystem abgelegt auf das jeder Teilnehmer Zugriff hat, kann allerdings nur von autorisierten Teilnehmern des Netzwerks abgerufen werden.

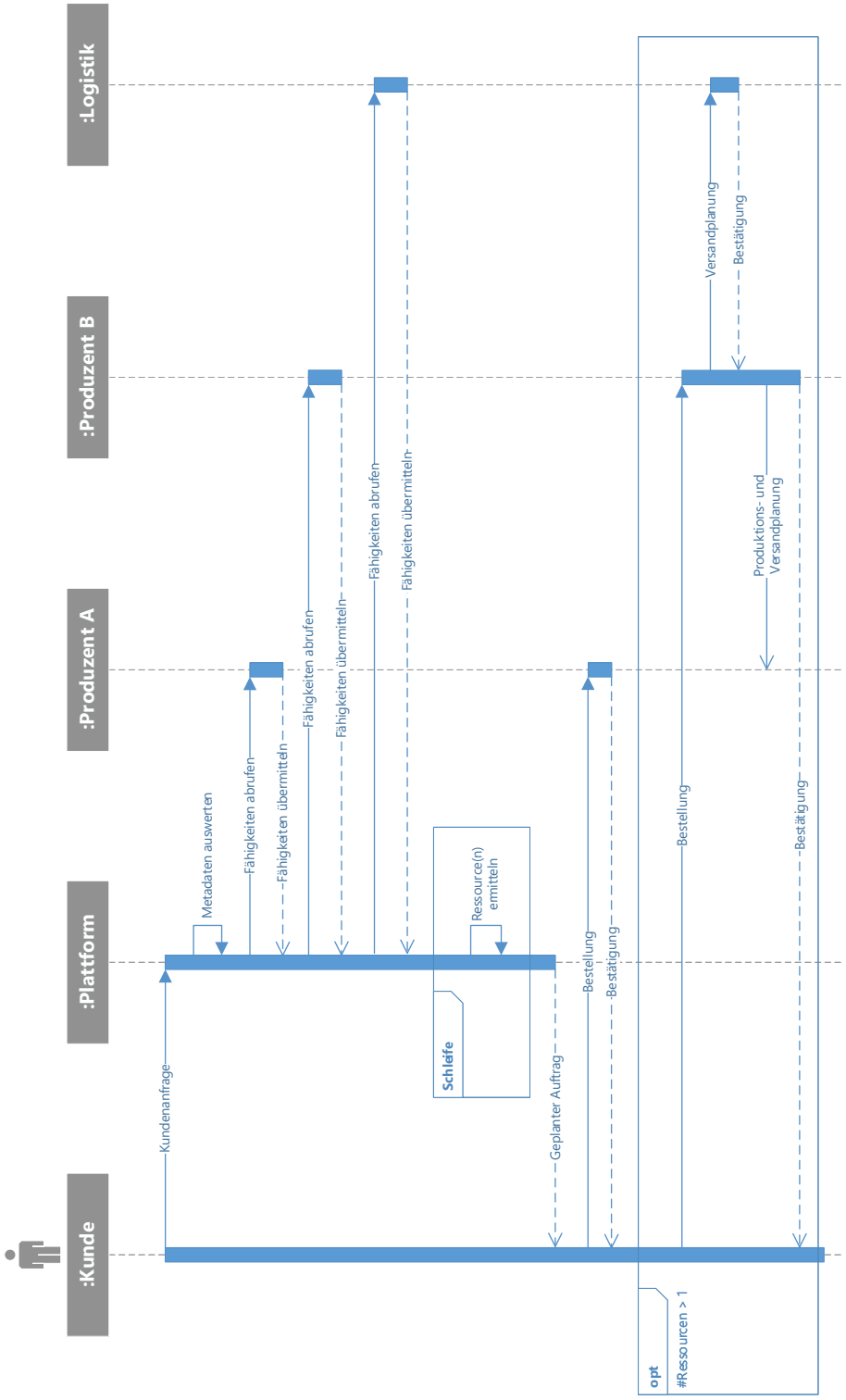

Abbildung 5.3: Sequenzdiagramm – Auftragserteilung. Eigene Darstellung

Abbildung 5.3 zeigt schematisch den Ablauf der Auftragserteilung. Die Zuordnung eines Auftrags zu den Fertigungsressourcen wird von Smart Contracts vorgenommen, die für eine Zerteilung und Planung bereitgestellt werden. Diese werden von den Herstellern selbst oder Operatoren erstellt und veröffentlicht. Die Teilung und Planung erfolgt iterativ, dementsprechend wird ein Kundenauftrag soweit zerteilt bis eine Zuordnung von dem (Teil-) Auftrag zu einer Ressource gefunden wurde.

Die Auslastung der Fertigungsressourcen kann direkt über den Status abgefragt werden und kann in der Planung mitberücksichtigt werden. Ebenfalls kann die derzeitige Auslastung direkt über die aufgezeichneten Transaktionen ermittelt werden, da jede Auftragserteilung an die Fertigungsressourcen auf der Blockchain gespeichert wird.

Wird mehr als eine Ressource ermittelt, beziehungsweise die Ressourcen räumlich verteilt sind, wird ein Logistik-Dienstleister in die Planung miteinbezogen. Hierfür können Logistiker ebenfalls als Akteure im Netzwerk teilnehmen. Die Dienstleistung der Logistik wird dann auch als Service über Smart Contracts auf der CM-Plattform zur Verfügung gestellt. Die Planung der Fertigung muss dann diesen Zwischenschritt ebenfalls mitberücksichtigen. Ebenso der damit verbundene Zusammenbau der Teilprodukte zu einen Fertigprodukt und die anschließende Auslieferung an den Kunden.

5.4 Schichtendarstellung

Für eine CM-Plattform, die mit einer Blockchain realisiert werden soll, können spezifische Funktionen definiert werden. In Anlehnung an Xu und Adamson werden die verschiedenen Funktionen auf Schichten verteilt [4] [17]. Den Schichten können weiterhin Smart Contracts zugeordnet werden, die auf der Blockchain veröffentlicht werden. Die Smart Contracts mit ihren jeweiligen Datenmodellen werden im nächsten Kapitel vorgestellt. Abbildung 5.4 visualisiert diese Schichtendarstellung. Nachfolgend werden die verschiedenen Schichten mit ihren Funktionen erläutert.

Ressourcenschicht
Diese Schicht bezieht sich auf die physischen Ressourcen der Hersteller und Logistik-Dienstleister die als Provider im Netzwerk auftreten. Die jeweiligen Ressourcen haben bestimmte Fähigkeiten und Kapazitäten die von den Providern beschrieben werden müssen.

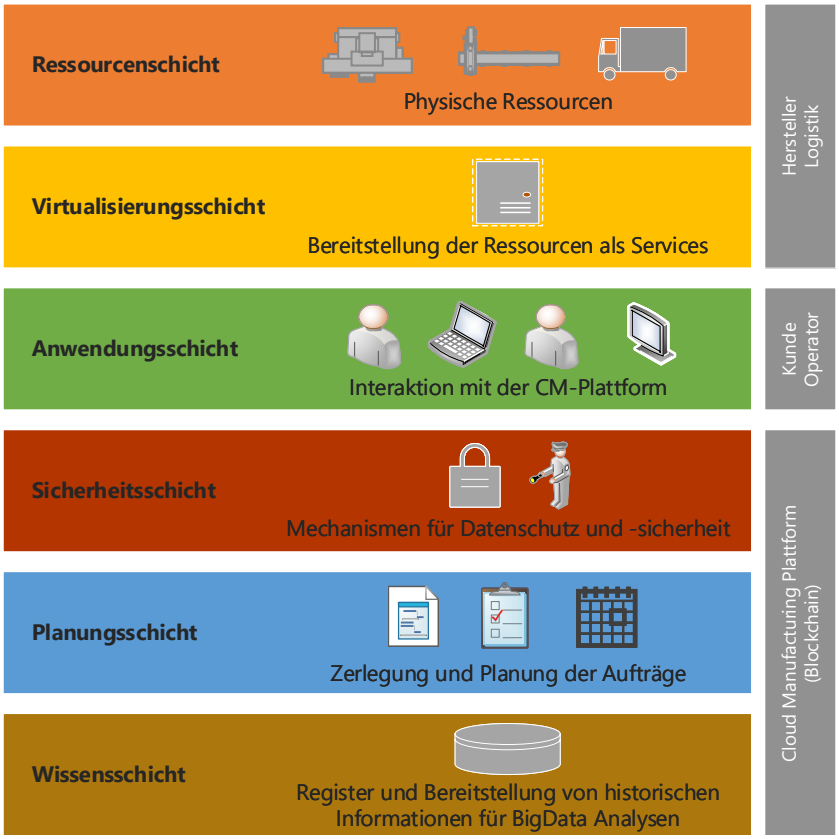

Abbildung 5.4: Darstellung der Schichten und deren Funktionen. Eigene Darstellung

Virtualisierungsschicht

Die Virtualisierungsschicht ist für die Erfassung der Ressourcen bei den Providern zuständig. Diese werden als Services auf der CM-Plattform zur Verfügung gestellt. Die Beschreibung der Fähigkeiten und Kapazitäten erfolgt in Smart Contracts. Jede Ressource die als Service angeboten wird, wird über ein Smart Contract repräsentiert. Dabei werden die Fähigkeiten der Ressourcen in einen Datenmodell beschrieben, um dann die Zuordnung von Aufträgen zu ermöglichen.

Anwendungsschicht

Diese Schicht dient als Möglichkeit zur Interaktion mit der Blockchain. Kunden können ihre Kundenaufträge als Smart Contracts auf der Blockchain veröffentlichen und den Status der Herstellung einsehen. Dateien, wie CAD-Daten mit den Konstruktionspläa-

nen, können auf das Dateisystem hochgeladen werden. Ebenso können die Operatoren ihre Dienstleistung für die Planung auf der Blockchain veröffentlichen und somit den Kunden anbieten. Weiterhin können alle Smart Contracts, die auf der Blockchain veröffentlicht worden sind, abgerufen werden. Das bietet den Kunden sowie den Operatoren eine Möglichkeit die Datenmodelle der Ressourcen einzusehen.

Sicherheitsschicht
Mit dieser Schicht soll der Schutz sowie die Sicherheit der Daten der Teilnehmer ermöglicht werden. CAD-Dateien die auf das Dateisystem hochgeladen werden, sollen nur von autorisierten Teilnehmern abgerufen werden können. Weiterhin sollen die Teilnehmer sensible Daten nicht auf der Blockchain veröffentlichen müssen. So sind Kundenangaben wie der Firmenname oder die Adresse nur notwendig, um einen Auftrag zu planen und werden nicht dauerhaft auf der Blockchain abgelegt. Diese Angaben werden lediglich an die Ressourcen weitergegeben, die diese Informationen benötigen, um beispielsweise das Fertigprodukt zu versenden.

Planungsschicht
Die Zerlegung und Planung der Kundenaufträge wird dieser Schicht zugeordnet. Die veröffentlichten Smart Contracts die einen Kundenauftrag repräsentieren, enthalten Meta-Informationen über das Projekt beziehungsweise das Produkt. Der Kundenauftrag wird so lange zerlegt, bis für jede (Teil-) Aufgabe Fertigungsressourcen ermittelt werden können. Dabei werden die Meta-Daten mit den beschriebenen Fähigkeiten der Ressourcen verglichen, um eine Zuordnung zu ermöglichen.

Wissensschicht
Die Wissensschicht hat hauptsächlich die Funktion eines Verzeichnisses für die CM-Plattform. Das Verzeichnis ist ein Smart Contract, der als erstes veröffentlicht wird. Die Smart Contracts für die Ressourcen sowie die der Operatoren melden sich über die Adresse des Verzeichnisses auf der CM-Plattform an. Smart Contracts mit den Kundenaufträgen können dann eine Liste der veröffentlichten Smart Contracts für die Planung und deren Adressen auf der Blockchain abrufen. Die Smart Contracts für die Planung können ihrerseits eine Liste mit allen Ressourcen abrufen.

Auch beinhaltet diese Schicht alle historischen Daten beziehungsweise Transaktionen. Die Informationen können dazu verwendet werden um Analysen durchzuführen. Somit

können einerseits Prognosen für Ressourcenauslastungen erstellt werden. Andererseits können Planungen effizienter erfolgen. Bereits ausgewertete Meta-Informationen und die zugeordneten Fertigungsressourcen können gespeichert und bei Bedarf abgerufen werden.

5.5 Smart Contracts

Für die CM-Plattform werden acht Smart Contract Typen eingeführt. Diese werden nachfolgend näher beschrieben.

5.5.1 Smart Contract Typen

Smart Contract für das Verzeichnis (Verzeichnis-Smart-Contract)
Über ein Verzeichnis können die Teilnehmer der Plattform die Adressen der Smart Contracts für die verschiedenen Services abrufen. Dafür ist es notwendig, dass sich die Anbieter der Services beim Verzeichnis-Smart-Contract (VSC) anmelden. Anschließend können die Teilnehmer die Adressen der Smart Contracts für die Services abrufen.

Abbildung 5.5: Klassendiagramm für Verzeichnis-Smart-Contract. Eigene Darstellung

Abbildung 5.5 zeigt das Klassendiagramm für den VSC. Die Adressen der angemeldeten Ressourcen sowie der Operatoren werden in jeweils einer Map als Datenstruktur gespeichert. Über die Methoden *setzeNeueRessource()* sowie *setzeNeuenPlanungsdienst()* werden die Ressourcen beziehungsweise OSCs angemeldet. Dabei wird die Adresse des Smart Contract, der die Methode im VSC aufruft, als Wert und mit einer eindeutigen Identifikationsnummer als Schlüssel, in der jeweiligen Map abgelegt. Die Identifikationsnummer ist eine Kombination aus HerstellerId und RessourcenId beziehungsweise OperatorId und der Id des Smart Contracts. Diese Ids werden als Argumente beim

Aufruf übergeben. Die Parameter sind in der Signatur der Methode aus Gründen der Übersichtlichkeit ausgelassen worden. Über die Methoden *holeAlleRessourcen()* sowie *holeAllePlanungsdienste()* werden alle Adressen der jeweiligen Smart Contracts zurück gegeben.

Smart Contract für Services (Service-Smart-Contract)
Der Service-Smart-Contract ist eine abstrakte Klasse und dient als Strukturelement. Die Unterklassen sind die Services die als Smart Contracts veröffentlicht werden.

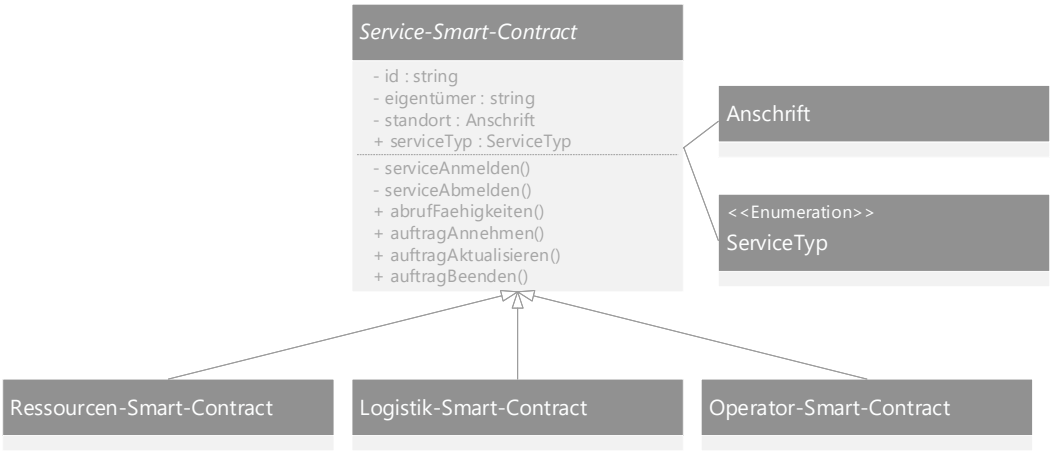

Abbildung 5.6: Klassendiagramm für Service-Smart-Contract. Eigene Darstellung

Abbildung 5.7 zeigt das Klassendiagramm für den Service-Smart-Contract und die Unterklassen die im weiteren Verlauf erläutert werden. Das Datenfeld *standort* ist vom Typ *Anschrift*. Es ist an das Datenmodell von Lu et al. angelehnt [35]. Die Anschrift enthält das Land, die Region, eine PLZ sowie die Stadt und Straße mit der Hausnummer. Die Adressangaben werden für eine Zuordnung von Aufgaben auf Ressourcen verwendet, da die Fertigungsressourcen räumlich verteilt liegen können. Das Datenfeld *serviceTyp* ist eine Enumeration und es können Aufzählungstypen wie „Spedition“ oder „Zusammenbau“ definiert werden. Die Methoden *serviceAnmelden()* und *serviceAbmelden()* sind für die Registrierung im Verzeichnis zuständig. Weiterhin wird die Ressource über diese Methode bei dem Hersteller oder den Logistik-Dienstleister angemeldet. Über die Methode *abrufFaehigkeiten()* können die Fähigkeiten der Ressourcen oder der Logistik abgerufen werden. Die Methode *auftragAnnehmen()* weißt dem Service einen Auftrag zu und

auftragAktualisieren() fragt den aktuellen Status des Auftrags an. Über die Methode *auftragBeenden()* kann ein spezifischer Auftrag abgebrochen werden.

Smart Contract für Ressourcen (Ressourcen-Smart-Contract)
Dieser Smart Contract, als Ressourcen-Smart-Contract (RSC) bezeichnet, enthält relevante Informationen über die Fertigungsressource, die als Service auf der CM-Plattform veröffentlicht wird. Der RSC ist eine Unterklasse des Service-Smart-Contracts.

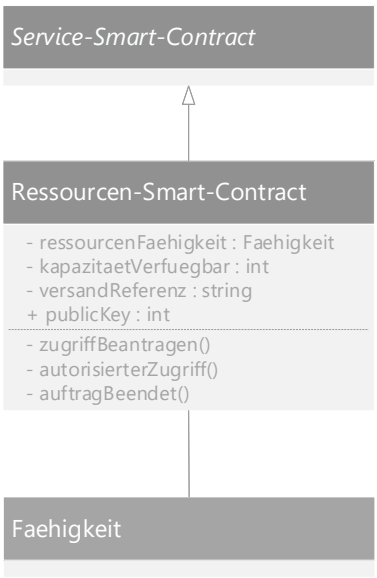

Abbildung 5.7: Klassendiagramm für Ressourcen-Smart-Contract. Eigene Darstellung

Abbildung 5.7 zeigt das Klassendiagramm für die RSCs. Das Datenfeld *publicKey* enthält den öffentlichen Schlüssel dieses Smart Contracts und wird für den autorisierten Zugriff auf eine CAD-Datei benötigt. Über das Datenfeld *kapazitaetVerfuegbar* wird die Kapazität der Ressource gespeichert.

Wurde ein Auftrag einer Ressource zugeordnet wird die Methode *zugriffBeantragen()* aufgerufen. Zusätzlich wird der Ressource eine Referenznummer für den Versand mitgeteilt, die im Datenfeld *versandReferenz* gespeichert wird. Diese enthält die Versandinformationen für das Produkt. Der Empfänger kann der Kunde, aber auch eine weitere Fertigungsressource sein.

Weiterhin wird der PublicKey der Ressource an den Auftraggeber übermittelt. Dieser dient dazu, eine CAD-Datei zu verschlüsseln und so einen autorisierten Zugriff zu ermöglichen. Ist die CAD-Datei mit dem PublicKey der Ressource verschlüsselt, kann diese über den eigenen PrivateKey die CAD-Datei entschlüsseln. Als Bestätigung erhält der RSC die Referenz zu der Datei auf dem Dateisystem. Über die Methode *autorisierterZugriff()* kann die Datei abgerufen werden.

Ist ein Auftrag beendet worden, wird die Methode *auftragBeendet()* aufgerufen. Es werden alle notwendigen Informationen an den Hersteller als Provider der Ressource und den Kunden. Weiterhin wird an den Smart Contract für die Logistik die Abholbereitschaft des Produkts signalisiert.

Der RSC hält in einer Datenstruktur die eigenen Fähigkeiten als Datenmodell. Das hier vorgestellte und vereinfachte Datenmodell ist an die Arbeit von Lu et al. angelehnt [35].

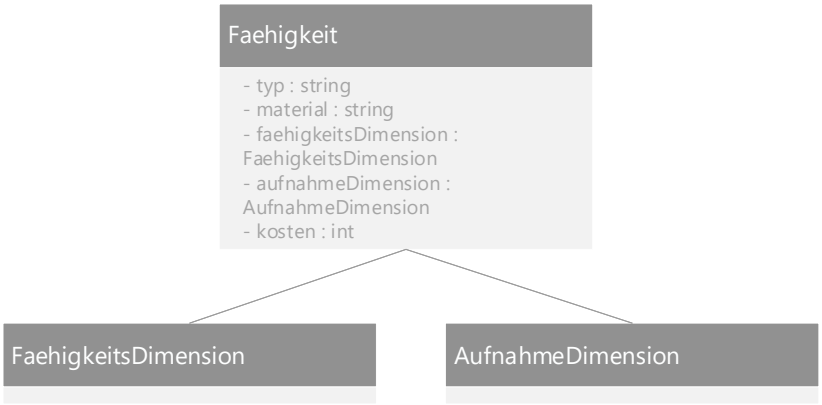

Abbildung 5.8: Klassendiagramm für das Datenmodell der Fähigkeiten. Eigene Darstellung

Abbildung 5.8 zeigt das Klassendiagramm für das Datenmodell der Fähigkeiten einer Fertigungsressource. Das Datenfeld *material* gibt das Material an, das von der Ressource verarbeitet werden kann. Die *faehigkeitsDimension* beschreibt beispielsweise die Lochbreite einer Bohrmaschine, Toleranzen bei der Herstellung oder wie die Werkstücke verarbeitet werden können. Es kann je nach Typ der Fertigungsressource variieren. Das definierte Modell kann dann für einen Kundenauftrag durch den Kunden verwendet werden oder für eine Zuordnung beziehungsweise Planung eines Operators. Die *aufnah-*

meDimension beschreibt welche Werkstücke durch die Fertigungsressource bearbeitet werden können.

Smart Contract für die Logistik (Logistik-Smart-Contract)
Transportunternehmen, welche die Logistik der (Teil-)Produkte übernehmen, können ebenfalls ihre Dienstleistungen als Services anbieten. Für diesen Zweck werden Logistik-Smart-Contract (LSC), als Unterklasse des Service-Smart-Contracts, auf der Plattform veröffentlicht. Wie die RSCs werden sie bei der Planung berücksichtigt, wenn ein Produkt nach der Herstellung entweder als Fertigprodukt zum Kunden versendet wird oder als Teilprodukt zu der nächsten Fertigungsressource. Für die Planung können Faktoren wie Transportdauer, Versandart sowie die Kosten beschrieben werden [35].

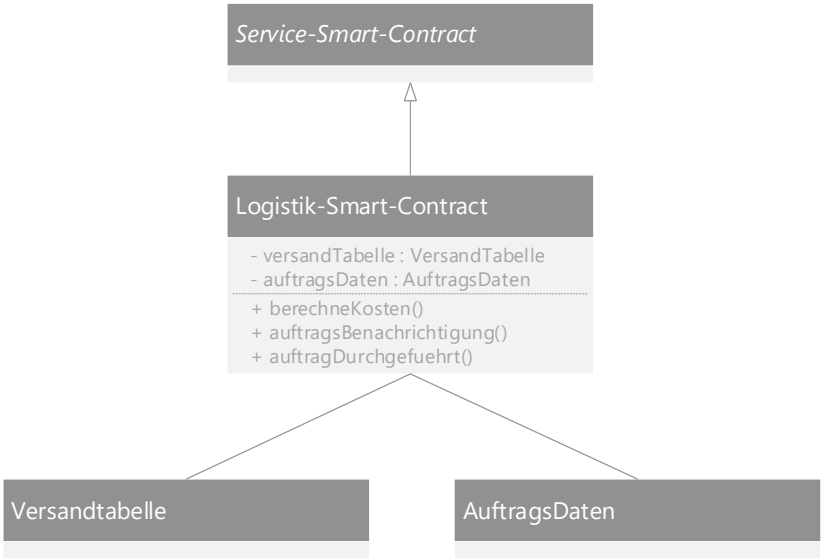

Abbildung 5.9: Klassendiagramm für Logistik-Smart-Contract. Eigene Darstellung

Abbildung 5.7 zeigt das Klassendiagramm für die LSCs. Das Datenfeld *versandTabelle* beinhaltet das Datenmodell für die Versandtabelle. Hier werden Angaben wie Versandart oder Transportdauer abhängig vom Versand- und Zielort beschrieben. Die Methode *berechneKosten()* bekommt beim Aufruf den Versandort sowie das Ziel, die Versandart und andere Angaben die für eine Berechnung der Versandkosten berücksichtigt werden sollen. Die berechneten Kosten sind der Rückgabewert der Methode.

Wenn der LSC einen Auftrag zugewiesen bekommt, wird die Methode *auftragsBenachrichtigung()* aufgerufen. Die Informationen zum Auftrag werden gespeichert und eine Referenznummer wird zurückgegeben. Diese wird an den RSC übertragen, der die Fertigung des Produkts durchführt, welches dann anschließend versendet wird. Die Methode *auftragDurchgefuert()* wird aufgerufen, wenn das Produkt an den Kunden oder an die nächste Fertigungsressource ausgeliefert worden ist. Damit erhält der Logistik-Dienstleister eine Benachrichtigung, dass der Auftrag durchgeführt worden ist.

Smart Contract für die Provider (Provider-Smart-Contract)
Die Ressourcen melden sich einerseits bei dem VSC und andererseits auch bei den Providern an, denen sie gehören. Nimmt eine Ressource einen Auftrag an, wird das ebenfalls dem entsprechenden Provider-Smart-Contract (PSC) durch die Ressource mitgeteilt. Hat eine Ressource die Bearbeitung abgeschlossen, wird der PSC von dem entsprechenden Smart Contract benachrichtigt. Sobald alle Aufgaben eines Kundenauftrages abgeschlossen sind, bekommt der Provider über diesen Smart Contract eine Benachrichtigung. Die Informationen der Benachrichtigung enthalten alle in Anspruch genommenen Ressourcen, die für ein Kundenprojekt verwendet wurden und die Kosten für die Nutzung der Ressourcen. Der Hersteller kann dann gekapselt eine Rechnung für diesen Auftrag erstellen und die Zahlung anfordern.

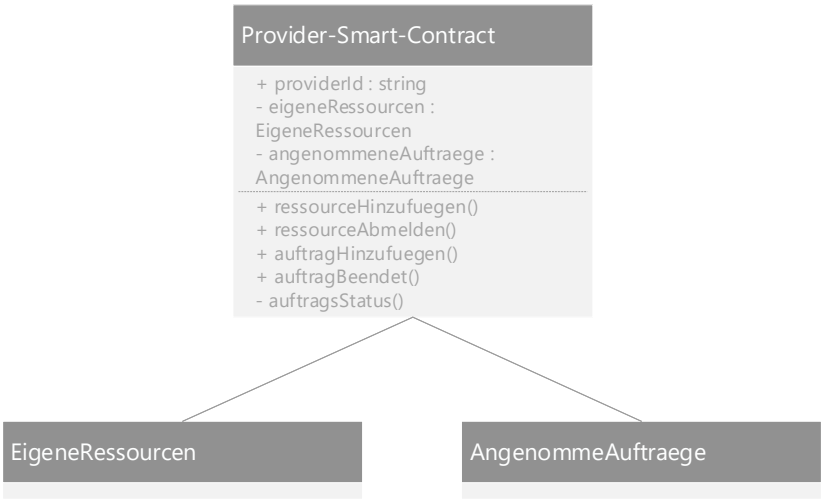

Abbildung 5.10: Klassendiagramm für Provider-Smart-Contract. Eigene Darstellung

Abbildung 5.10 zeigt das Klassendiagramm für den Provider-Smart-Contract. Die *providerId* ist ein String der eine eindeutige Identifikation des Providers auf der CM-Plattform ermöglicht. Die als Service angemeldeten Ressourcen werden in der Datenstruktur *eigeneRessourcen* gespeichert. Alle angenommenen Aufgaben der eigenen Ressourcen werden mit einen Status und einer Auftragsnummer im Datenfeld *angenommeneAuftraege* gespeichert. Die Auftragsnummer repräsentiert den Kundenauftrag und die einzelnen dazugehörenden Aufgaben als Teilaufgaben.

Die Methode *ressourceHinzufuegen()* fügt neue Ressource zu den eigenen Ressourcen hinzu und *ressourceEntfernen()* entfernt diese wieder. Wird ein Auftrag beziehungsweise eine Aufgabe von einer Ressource bearbeitet, wird die Methode auftragHinzufuegen() aufgerufen. Es wird dann, wie bereits beschrieben, der Auftrag und die dazugehörenden Aufgaben gespeichert. Hat eine Ressource ihren Auftrag beendet wird die Methode *auftragBeendet()* aufgerufen und der Status des Auftrags wird entsprechend angepasst. Weiterhin wird intern die Methode auftragsStatus() aufgerufen und es wird geprüft, ob alle Aufgaben eines Auftrags beendet worden sind. Ist das der Fall, erhält der Provider eine Benachrichtigung.

Smart Contract für die Zerteilung und Planung (Operator-Smart-Contract)

Für die Zerteilung und Planung werden ebenfalls Smart Contracts verwendet. Diese werden als Operator-Smart-Contract (OSC) bezeichnet. Sie sollen die Zuordnung von Aufträgen zu den angebotenen und verfügbaren Services vornehmen. Alle vorhanden Service-Smart-Contracts werden dafür abgeglichen, um einen passenden Service zu finden. Wird eine Übereinstimmung nicht erreicht, erfolgt eine Zerteilung des Auftrags in Teilaufträge. Für die resultierenden Teilaufträge werden dann wieder Fertigungsressource ermittelt. Konnte keine Zuordnung erfolgen und der Auftrag kann nicht weiter zerlegt werden, erhält der Kunde eine entsprechende Benachrichtigung

Abbildung 5.11 zeigt das Klassendiagramm für den OSC. Die Datenstruktur *zuordnungsArgumente* enthält als Datenmodell das der Fähigkeiten der Ressourcen (5.8). Die Ressourcen werden temporär im Datenfeld *ressourcenListe* abgelegt. Das Datenfeld wird nicht auf der Blockchain persistiert und entsprechend nach der Planung eines Auftrags verworfen. Ebenfalls wird das Datenfeld *kundenAuftrag* lediglich temporär für die Auftragsplanung verwendet. Die hier verwendeten Kundendaten, wie beispielsweise die Anschrift, sollen nicht dauerhaft auf der Blockchain abgelegt werden.

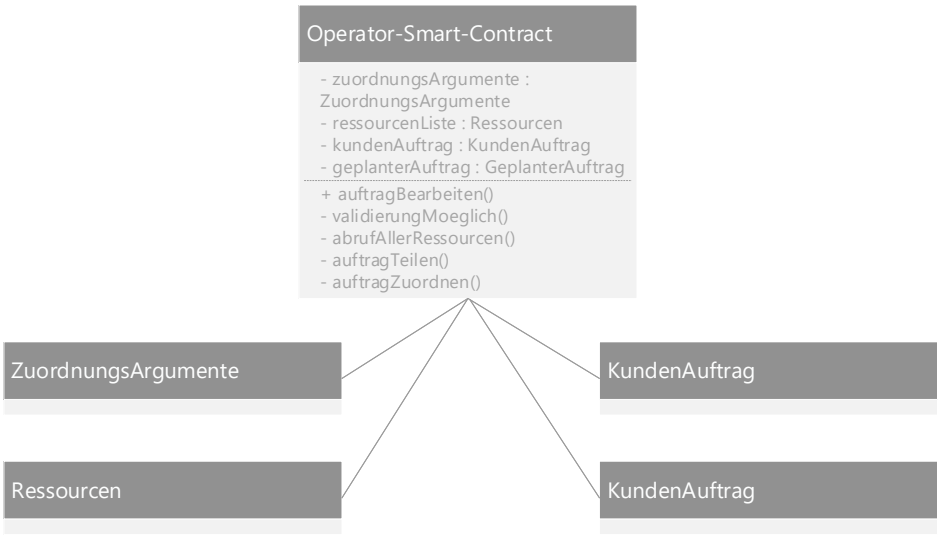

Abbildung 5.11: Klassendiagramm für Operator-Smart-Contract. Eigene Darstellung

Der geplante Auftrag wird mit den Aufgaben und den jeweils zugeordneten Ressourcen für die Fertigung und den Versand im Datenfeld *geplanterAuftrag* abgelegt. Es werden lediglich Meta-Informationen zum Produkt beziehungsweise die Aufgaben und die zugeordneten Ressourcen auf der Blockchain gespeichert. Diese Informationen können dann zukünftig bei Planungen berücksichtigt werden.

Über die Methode *auftragBearbeiten()* übermittelt ein Kunde die Meta-Informationen zu einen Auftrag. Diese Informationen werden über das gleiche Datenmodell repräsentiert (5.8). Zunächst wird die Methode *validierungMoeglich()* aufgerufen. Es wird geprüft ob der OSC diesen Auftrag bearbeiten kann, indem die Datenmodelle verglichen werden. Wenn der OSC den Auftrag planen kann, werden alle im Netzwerk vorhandenen Ressourcen mit den jeweiligen Fähigkeiten über die Methode *abrufAllerRessourcen()* abgerufen. Ansonsten erhält der Kunde eine entsprechende Benachrichtigung.

Anschließend wird der Auftrag über die Methode *auftragTeilen()* daraufhin geprüft, ob dieser aus Aufgaben besteht und diese zerteilt beziehungsweise einzeln geplant werden müssen. Für die Analyse können bereits geplante Aufträge berücksichtigt werden, um einzelne Aufgaben oder ganze Aufträge den entsprechenden Ressourcen zuzuordnen. Die einzelnen Aufgaben werden dann den entsprechenden Ressourcen zugeordnet und in *geplanterAuftrag* abgelegt. Dies erfolgt über die Methode *auftragZuordnen()*. Abschließend

erhält der Kunde, beziehungsweise der Smart Contract für den Auftrag, den geplanten Auftrag als Rückgabewert. Können keine Ressourcen während der Planung der Aufgaben zugeordnet werden können, so erhält der Kunde eine entsprechende Rückmeldung.

Smart Contract für Kundenaufträge (Auftrags-Smart-Contract)
Ein Auftrag wird durch einen Kunden ebenfalls als Smart Contract veröffentlicht und als Auftrag-Smart-Contract (ASC) bezeichnet. Dieser enthält Meta-Informationen über das Produkt. Weiterhin enthält er weitere Angaben wie beispielsweise zu Zahlungszielen, Versandanforderungen und Kostenvorstellungen für das Produkt, die bei der Auswahl eines Services berücksichtigt werden müssen. Er enthält weiterhin eine Referenz für die CAD-Datei, die auf dem Dateisystem abgelegt wurde. Kundenangaben, wie beispielsweise die Adresse oder personenbezogene Daten, werden nicht im Smart Contract gespeichert. Diese werden lediglich an den OSC übermittelt. Der ASC wird beendet, sobald das fertige Produkt beim Kunden eingetroffen ist.

Abbildung 5.12 zeigt das Klassendiagramm für den ASC. Eine eindeutige Identifikationsnummer für den Kunden als Teilnehmer auf der CM-Plattform wird in *kundenId* als String gespeichert. Die Identifikationsnummer für den Auftrag der über den ASC repräsentiert wird, wird in der *auftragsId* als String gespeichert. Im Datenfeld *auftrag* wird der Auftrag des Kunden abgelegt. Der Auftrag kann aus einer oder mehreren Aufgaben bestehen und wird im Datenfeld *aufgabe* gespeichert. Weiterhin werden hier Konditionen für den Auftrag abgelegt, die vor der Erteilung überprüft werden. Die Konditionen sind im Modell die maximalen Gesamtkosten für den Auftrag als *maxGesamtKosten* sowie Lieferkonditionen als *lieferKonditionen*.

Wenn die Auftrage an die OSC übermittelt werden, dann werden diese im Datenfeld *angefragteOperatoren* gespeichert, um zu Überprüfen ob alle Verfügbaren OSC geantwortet haben. Die von einen OSC erhaltenen geplanten Aufträge werden im Datenfeld *geplanterAuftrag* abgelegt. Mit der Veröffentlichung des ASC wird ebenfalls ein Smart Contract für die Genehmigung veröffentlicht. Die Adresse zu diesem Smart Contract wird in *gscReferenz* abgelegt.

Die Methode *auftragAnfragen()* übermittelt den Auftrag an alle im Verzeichnis abgelegten OSCs. Wird als Rückgabe der Wert erhalten, dass eine Planung nicht möglich ist, so wird das in *angefragteOperatoren* abgelegt. Es wird dann überprüft, ob von jedem angefragten OSC eine Antwort erhalten worden ist. Wenn eine Planung von keinen OSC

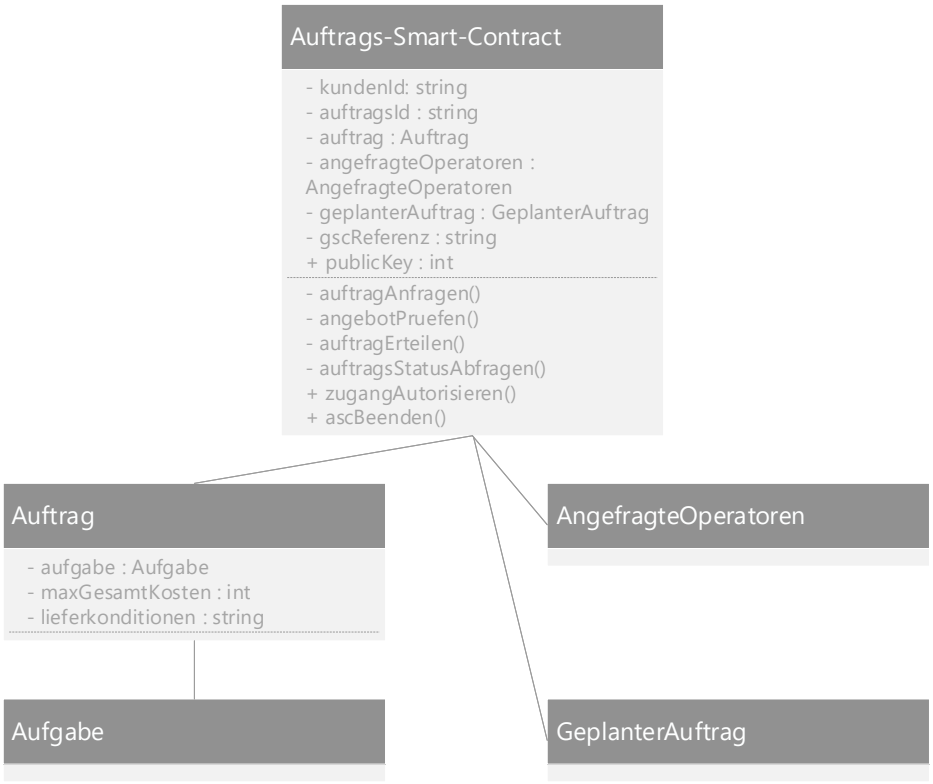

Abbildung 5.12: Klassendiagramm für Auftrags-Smart-Contract. Eigene Darstellung

durchgeführt werden kann oder eine festgelegte Antwortzeit überschritten ist, wird eine entsprechende Benachrichtigung an den Kunden übermittelt. Der Kunde hat dann die Möglichkeit das Datenmodell anzupassen, damit sein Auftrag durch die OSCs verarbeitet werden kann.

Anschließend wird die Methode *angebotPruefen()* ausgeführt. Es werden alle geplanten Aufträge analysiert und mit den Konditionen abgeglichen. Der Auftrag, beziehungsweise die Aufgaben werden dann den ausgewählten Ressourcen über die Methode *auftragErteilen()* zugewiesen und in *geplanterAuftrag* vermerkt. Über die Methode *auftragsStatusAbfragen()* wird der Status des Auftrags bei den zugeteilten Ressourcen abgefragt. Die Methode *zugangAutorisieren()* wird von den Fertigungsressourcen aufgerufen, um den Zugang für die auf den Dateisystem abgelegte CAD-Datei zu erhalten. Es wird anhand der zugewiesenen Ressourcen in *geplanterAuftrag* überprüft, ob die Ressource den Zugang benötigt. Um die Autorisierung vorzunehmen, wird der PublicKey der Ressource

an den Genehmigungs-Smart-Contract weitergeleitet und die Referenz wird zurück erhalten. Diese Referenz wird dann als Bestätigung an die Ressource zurückgegeben. Die Methode *ascBeenden()* beendet den Smart Contract und er kann nicht mehr aufgerufen werden.

Smart Contract für die Genehmigung (Genehmigungs-Smart-Contract)
Der Genehmigungs-Smart-Contract (GSC) wird verwendet, um eine CAD-Datei auf dem Dateisystem zu verschlüsseln. Beendet wird der Smart Contract, sobald der Kundenauftrag beendet wird.

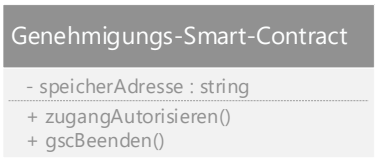

Abbildung 5.13: Klassendiagramm für Genehmigungs-Smart-Contract. Eigene Darstellung

Abbildung 5.13 zeigt das Klassendiagramm für den GSC. Die Referenz zu der CAD-Datei wird in der *speicherAdresse* abgelegt. Die Methode *zugangAutorisieren()* erhält den *PublicKey* der Ressource sowie den *PrivateKey* des Kunden. Diese beiden Schlüssel werden nicht gespeichert und werden verworfen, nachdem die Autorisierung für die entsprechende Ressource erfolgt ist. Mit den beiden Schlüsseln wird die CAD-Datei verschlüsselt und die Referenz wird zurückgegeben. Über die Methode gscBeenden() wird der Smart Contract beendet.

5.5.2 Interaktion der Smart Contracts

Abbildung 5.14 zeigt ein Sequenzdiagramm mit der Interaktion der Smart Contracts, um einen Auftrag zu Planen und den Ressourcen zuzuweisen. Das Diagramm veranschaulicht, wie der Zugang zu einer CAD-Datei autorisiert wird. Die Datei ist auf dem Dateisystem abgelegt und wird aus Gründen der Übersichtlichkeit im Diagramm nicht aufgeführt. Weiterhin ist die Interaktion der Ressourcen mit den jeweiligen Providern

und deren Smart Contracts nicht dargestellt. Die Interaktion bezieht sich auf die Veröffentlichung eines Auftrages über einen ASC, was auch der Startpunkt für das Sequenzdiagramm ist. Beendet wird die Sequenz mit der Rückgabe der Referenz zu der CAD-Datei, die dann von der Fertigungsressource abgerufen werden kann. Die Interaktion zwischen den Smart Contracts erfolgt wie nachfolgend beschrieben:

- Mit der Veröffentlichung des Auftrags wird der GSC initialisiert. Dieser beinhaltet die Referenz für die CAD-Datei, die auf dem Dateisystem abgelegt wurde.

- Der ASC fragt beim VSC alle verfügbaren OSCs an und erhält eine Liste mit diesen Smart Contracts.

- Die Meta-Informationen des Auftrags werden an die verfügbaren OSCs gesendet. Die OSCs analysieren die Meta-Informationen um zu prüfen, ob sie eine Planung vornehmen können. Das wird anhand des gespeicherten Datenmodells vorgenommen. Das Seuquenzdiagramm stellt den Fall da, dass das Datenmodell übereinstimmt und die Planung vorgenommen werden kann.

- Beim VSC werden alle verfügbaren Ressourcen angefragt und der OSC erhält eine entsprechende Liste.

- Alle vorhanden Ressourcen werden angefragt und der OSC erhält die Fähigkeiten und Kapazitäten der Ressourcen zurück. Anhand des Typen des Services (*service-Typ*) kann der OSC unterscheiden, ob es sich um eine Fertigungsressource oder eine Logistik-Dienstleistung handelt.

- Der OSC prüft anhand des Datenmodells ob eine Fertigungsressource zugewiesen werden kann. Konnte keine Übereinstimmung vorgenommen werden, wird der Auftrag geteilt und die einzelnen resultierenden Aufgaben werden anhand ihres Datenmodells mit den Ressourcen verglichen. Weiterhin werden die LSC für mögliche Transporte von Zwischenprodukten zwischen den Fertigungsressourcen oder für das Endprodukt zum Kunden analysiert und ausgewählt.

- Der geplante Auftrag wird mit den zugeordneten Ressourcen an den Kunden zurück gegeben.

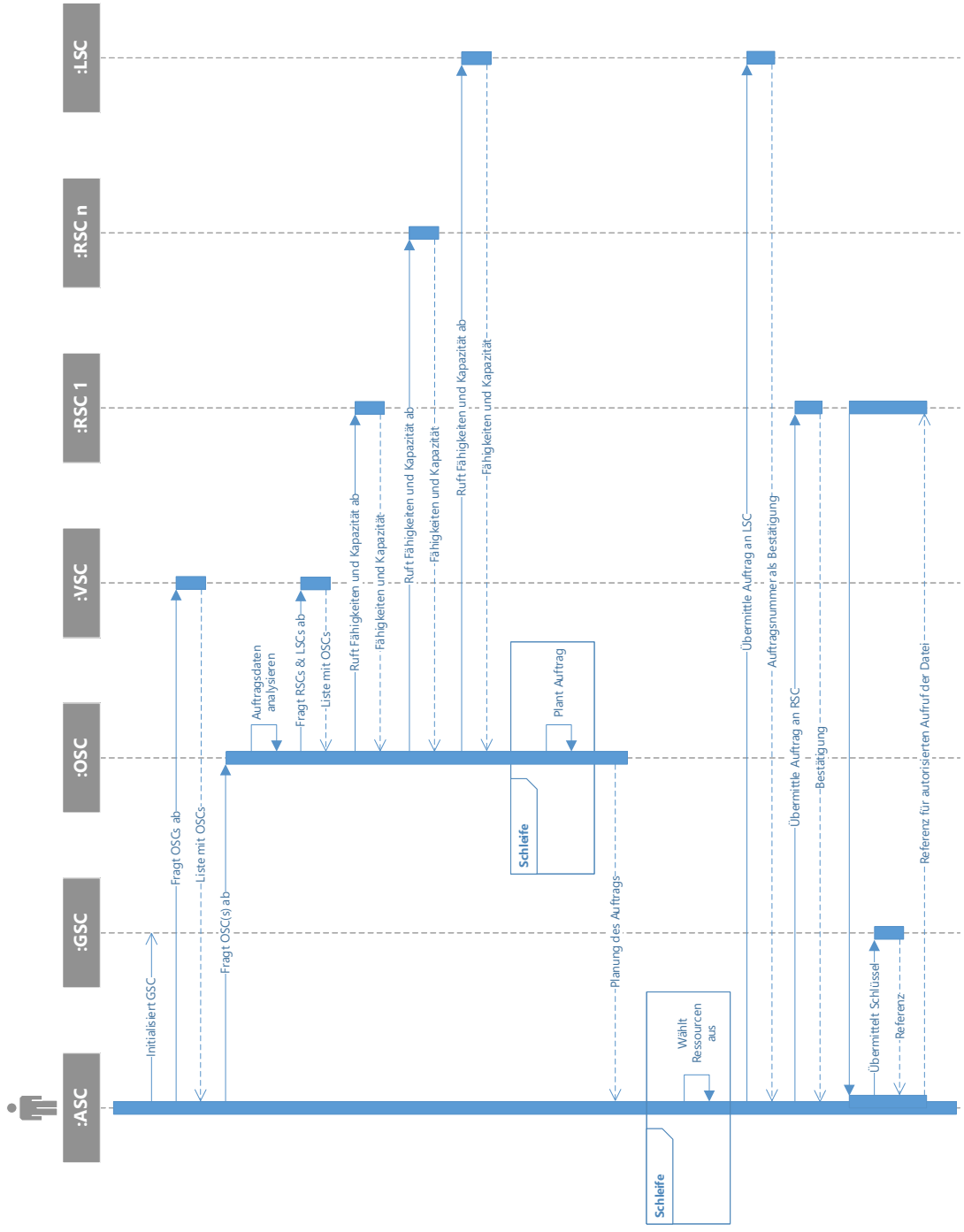

Abbildung 5.14: Sequenzdiagramm – Interaktion der Smart Contracts für die Planung und Zuweisung von Aufträgen. Eigene Darstellung

- Der ASC analysiert die geplanten Aufträge und wählt eine Planung daraus aus. Die Entscheidungskriterien für die Auswahl können beispielsweise Versandbedingungen oder Gesamtkosten sein.
- Die ausgewählten Ressourcen für die Logistik werden benachrichtigt und der ASC erhält eine Auftragsnummer als Referenz zurück.
- Der Auftrag beziehungsweise die Aufträge werden dann an die ausgewählten Fertigungsressourcen übermittelt. Weiterhin wird die Auftragsnummer der Logistik mitgesendet.
- Die entsprechenden RSCs fragen dann die CAD-Datei an, indem sie ihren PublicKey an den ASC senden. Es wird überprüft ob die Ressource die CAD-Datei benötigt, indem geprüft wird, ob die Ressource für den entsprechenden Auftrag eingeplant und ausgewählt wurde.
- Um den Aufruf der CAD-Datei für die Ressource zu autorisieren, wird der PublicKey der Ressource sowie der PrivatKey des ASCs an den GSC übermittelt. Dieser autorisiert den Zugriff und schickt die Referenz zu der CAD-Datei an den ASC zurück. Der gibt die Referenz als Rückgabewert weiter an die Ressource zurück die angefragt hat.

Kapitel 6

Implementierung

Im Kapitel 5 wurde ein Konzept für eine CM-Plattform ausgearbeitet und vorgestellt. Das Konzept basiert auf einer dezentralen Architektur mit einer Blockchain. Um das Design zu validieren wurde eine Implementierung vorgenommen. Zunächst wurde eine Blockchain als System ausgewählt. Auch wurde ein Dateisystem, auf dem beispielsweise eine CAD-Datei abgelegt werden kann, ausgewählt. Nachfolgend wird die Auswahl der entsprechenden Technologien für die Implementierung vorgestellt.

6.1 Auswahl der Blockchain-Technologie

Ethereum
Ethereum ist eine Implementierung der Blockchain, bei der jeder Teilnehmer die gleichen Rechte hat [58]. Es ist darauf ausgelegt Smart Contracts als Anwendungen auszuführen. Smart Contracts können für Ethereum in Solidity, eine statisch typisierte Sprache mit Ähnlichkeiten zu JavaScript, geschrieben werden. Für die Ausführung der Smart Contracts auf Ethereum wird die Laufzeit Ethereum Virtual Machine (EVM) verwendet, die Bytecode ausführt. Alle getätigten Transaktionen sowie die veröffentlichten Smart Contracts können von allen Teilnehmern eingesehen werden. Das bedeutet auch, dass alle Teilnehmer im Netzwerk den gleichen Ledger haben müssen. Als Konsensmechanismus wird PoW verwendet.

Ethereum kann auch als private Blockchain eingerichtet werden. Für diesen Zweck kann Geth herunter geladen und installiert werden [59]. Es ist ein Ethereum-Client, der sich mit dem öffentlichen Netzwerk verbinden kann. Allerdings ist es auch möglich einen Genesis-Block zu erstellen und dem Ethereum-Client diesen als Einstiegspunkt bekannt zu geben.

Hyperledger Fabric

Hyperledger Fabric ist ebenfalls eine Implementierung der Blockchain, wobei das Hauptmerkmal auf Modularität liegt [60]. Das bedeutet, dass beispielsweise der Konsensmechanismus als Dienst für die Plattform erweitert oder ersetzt werden kann. Auch bietet es eine Vergabe von Rechten für die Teilnehmer. So kann der Betreiber der Blockchain festlegen, welche Blocks der Blockchain von bestimmten Teilnehmern eingesehen werden können. Somit benötigen nicht alle Teilnehmer dieser Blockchain-Implementierung den gleichen Ledger.

Smart Contracts werden in Hyperledger Fabric als Chaincode bezeichnet und sind in Golang geschrieben, eine von Google entwickelte Programmiersprache. Für die Veröffentlichung von Chaincode können Kanäle definiert werden. Diese Kanäle ermöglichen bei der Veröffentlichung, dass nur bestimmte Teilnehmer diese Chaincodes ausführen und einsehen können.

IOTA

IOTA ist speziell für das IoT konzipiert worden und baut auf einen anderen Ansatz als die Blockchain Technologie auf [61]. Das Konzept beruht auf azyklischen Graphen, was bedeutet, dass eine Menge von Knotenpunkten eine eindeutig festgelegte Laufrichtung haben, wobei der Weg nicht mehr zum Ursprung zurückkehren kann. Diese Knotenpunkte repräsentieren Transaktionen.

Eine Transaktion eines Teilnehmers kann erst bestätigt werden, wenn dieser bereits zwei andere Transaktionen bestätigt hat. Somit ist die Besonderheit bei diesem Konzept, dass je mehr Teilnehmer das Netzwerk hat, desto schneller ist die Beglaubigung einer Transaktion. Damit ist es sehr gut geeignet für den Einsatz im IoT, da die Teilnehmer die miteinander vernetzten Maschinen repräsentieren. Derzeit werden von IOTA keine Smart Contracts unterstützt.

Fazit
Für die Implementierung wurde Ethereum ausgewählt, da es bereits den Aspekt, dass Smart Contracts eingesetzt werden können, mit sich bringt. Auch sind die Vorteile von Hyperledger für den Anwendungsfall nicht essentiell, da gerade die Transparenz für alle Teilnehmer für das CM entscheidend ist. Weiterhin hat Ethereum den Vorteil, dass das Netzwerk nicht von einer Stelle administriert werden muss. Demnach haben alle Teilnehmer im Netzwerk die gleichen Rechte und können die gesamte Blockchain einsehen.

6.2 Auswahl des Dateisystems

Eine CAD-Datei auf einer Blockchain wie Ethereum hochzuladen ist aufgrund der Größe dieser Dateien nicht sinnvoll. Einerseits muss diese im ganzen Netzwerk auf jeden teilnehmenden Knoten verteilt werden und andererseits sind Transaktionen mit Kosten verbunden. Der Anwendungsfall ist, dass der Dateizugriff nur für autorisierte Teilnehmer des Netzwerks ermöglicht werden soll. Für diesen Einsatzzweck eignet sich ein dezentrales Dateisystem, da dadurch auch die Notwendigkeit einer weiteren Instanz entfällt, wie beispielsweise einen Cloud-Provider, der die Speicherung der Datei übernimmt. Die Dateien auf einen dezentralen Dateisystem können verschlüsselt werden und sind somit nur für die Teilnehmer abrufbar, die die Datei entschlüsseln können.

So ein verteiltes Dateisystem ist Swarm [62]. Swarm ist ein nativer Dienst des Ethereum Web3 Systems, was eine Kollektion von Bibliotheken ist, die beispielsweise auch die Interaktion zwischen einen Frontend und Ethereum bereitstellt. Zum Zeitpunkt der Implementierung befand sich allerdings Swarm noch in der Entwicklungsphase.

Ein weiteres dezentrales Dateisystem ist das Interplanetary File System (IPFS) [63]. Wird eine Datei auf das IPFS hochgeladen, wird ein kryptografischer Hashwert berechnet, der die Speicheradresse als Referenz darstellt. Damit kann auch sichergestellt werden, dass über diese Referenz genau die Datei abgerufen werden kann, die auch hochgeladen worden ist. Eine Manipulation dieser Datei würde einen anderen kryptografischen Hashwert erzeugen. Die Datei wird verteilt auf das Netzwerk abgelegt. In Routingtabellen wird gespeichert, welche Knoten im Netzwerk die Datei beinhalten.

6.3 Umsetzung

Um das vorgestellte Konzept zu validieren wurde die Implementierung der Smart Contracts für Ethereum vorgenommen. Hierfür wurde eine private Ethereum Blockchain über Geth aufgesetzt und Truffle als Entwicklungswerkzeug [64] verwendet. Truffle kann für die Durchführung von Unit Test der Smart Contracts verwendet werden, für das compilieren und das Veröffentlichen. Ein Vorteil einer privaten Ethereum Blockchain ist, dass keine echten Kosten für die Transaktionen anfallen um die Smart Contracts auf der Blockchain zu veröffentlichen und auszuführen. Der Mining-Prozess kann auch schneller durchgeführt werden, um die Transaktionen und somit die Veröffentlichung und Verteilung der Smart Contracts auf der Blockchain zu ermöglichen. Weiterhin wurde IPFS lokal installiert und mit dem Netzwerk verbunden, um eine Datei hochzuladen, die eine CAD-Datei repräsentiert.

Für die Validierung des Szenarios (Kapitel 7) wurde ein VSC, ASC, ein OSC, ein LSC sowie drei RSCs entwickelt. Dabei sollte die Interaktionen der Smart Contracts, wie in der Abbildung 5.14 vorgestellt, anhand dieser Implementierung demonstriert werden. Weiterhin wurden vier verschiedene Konten erstellt, um mit der Blockchain zu interagieren und um unterschiedliche Teilnehmer zu simulieren. Jedes Konto hat einen Private Key (Privater Schlüssel) sowie einen Public Key (Öffentlicher Schlüssel).

Die Berechnung der Kosten weicht vom Konzept ab. Das Konzept sieht vor, dass die Gesamtkosten mit den durch den OSC berechneten Kosten übereinstimmen, beziehungsweise kleiner sein müssen. Für die Implementierung wurden die einzelnen Kosten der Aufgaben gegen die Kosten der Ressource überprüft.

Kapitel 7

Validierung

Dieses Kapitel stellt die Validierung der umgesetzten Smart Contracts für die CM-Plattform vor. Anhand eines Szenarios wird das im Konzept (Kapitel 5) vorgestellte Datenmodell befüllt. Die bei der Implementierung entwickelten Smart Contracts und die Interaktion zwischen diesen, soll zeigen, dass mit dem ausgearbeiteten Konzept eine Planung von Aufträgen sowie die Orchestrierung der als Smart Contracts umgesetzten Services auf der CM-Plattform durchgeführt werden kann.

7.1 Szenario

Um die prototypische Implementierung zu validieren, wurde ein Szenario ausgearbeitet. Das Szenario ist darauf ausgelegt, dass ein Auftrag mit vier Auftragsbestandteilen über die CM-Plattform den entsprechenden Fertigungsressourcen zugeordnet wird. Damit die Implementierung und das Szenario überschaubar bleiben, wird das Datenmodell (Abbildung 5.8) lediglich Basistypen beinhalten, die für eine Zuordnung relevant sind.

Fertigungsressourcen

Nachfolgend werden die Fertigungsressourcen und ihre Fähigkeiten gezeigt. Tabelle 7.1 zeigt das befüllte Datenmodell der Ressourcen (R1 bis R4) und deren Fähigkeiten (F1 und F2). Die Dimensionen, in der Tabelle mit Dim. abgekürzt, haben einen abstrahierten Wert, der ein weiteres Datenmodell repräsentiert. Dieses weitere Datenmodell kann

beispielsweise Abmessungen der Maschine enthalten oder Legierungen die verarbeitet werden können. Für die Zuordnung muss die Dimension in den Meta-Informationen des Kundenauftrages kleiner gleich der Dimension sein, die in der Fertigungs-Ressource hinterlegt ist. Der Faktor Kosten wird als ganzzahliger Wert ohne Einheit, beziehungsweise bestimmte Währung geführt. In diesem Szenario wird vom Kunden in den Meta-Informationen des Kundenauftrages ein Wert für die gesamten Auftrag hinterlegt. Die einzelnen Ressourcen haben auch einen Standort. Ressource 1 sowie Ressource 2 sind in Deutschland und Ressource 3 sowie Ressource 4 sind in Frankreich.

	Typ	**Material**	**Dim. der Fähigkeit**	**Dim. der Aufnahme**	**Kosten**
R1	Fräse	Aluminium	5	8	5
R2, F1	Bohrer	Aluminium	2	10	2
R2, F2	Bohrer	Holz	2	10	2
R3, F1	Gewindeschneider	Aluminium	5	3	4
R3, F2	Gewindeschneider	Metall	6	3	4
R4	Schleifen	Aluminium	8	10	1

Tabelle 7.1: Befülltes Datenmodell der Ressourcen

Kundenauftrag

Der Kundenauftrag beinhaltet die Herstellung eines Pedals für ein Fahrrad. Für den Auftrag werden die Gesamtkosten von 20 hinterlegt. Die Zuordnung der Ressourcen erfolgt allerdings über den Vergleich der maximalen Kosten der einzelnen Aufgaben mit den Kosten der Ressourcen. Abbildung 7.1 zeigt das Pedal nach dem letzten Arbeitsschritt.

Abbildung 7.2 zeigt die Arbeitsschritte, die für die Herstellung des Pedals notwendig sind. Vier Arbeitsschritte bauen aufeinander auf. Das Rohmaterial für das Pedal wird von der ersten Ressource gestellt.

Der Kundenauftrag beinhaltet somit vier Auftragsbestandteile, die sequentiell abgearbeitet werden müssen. Die Meta-Informationen für den Auftrag wurden in dem Datenmodell abgelegt, die von den Ressourcen verwendet werden. Tabelle 7.2 stellt die Auftragsbestandteile (ABT1 bis ABT4) vor.

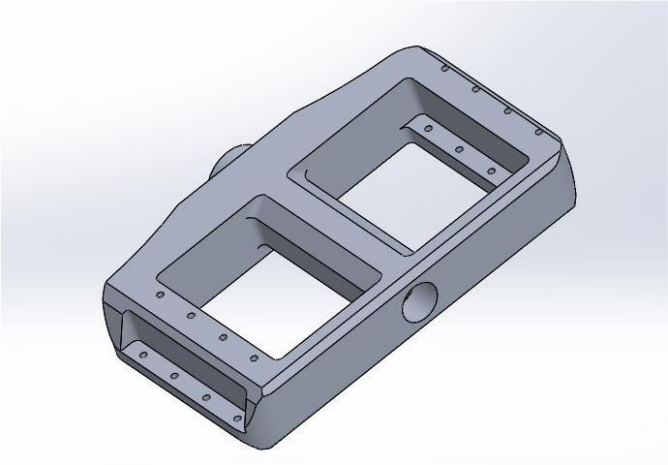

Abbildung 7.1: Darstellung des Kundenproduktes. Quelle [65]

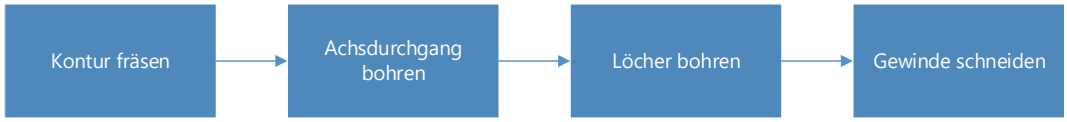

Abbildung 7.2: Prozess der Herstellung eines Pedals. Quelle in Anlehnung an [65]

	Zuordnung	**Material**	**Dim. des Arbeit**	**Dim. des Produkts**	**Kosten**
ABT1	Fräse	Aluminium	5	8	10
ABT2	Bohrer	Aluminium	2	10	5
ABT3	Bohrer	Aluminium	1	10	4
ABT4	Gewindeschneider	Aluminium	2	3	4

Tabelle 7.2: Befülltes Datenmodell eines Auftrags

Eine für den Auftrag hinterlegte CAD-Datei beinhaltet die technische Zeichnung für das Pedal. Diese wird im öffentlichen IPFS Netzwerk abgelegt. Die Referenz zu der Datei wird im GSC, bei der Veröffentlichung des Smart Contracts gespeichert.

Operator

Weiterhin wird auf der CM-Plattform ein OSC veröffentlicht, der die Zuordnung der

vorgestellten Ressourcen bearbeiten kann. Als Datenmodell soll ebenfalls das der Ressourcen verwendet werden.

Logistik
Für die Logistik wurde ein LSC erstellt, der die Länder Deutschland, Österreich sowie Frankreich bedienen kann. Der Kostenfaktor wird hier nicht weiter betrachtet, kann aber beispielsweise über die Gesamtkostenvorstellung des Kunden mitberücksichtigt werden.

Veröffentlichung der Smart Contracts
Wie bereits aufgeführt, wurden vier Konten erstellt, die jeweils einen Teilnehmer auf der CM-Plattform repräsentieren. Tabelle 7.3 zeigt die verschiedenen Teilnehmer und welche Smart Contracts von diesen veröffentlicht werden sollen.

Teilnehmer 1	**Teilnehmer 2**	**Teilnehmer 3**	**Teilnehmer 4**
VSC, RSC1	OSC, RSC2, RSC3	ASC, GSC, RSC4	LSC

Tabelle 7.3: Teilnehmer und ihre Smart Contracts

7.2 Validierung der Implementierung

Die Implementierung des Designs wird mit dem vorangestellten Szenario validiert. Die Ressourcen wurden so gewählt, dass es eine eindeutige Zuordnung durch den OSC gibt. Damit die Validierung erfolgreich ist, werden folgende Ergebnisse erwartet.

- Die entsprechenden Ressourcen erhalten eine Auftragsbenachrichtigung mit einer Sequenznummer welche die Reihenfolge festlegt.
- Die Ressourcen erhalten einen autorisierten Zugriff auf die CAD-Datei.
- Die Ressource für die Logistik erhält eine Auftragsbenachrichtigung und antwortet mit einer Referenz die eine Auftragsnummer repräsentiert.
- Die hochgeladene CAD-Datei darf nicht von unautorisierten Teilnehmern geöffnet werden.
- Sensible Daten wie die Adresse des Kunden sind nicht auf der Blockchain abgelegt worden und können daher nicht angezeigt werden.

Das hier vorgestellte Szenario konnte allerdings nicht mit den Smart Contracts und Solidity realisiert und validiert werden. Zwar konnten die Smart Contracts Daten austauschen, aber die korrekte Verarbeitung der Daten konnte nicht bestätigt werden. So konnte der OSC alle Ressourcen und deren Fähigkeiten nach dem übermitteln eines Auftrags durch den ASC abrufen. Der implementierte Algorithmus, um die Zuordnung der Aufgabenbestandteile zu den Ressourcen zu ermöglichen, war zum Zeitpunkt der Abgabe nicht ausgereift.

Auch wird zwar der Hashwert als Referenz, für die auf IPFS abgelegte Datei, bis zur Ressource übergeben werden, doch eine Verschlüsselung der Daten nativ von IPFS ist derzeit nicht möglich. Zum Zeitpunkt der Implementierung befand sich die Entwicklung dieser Funktion in der Phase *Work in progress* [1].

Allerdings konnte festgestellt werden, dass die Kundendaten, die mit dem Auftrag an den OSC übergeben worden sind, nicht auf der Blockchain abgelegt wurden. Diese wurden für die Verarbeitung im Speicher vorgehalten und wurden nach dem Aufruf der Methode wieder verworfen.

7.3 Diskussion

Wie bereits geschildert, konnte die Implementierung des Konzepts nicht gegen das Szenario validiert werden. Zwar hat Solidity große Ähnlichkeiten zu Java und JavaScript, das Konzept dahinter ist aber ausgelegt auf die Verarbeitung von Hashwerten. Das wurde deutlich bei den Vergleichen der jeweiligen Datenmodelle, da diese nicht direkt miteinander verglichen werden können. Die Werte müssen als Hashwerte übereinstimmen. Das macht es gerade für komplexe Datenstrukturen schwierig.

Weiterhin war eine Serialisierung von Datenstrukturen in Solidity nicht möglich. Diese Funktionalität ist zwar von den Entwicklern angekündigt, wird auch von einen experimentellen Compiler unterstützt, wobei es für Objekte aus unterschiedlichen Datentypen wie strings und uints nicht funktionierte.

Wie auch bereits beschrieben konnten Dateien, die auf dem IPFS abgelegt worden, sind nicht bei Bedarf wieder neu verschlüsselt werden, um so einen autorisierten Zugriff zu ermöglichen. Dies konnte aber umgangen werden, indem die Datei mit GnuPG[2] zunächst

[1] https://github.com/ipfs/specs/tree/master/keystore
[2] https://www.gnupg.org/

verschlüsselt worden ist, bevor sie auf IPFS abgelegt wurde. Das erfordert aber bereits im Vorfeld den öffentlichen Schlüssel der Ressource, die die Datei abrufen möchte. Dieses Verhalten könnte aber automatisiert durch den GSC erfolgen, wobei das nicht validiert wurde.

Abschließend soll auf die Forschungsfrage eingegangen werden, wie im Kontext des kollaborativen Cloud Manufacturing eine Orchestrations- und Planungsplattform konzipiert sein muss. Das Konzept aus Kapitel 5 kann als Grundlage für so eine Plattform verwendet werden. Die Blockchain Technologie bietet dafür die notwendigen Voraussetzungen. Zwar konnte bei der Validierung nur die Orchestrierung von vier Smart Contracts als Services nachgewiesen werden, doch das beruht auf der Tatsache, dass die Kenntnisse von Solidity und die Sprache selbst für die Implementierung noch nicht ausgereift waren. Ebenfalls konnte die Planung nur für ein sehr vereinfachtes Szenario gewährleistet werden. Dies kann aber mit einen ausgereiften Algorithmus durchaus über Smart Contracts erfolgen. Allerdings bietet Solidity auch die Möglichkeit Variablen und Datenstrukturen nur im Speicher vorzuhalten. Diese Funktion dient bei Ethereum in erster Linie dazu, teure Transaktionen auf der Blockchain zu vermeiden. Das kann aber dafür verwendet werden, dass sensible Daten nur für Berechnungen herangezogen und dann wieder verworfen werden.

Kapitel 8

Verwandte Arbeiten

Dieser Teil der Arbeit soll einen Überblick über verwandte Arbeiten geben.

Barenji et al. stellen ein Konzept für das CM mit dem Einsatz der Blockchain Technologie vor [66]. Sie stellen eine zweigeteilte Architektur vor. Der erste Teil ist für den Endbenutzer, der damit mit dem Gesamtsystem interagieren kann. Der zweite Teil besteht aus drei Schichten. Der Core Layer ist die Management Schicht und bietet Services für beispielsweise das Monitoring sowie die Sicherheit. Die Public Blockchain Network Schicht ist ein öffentliches Blockchain Netzwerk. Es dient zur Kommunikation zwischen den Providern und den Endbenutzern. Der Cloud Manufacturing Provider Layer ist eine Schicht die in weitere vier Schichten aufgeteilt wird. So ist ein Bestandteil eine private Blockchain, welche die Kommunikation innerhalb der Produktionsmaschinen ermöglichen soll. Weiterhin werden hier die Big Data Analysen verortet.

Eine bereits im Kapitel 3 vorgestellte Arbeit ist die BPIIoT Plattform von Bahga et al. [50]. Das Konzept verfolgt den Einsatz der Blockchain für die Kommunikation von Maschine-zu-Maschine über Smart Contracts. In der Blockchain stellt eine Maschine einen Knoten als Teilnehmer dar. Kunden können an die Maschinen direkt Aufträge senden, die von den diesen verarbeitet werden. Über die Maschine-zu-Maschine können dann die Aufträge weitergeleitet werden, wenn der Auftrag nicht erfüllt werden kann. Dafür können auch Zahlungen zwischen den Maschinen vorgenommen werden. Als ein Vorteil dieses Konzept führen die Autoren an, dass es bei diesem dezentralen Netzwerk keine zentrale Instanz gibt, die die Plattform regulieren kann.

Kapitel 9

Zusammenfassung und Ausblick

Das CM bietet für Kunden, Hersteller sowie Dienstleister großes Potential um das gesamte SCM flexibel zu gestalten. Fertigungsunternehmen können miteinander in einer Weise kollaborieren, die bislang noch nicht möglich war. Für diesen Zweck wird eine Plattform benötigt, über die Kunden, Hersteller sowie Dienstleister miteinander transparent als Partner interagieren können. Transparenz darf aber in diesem Fall nicht das Aufheben von Mechanismen für den Schutz oder die Sicherheit von Daten bedeuten.

Der Einsatz einer zentralen Plattform setzt aber auch das Vertrauen der Teilnehmer in diese voraus. Wissen, sensible Daten und geistiges Eigentum wird mit und über diese Plattform ausgetauscht. Somit ist der Betreiber der Plattform in der Lage, alle ausgetauschten Daten und Informationen einzusehen. Weiterhin stellt ein zentrales System einen Single Point of Failure dar. Fällt dieses System aus, ist eine Zusammenarbeit zwischen den einzelnen Teilnehmern über die Plattform nicht mehr möglich.

Um ein Konzept für die Beantwortung der Forschungsfrage zu erarbeiten, wurde zunächst eine Analyse von bereits vorhandenen Ansätzen durchgeführt. Hierfür wurden vier wissenschaftliche Publikationen, die aus dem Stand der Technik ermittelt werden konnten, analysiert. Mit definierten Kriterien wurde eine Bewertung der jeweiligen Ansätze vorgenommen. Der Stand der Technik sowie das Resultat aus der Analyse, wurden für die Ausarbeitung von Anforderungen an eine CM-Plattform verwendet. Anschließend wurde das Konzept prototypisch implementiert und Szenario ausgearbeitet, um die Implementierung zu validieren.

9.1 Ausblick

Wie im Kapitel 7.3 erörtert, konnten mehrere Probleme bei der Validierung des Konzepts ermittelt werden. Dies hängt aber nicht zwangsläufig von dem Konzept selbst ab, sondern von der Implementierung der Smart Contracts in Solidity. So könnte eine zukünftige Arbeit sich damit befassen, effiziente Algorithmen für den Einsatz auf der Ethereum Blockchain zu finden, die eine Teilung und Planung von Aufträgen ermöglichen. Bei der Validierung des Szenarios wurde auch festgestellt, dass die Laufzeit der Methodenaufrufe bei ca. sieben Sekunden liegt. Eine Ursache kann ein ineffizienter Methodenaufruf bei der Kommunikation zwischen den Smart Contracts sein. So konnte festgestellt werden, dass ein Aufruf mit nur einen Rückgabewert etwa fünf Sekunden benötigt. Das kann auf der Konsensmechanismus zurückgeführt werden, der etwa vier Sekunden benötigt um einen Block zu validieren. Auch könnte eine interessante Möglichkeit der Einsatz von Hyperledger Fabric und Chaincode als Smart Contracts sein. Hyperledger unterstützt bereits die Möglichkeit, dass Rechte vergeben können und nicht alle Teilnehmer Einsicht auf bestimmte Daten haben. Zwar besteht immer noch das Risiko, dass eine Autorität dieses Netzwerk verwaltet, aber dafür müsste Hyperledger näher untersucht werden.

Ein Faktor auf den das Konzept nicht eingegangen ist, ist die Kostenabwicklung. Über Ethereum können auch Zahlungen vorgenommen werden, die von den Hersteller angefordert werden könne, auf der Basis der genutzten Ressourcen. Dieses geschlossene Ecosystem mit einen Waren-, Dienstleistungs-, und Zahlungsfluss könnte ein weiteres Potenzial der Blockchain Technologie aufzeigen. So kann der Operator, für die Nutzung seines Services für die Planung, ebenfalls in Form von der Blockchain-Währung bezahlt werden.

Im Zusammenhang mit den Kosten, wurden die implementierten Smart Contracts nicht darauf hin untersucht, welche Kosten tatsächlich durch die Veröffentlichung sowie die notwendigen Transaktionen anfallen. Das hängt aber maßgeblich von der Effizienz der eingesetzten Algorithmen in den Smart Contracts ab und müsste darauf hin überprüft werden.

Weiterhin können für die Zuordnung und Planung Ontologien verwendet werden. Eine interessante Arbeit als Einstiegspunkt ist von Kim und Laskowski [67]. Sie verwenden die Ethereum Blockchain für die Nachverfolgung im SCM. Es wird die Möglichkeit erörtert Ontologien in Smart Contracts zu implementieren.

Literatur

[1] S. Mittal, M. A. Khan, D. Romero und T. Wuest, „Smart manufacturing: Characteristics, technologies and enabling factors“, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, Jg. 65, 2017.

[2] B. Andres, R. Sanchis und R. Poler, „A Cloud Platform to support Collaboration in Supply Networks“, *International Journal of Production Management and Engineering*, Jg. 4, Nr. 1, S. 5, 2016.

[3] M. Hermann, T. Pentek und B. Otto, *Design Principles for Industrie 4.0 Scenarios: A Literature Review*, 2015.

[4] X. Xu, „From cloud computing to cloud manufacturing“, *Robotics and Computer-Integrated Manufacturing*, Jg. 28, Nr. 1, S. 75–86, 2012.

[5] L. Thames und D. Schaefer, „Software-defined Cloud Manufacturing for Industry 4.0“, *Procedia CIRP*, Jg. 52, S. 12–17, 2016.

[6] C. Yu, X. Xu und Y. Lu, „Computer-Integrated Manufacturing, Cyber-Physical Systems and Cloud Manufacturing – Concepts and relationships“, *Manufacturing Letters*, Jg. 6, S. 5–9, 2015.

[7] S. Subashini und V. Kavitha, „A survey on security issues in service delivery models of cloud computing“, *Journal of Network and Computer Applications*, Jg. 34, Nr. 1, S. 1–11, 2011.

[8] L. Ren, L. Zhang, F. Tao, C. Zhao, X. Chai und X. Zhao, „Cloud manufacturing: from concept to practice“, *Enterprise Information Systems*, Jg. 9, Nr. 2, S. 186–209, 2015.

[9] C. Wohlin, „Guidelines for snowballing in systematic literature studies and a replication in software engineering“, in *EASE 2014*, M. Shepperd, T. Hall und I. Myrtveit, Hrsg., Ser. ICPS, New York, New York: Association for Computing Machinery, 2014, S. 1–10.

[10] J. Delaram und O. F. Valilai, „Development of a Novel Solution to Enable Integration and Interoperability for Cloud Manufacturing“, *Procedia CIRP*, Jg. 52, S. 6–11, 2016.

[11] Y. Liu, L. Wang, Y. Wang, X. V. Wang und L. Zhang, „Multi-agent-based scheduling in cloud manufacturing with dynamic task arrivals“, *Procedia CIRP*, Jg. 72, S. 953–960, 2018.

[12] E. Rauch, P. Dallasega und D. T. Matt, „Distributed manufacturing network models of smart and agile mini-factories“, *International Journal of Agile Systems and Management*, Jg. 10, Nr. 3/4, S. 185, 2017.

[13] E. Rauch, S. Seidenstricker, P. Dallasega und R. Hämmerl, „Collaborative Cloud Manufacturing: Design of Business Model Innovations Enabled by Cyberphysical Systems in Distributed Manufacturing Systems“, *Journal of Engineering*, Jg. 2016, Nr. 3, S. 1–12, 2016.

[14] P. Zheng, H. wang, Z. Sang, R. Y. Zhong, Y. Liu, C. Liu, K. Mubarok, S. Yu und X. Xu, „Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives“, *Frontiers of Mechanical Engineering*, Jg. 13, Nr. 2, S. 137–150, 2018.

[15] C. Fehling, F. Leymann, R. Retter, W. Schupeck und P. Arbitter, *Cloud computing patterns : fundamentals to design, build, and manage cloud applications*. Wien und Heidelberg [u.a.]: Springer, 2014.

[16] P. M. Mell und T. Grance, *The NIST definition of cloud computing*. Gaithersburg, MD: National Institute of Standards and Technology, 2011.

[17] G. Adamson, L. Wang, M. Holm und P. Moore, „Cloud manufacturing – a critical review of recent development and future trends“, *International Journal of Computer Integrated Manufacturing*, Jg. 1, Nr. 1, S. 1–34, 2015.

[18] J. Siderska und K. S. Jadaan, „Cloud manufacturing: a service-oriented manufacturing paradigm. A review paper“, *Engineering Management in Production and Services*, Jg. 10, Nr. 1, S. 22–31, 2018.

[19] Y. Lu, J. Xu und X. Xu, „A New Paradigm Shift for Manufacturing Businesses“, in *Volume 11: Emerging Technologies*, ASME, 2013.

[20] F. Ning, W. Zhou, F. Zhang, Q. Yin und X. Ni, „The architecture of cloud maufacturing and its key technologies research“, in *IEEE International Conference on Cloud Computing and Intelligence Systems (CCIS), 2011*, D. Li, Hrsg., Piscataway, NJ: IEEE, 2011, S. 259–263.

[21] Y. Liu, L. Wang, X. V. Wang, X. Xu und L. Zhang, „Scheduling in cloud manufacturing: state-of-the-art and research challenges“, *International Journal of Production Research*, Jg. 30, Nr. 4, S. 1–26, 2018.

[22] D. Wu, J. L. Thames, D. W. Rosen und D. Schaefer, „Towards a Cloud-Based Design and Manufacturing Paradigm: Looking Backward, Looking Forward“, in *Volume 2: 32nd Computers and Information in Engineering Conference, Parts A and B*, ASME, 2012, S. 315–328.

[23] i. Liu und H. Jiang, „Research on key technologies for design services collaboration in cloud manufacturing“, in *IEEE 16th International Conference on Computer Supported Cooperative Work in Design (CSCWD), 2012*, L. Gao, Hrsg., Piscataway, NJ: IEEE, 2012, S. 824–829.

[24] F. Tao, Y. Zuo, L. D. Xu und L. Zhang, „IoT-Based Intelligent Perception and Access of Manufacturing Resource Toward Cloud Manufacturing“, *IEEE Transactions on Industrial Informatics*, Jg. 10, Nr. 2, S. 1547–1557, 2014.

[25] S. Nakamoto, „Bitcoin: A Peer-to-Peer Electronic Cash System“, *Cryptography Mailing list at https://metzdowd.com*, 2009.

[26] K. Christidis und M. Devetsikiotis, „Blockchains and Smart Contracts for the Internet of Things“, *IEEE Access*, Jg. 4, S. 2292–2303, 2016.

[27] S. Singh und N. Singh, „Blockchain: Future of financial and cyber security“, in *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, IEEE, 2016, S. 463–467.

[28] F. Tschorsch und B. Scheuermann, „Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies“, *IEEE Communications Surveys & Tutorials*, Jg. 18, Nr. 3, S. 2084–2123, 2016.

[29] I. Karamitsos, M. Papadaki und N. B. A. Barghuthi, „Design of the Blockchain Smart Contract: A Use Case for Real Estate“, *Journal of Information Security*, Jg. 09, Nr. 03, S. 177–190, 2018.

[30] Z. Zheng, S. Xie, H. Dai, X. Chen und H. Wang, „An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends“, in *2017 IEEE International Congress on Big Data (BigData Congress)*, IEEE, 2017, S. 557–564.

[31] X. V. Wang, M. Givehchi und L. Wang, „Manufacturing System on the Cloud: A Case Study on Cloud-based Process Planning“, *Procedia CIRP*, Jg. 63, S. 39–45, 2017.

[32] C. Esposito, A. Castiglione, B. Martini und K.-K. R. Choo, „Cloud Manufacturing: Security, Privacy, and Forensic Concerns“, *IEEE Cloud Computing*, Jg. 3, Nr. 4, S. 16–22, 2016.

[33] F. Tao, L. Zhang, Y. Liu, Y. Cheng, L. Wang und X. Xu, „Manufacturing Service Management in Cloud Manufacturing: Overview and Future Research Directions“, *Journal of Manufacturing Science and Engineering*, Jg. 137, Nr. 4, S. 040 912, 2015.

[34] Y. Liu, X. Xu, L. Zhang, L. Wang und R. Y. Zhong, „Workload-based multi-task scheduling in cloud manufacturing“, *Robotics and Computer-Integrated Manufacturing*, Jg. 45, S. 3–20, 2017.

[35] Y. Lu, H. Wang und X. Xu, „ManuService ontology: a product data model for service-oriented business interactions in a cloud manufacturing environment“, *Journal of Intelligent Manufacturing*, Jg. 284, Nr. 5, S. 34, 2016.

[36] J. Xinjuan und L. Quan, „Research on the On-demand Service mode in cloud manufacturing“, in *Proceedings of 2016 IEEE 7th International Conference on Software Engineering and Service Science*, M. S. P. Babu, Hrsg., Piscataway, NJ: IEEE, 2016, S. 285–288.

[37] X. Wang, S. K. Ong und A. Nee, „A comprehensive survey of ubiquitous manufacturing research“, *International Journal of Production Research*, Jg. 56, Nr. 1-2, S. 604–628, 2018.

[38] F. Tao, L. Zhang, V. C. Venkatesh, Y. Luo und Y. Cheng, „Cloud manufacturing: a computing and service-oriented manufacturing model“, *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, Jg. 225, Nr. 10, S. 1969–1976, 2011.

[39] L. Zhang, Y. Luo, F. Tao, B. H. Li, L. Ren, X. Zhang, H. Guo, Y. Cheng, A. Hu und Y. Liu, „Cloud manufacturing: a new manufacturing paradigm“, *Enterprise Information Systems*, Jg. 8, Nr. 2, S. 167–187, 2014.

[40] Y. Y. Zhao, Q. Liu, W. J. Xu und L. Gao, „Modeling of Resources Capability for Manufacturing Equipments in Cloud Manufacturing“, *Applied Mechanics and Materials*, Jg. 271-272, S. 447–451, 2012.

[41] B. Huang, C. Li, C. Yin und X. Zhao, „Cloud manufacturing service platform for small- and medium-sized enterprises“, *The International Journal of Advanced Manufacturing Technology*, Jg. 65, Nr. 9-12, S. 1261–1272, 2013.

[42] P. Helo, M. Suorsa, Y. Hao und P. Anussornnitisarn, „Toward a cloud-based manufacturing execution system for distributed manufacturing“, *Computers in Industry*, Jg. 65, Nr. 4, S. 646–656, 2014.

[43] O. F. Valilai und M. Houshmand, „A collaborative and integrated platform to support distributed manufacturing system using a service-oriented approach based on cloud computing paradigm“, *Robotics and Computer-Integrated Manufacturing*, Jg. 29, Nr. 1, S. 110–127, 2013.

[44] Y. Liu, L. Wang und X. Vincent Wang, „Cloud manufacturing: latest advancements and future trends“, *Procedia Manufacturing*, Jg. 25, S. 62–73, 2018.

[45] D. Wu, D. W. Rosen, L. Wang und D. Schaefer, „Cloud-based design and manufacturing: A new paradigm in digital manufacturing and design innovation“, *Computer-Aided Design*, Jg. 59, S. 1–14, 2015.

[46] L. Gao, Q. Liu und P. Lou, „Computational Trust in Cloud Manufacturing“, *Advanced Materials Research*, Jg. 774-776, S. 1908–1913, 2013.

[47] X. Cai, W. Li, F. He und X. Li, „Customized Encryption of Computer Aided Design Models for Collaboration in Cloud Manufacturing Environment“, *Journal of Manufacturing Science and Engineering*, Jg. 137, Nr. 4, S. 040 905, 2015.

[48] Z. Li, L. Liu, A. V. Barenji und W. Wang, „Cloud-based Manufacturing Blockchain: Secure Knowledge Sharing for Injection Mould Redesign“, *Procedia CIRP*, Jg. 72, S. 961–966, 2018.

[49] A. Angrish, B. Craver, M. Hasan und B. Starly, „A Case Study for Blockchain in Manufacturing: “FabRec”: A Prototype for Peer-to-Peer Network of Manufacturing Nodes“, *Procedia Manufacturing*, Jg. 26, S. 1180–1192, 2018.

[50] A. Bahga und V. K. Madisetti, „Blockchain Platform for Industrial Internet of Things“, *Journal of Software Engineering and Applications*, Jg. 09, Nr. 10, S. 533–546, 2016.

[51] X. F. Liu, M. R. Shahriar, S. N. Al Sunny, M. C. Leu und L. Hu, „Cyber-physical manufacturing cloud: Architecture, virtualization, communication, and testbed“, *Journal of Manufacturing Systems*, Jg. 43, S. 352–364, 2017.

[52] R. Roman, J. Zhou und J. Lopez, „On the features and challenges of security and privacy in distributed internet of things“, *Computer Networks*, Jg. 57, Nr. 10, S. 2266–2279, 2013.

[53] X. Lu, Q. Li, Z. Qu und P. Hui, „Privacy Information Security Classification Study in Internet of Things“, in *2014 International Conference on Identification, Information and Knowledge in the Internet of Things, IIKI 2014*, Piscataway, NJ: IEEE, 2014, S. 162–165.

[54] Y. Sun, J. Zhang, Y. Xiong und G. Zhu, „Data Security and Privacy in Cloud Computing“, *International Journal of Distributed Sensor Networks*, Jg. 10, Nr. 7, S. 190 903, 2014.

[55] S. R. Chhetri, S. Faezi, N. Rashid und M. A. Al Faruque, „Manufacturing Supply Chain and Product Lifecycle Security in the Era of Industry 4.0“, *Journal of Hardware and Systems Security*, Jg. 2, Nr. 1, S. 51–68, 2018.

[56] Y. Lu und X. Xu, „Protecting Intellectual Property in a Cloud Manufacturing Environment: Requirements and Strategies“, in *Advances in Production Management Systems: Innovative Production Management Towards Sustainable Growth*, Ser. IFIP advances in information and communication technology, S. Umeda, M. Nakano, H. Mizuyama, H. Hibino, D. Kiritsis und G. von Cieminski, Hrsg., Bd. 460, Cham: Springer International Publishing, 2015, S. 404–411.

[57] A. Reyna, C. Martín, J. Chen, E. Soler und M. Díaz, „On blockchain and its integration with IoT. Challenges and opportunities“, *Future Generation Computer Systems*, Jg. 88, S. 173–190, 2018.

[58] Ethereum Foundation, *Ethereum Project*, 2018. URL: `https://www.ethereum.org/` (besucht am 27. 12. 2018).

[59] Ethereum, *Geth*, 2018. URL: `https://github.com/ethereum/go-ethereum/wiki/geth` (besucht am 28. 12. 2018).

[60] The Linux Foundation, *Hyperledger Fabric*, 2018. URL: `https://hyperledger-fabric.readthedocs.io/en/release-1.4/` (besucht am 27. 12. 2018).

[61] IOTA Foundation, *IOTA*, 2018. URL: `https://docs.iota.org/introduction/what-is-iota` (besucht am 27. 12. 2018).

[62] Ethersphere, *Swarm*, 2018. URL: https://swarm-guide.readthedocs.io/en/latest/index.html# (besucht am 29. 12. 2018).

[63] Protocol Labs, *IPFS*, 2018. URL: https://docs.ipfs.io/ (besucht am 29. 12. 2018).

[64] Consensys, *Truffle*, 2018. URL: https://truffleframework.com/ (besucht am 30. 12. 2018).

[65] Matthias Strljic, *Fertigungsauftrag*, inte, Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW), 2019.

[66] A. V. Barenji, Z. Li und W. M. Wang, „Blockchain Cloud Manufacturing: Shop Floor and Machine Level“, in *Smart SysTech 2018; European Conference on Smart Objects, Systems and Technologies*, 2018, S. 1–6.

[67] H. M. Kim und M. Laskowski, „Toward an ontology-driven blockchain design for supply-chain provenance“, *Intelligent Systems in Accounting, Finance and Management*, Jg. 25, Nr. 1, S. 18–27, 2018.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift