Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Till Vieth

Cloudbasierte Produktionssteuerung

Fakultät Technik und Informatik
Department Maschinenbau und Produktion

Faculty of Engineering and Computer Science
Department of Mechanical Engineering and Production Management

Till Vieth

Cloudbasierte Produktionssteuerung

Masterarbeit eingereicht im Rahmen des Masterstudiums

am Department Maschinenbau und Produktion
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Erstprüfer: Prof. Dr.-Ing. Randolf Isenberg
Zweitprüfer: Dr.-Ing Roland Schröder-Kroll

Abgabedatum: 10.09.2021

Zusammenfassung

Till Vieth

Titel der Masterarbeit

Cloudbasierte Produktionssteuerung

Stichworte

Industrie 4.0, Internet der Dinge, MQTT, Shop Floor Management, Smart Manufacturing, Cloud Computing

Kurzzusammenfassung

Diese Arbeit umfasst das Konzept eines cyberphysischen Systems zur cloudbasierten Steuerung von Produktionsdaten und -informationen. Hierbei wird verfolgt, dass die Produktionsdaten von ausgewählten Produktionsmitteln transparent zur Verfügung gestellt und Reaktionswege verkürzt werden sollen. Dies soll Stillstandzeiten der Produktionsmittel verkürzen und Entscheidungskompetenzen im Shopfloor Management verbessern. Weiterhin wird zu diesem Konzept eine Fallstudie entworfen, die ein solches Produktionsszenario vereinfacht darstellen und abdecken soll. Zur Realisierung dieses Konzeptes spielt die Industrie 4.0 eine große Rolle, da hierdurch neue Möglichkeiten wie „Internet of Things“ und „Cloud Computing“ geschaffen werden.

Till Vieth

Title of the paper

Cloudbased production control

Keywords

Industry 4.0, Internet of Things, MQTT, Shop Floor Management, Smart Manufacturing, Cloud Computing

Abstract

This thesis includes the concept of a cyber-physical system for cloud-based control of production data and information. It is aiming on a transparent and available production data of selected production equipment and shorten reaction paths. This should reduce the downtime of machines and also should improve the shop floor management’s decision-making process. Furthermore, a case study is designed for this concept, which should represent such a production scenario in a simplified way. In this case, Industry 4.0 plays a major role to realize this concept, because it creates new possibilities like “internet of things” or “cloud computing”.

Inhaltsverzeichnis

Abkürzungsverzeichnis

CPS	Cyber-Physical Systems/ Cyber-Physikalische Systeme
KI	künstliche Intelligenz
I4.0	Industrie 4.0 (vierte industrielle Revolution)
IDE	Integrierte Entwicklungsumgebung
IKT	Informations- und Kommunikationstechnik
IoT	Internet of Things/ Internet der Dinge
IT	Informationstechnologie
M2M	Machine to Machine (Kommunikationsweg)
OPC UA	Open Platform Communications United Architecture
SFM	Shopfloor Management

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Problemstellung

Die traditionelle Produktion ist durch die Industrie 4.0 (I4.0) zu einem ökonomischen und digitalen System transformiert. Hierbei werden unter anderem die Schlagwörter „Internet of Things“ und „Cyber-physical Systems“ verwendet, welche neue Horizonte der Kommunikation und automatischen Prozessen eröffnen sollen, die in der Vergangenheit nicht zu erreichen waren. (Mourtzis/Vlachou 2018). Weiter entsteht im digitalen Zeitalter von I4.0, die Digitalisierung des Shop Floors konkret der direkten Produktion bzw. Fertigung. Dabei wird das Schlagwort „Smart Manufacturing“ häufig verwendet, da diese die heutige Produktion zur erweiterten Digitalisierung und Automatisierung führen soll. Dies unterstreicht die Bedeutung von „Big Data“ und die Nutzung dieser Daten auf smarte Weise durch digitale Technologien. Diese Entwicklung beeinflusst das „Shop Floor Management“ und zielt auf die Nutzung von Echtzeitdaten ab, durch den Einsatz von technologischen Tools wie Computerplattformen und Kommunikationstechnologien wie der Cloud beispielsweise (Clausen et al. 2020).

In diesem Zusammenhang sollen in dieser Masterarbeit die Digitalisierung und Kommunikationsmöglichkeiten im Shop Floor bzw. in der Fertigung betrachtet werden, da diese Vorteile wie Transparenz, Reaktionsfähigkeiten durch Echtzeitdaten, Entscheidungskompetenzen und neue Kommunikationswege schaffen, die in der Vergangenheit nicht möglich waren (Clausen et al. 2020).

1.2 Motivation

Damit Unternehmen weiterhin wettbewerbsfähig bleiben und zu einer perfekten Produktion gelangen müssen die Zielgrößen des „magischen Dreiecks“ Kosten, Qualität und Zeit konsequent verfolgt und im Optimum zueinanderstehen. Dies ist aber nicht trivial, da sich die Zielgrößen untereinander beeinflussen. Wird beispielsweise die Zielgröße Zeit verfolgt, so wird versucht Durchlaufzeiten zu verringern, um früher das Produkt auf den Markt zu bringen oder dem Kunden auszuliefern. Auf der anderen Seite benötigt Qualität Zeit, da zur Qualitätskontrolle eine Produktionskettenerweiterung

notwendig ist, da Kontrollstationen zur Überwachung der Qualität gefordert werden. (Kletti/Schumacher 2014)

Zurzeit konzentrieren sich praktisch alle Unternehmen auf die Kostenreduzierung, dabei kennen die Unternehmen ihre Produktionskosten und Qualität, doch der Faktor Zeit ist häufig unbekannt und dadurch auch keine verfolgte Zielgröße (Kletti/Schumacher 2014).

Trotzdem sind im Kontext der Zielgröße Zeit große Wirtschaftlichkeitspotenziale zu entdecken, wie man durch einzelne Szenarien aus der Produktion erkennen kann: (Kletti/Schumacher 2014)

- Sobald eine Maschine stillsteht und dies nicht sofort erkannt wird steigen die Auftragskosten, da mehr Zeit für den Auftrag benötigt wird, genauso steigen auf die Personalkosten, da die Maschinenbediener in dieser Zeit nicht weiterarbeiten können.
- Das obige Szenario lässt sich auch auf die Qualität ableiten, denn sobald Ausschuss entsteht und dieser nicht sofort entdeckt wird, entstehen möglicherweise Nacharbeiten, um die gesamte Menge wieder aufzuholen. Dies erhöht nicht nur die Kosten, sondern auf die Zeit des jeweiligen Auftrags.
- Um dies nun entgegenzuwirken, werden Prüfstationen in den Prozess eingebunden, dadurch erhöhen sich sowohl die Zeit wie auch die Kosten.

Daher sollten die bisher bekannten Zielgrößen Qualität, Kosten und Zeit um zwei weitere ergänzt werden: Die Reaktionsfähigkeit und Transparenz eines Unternehmens. Damit Abweichungen vom Sollzustand schneller erkannt und schneller auf spontane Ereignisse wie Stillstände oder auch Eilaufträge reagieren werden kann (Kletti/Schumacher 2014). Um in diesem Zusammenhang Vorhersagen über mögliche Maschinenausfälle zu prognostizieren, ist es erforderlich, alle Informationen zu nutzen, wie unter anderem Temperatur, Vibrationen, Maschinenstandort, Maschinenhistorie/-laufzeit. Daraus resultierend können Maschinen-/ Anlagenverfügbarkeiten erhöht werden (Reinheimer 2017).

Die I4.0 Bewegung bringt auch eine digitale Transformation des Shop Floor Managements mit sich. Dies soll Abhilfe schaffen, denn durch die Digitalisierung werden Möglichkeiten geschaffen wie beispielsweise eine gesamte Datentransparenz, um so die

Wettbewerbssituation sowohl kurz- als auch langfristig zu verbessern. Dies wird zum Beispiel durch Echtzeit-Daten-Übertragung, Big Data und auch künstlicher Intelligenz (KI) erreicht (Clausen et al. 2020).

In den meisten der existierenden Produktionen werden nicht erreichbare Terminplanungen erarbeitet, da sie den genauen Zustand der Fertigung nicht kennen und nicht adaptiv planen können. Cloudbasierte Informationssysteme sollen in diesem Zuge Möglichkeiten schaffen, den Zustand der Produktion transparent und in Echtzeit zu vermitteln, um daran flexible und anpassungsfähige Planungen durchführen zu können (Mourtzis/Vlachou 2018).

Auch „Cyber-Physikalische Systeme“ (CPS) können einen Anteil daran haben, das Unternehmen ihre Wettbewerbssituationen ausbauen können, die in ein solches Datenmanagement wie Cloudsysteme angebunden werden können. Durch:

> „Verschmelzung von Informationstechnologien (IT) und Steuerungsaufgaben sind neuartige Steuerungsarchitekturen möglich und sinnvoll. Mit den bestehenden starren Steuerungsarchitekturen und -Systemen lassen sich innovative Konfigurations-, Planungs- und Optimierungsalgorithmen nicht umsetzen“ (Lechler/Schlechtendahl 2017).

1.3 Vorgehensweise

Hierzu soll der Stand der Technik erläutert und darauf basierend, ein schematisches Konzept der cloudbasierten Steuerung im Bereich der Fertigung entwickelt werden. Deshalb soll im Rahmen dieser Masterarbeit Recherche zu den Themen Shop Floor Management und daran angelehnt Schnittstellen der Produktionskommunikation und Möglichkeiten durch die Industrie 4.0 in Bezug auf Smart Manufacturing, Cloud Computing und Internet of Things durchgeführt werden.

Zur Veranschaulichung der Digitalisierung und die damit einhergehenden Kommunikationstechnologien im Kontext einer industriellen Fertigung soll eine Fallstudie entwickelt werden, die vereinfachte Produktionsszenarien darstellen soll, welche die Möglichkeiten der Industrie 4.0 aufzeigen soll, wie beispielsweise eine cloudbasierte Steuerung.

Daraus soll unteranderem hervorgehen, was für unterschiedliche Daten gehandhabt werden können und welche möglichen Daten für eine webbasierte Steuerung geeignet sind. Für die Nutzungen einer solche webbasierten Steuerung sollen Sicherheitshinweise betrachtet und Empfehlungen ausgesprochen werden.

2 Theoretische Grundlagen

In diesem Kapitel werden theoretischen Grundlagen vorgestellt, die zur Erläuterung des entwickelten Konzeptes beitragen. Die Grundlagen sind im Folgenden in allgemeine und technische Grundlagen aufgeteilt.

2.1 Allgemeine Grundlagen

Dieses Unterkapitel soll den Leser einen Überblick über die allgemeinen Grundlagen zu dieser Masterarbeit verschaffen. Beginnend wird die Entwicklung bis hin zur Industrie 4.0 erläutert und die daraus entstandenen Schlüsselwörter vorgestellt. Abschließend werden zwei Projekte zum Thema Industrie 4.0 vorgestellt, die im Kontext zu dem entwickelten Konzept dieser Masterarbeit stehen.

2.1.1 Industrie 4.0 Begriffsklärung und Zusammenhänge

Der Begriff „Industrie 4.0“ stammt von der Deutschen Bundesregierung (Hüning 2019) und wurde zunächst auf der Hannover-Messe im Jahr 2011 verwendet. Hierdurch hat der Begriff in der theoretischen wissenschaftlichen Durchdringung als auch in der Praxis zunehmend an Bedeutung erlangt. Zu den Anwendungsgebieten der I4.0 zählen unter anderem Digitalisierungstechnologien im Umfeld der Produktion bzw. Industrie. (Steven 2019)

I4.0 steht für die vierte industrielle Revolution und ist ein Sammelbegriff von Wertschöpfungsprozessen, die durch Konzepte und Technologien gesteuert und organisiert werden. Zu solchen Technologien gehören unter anderem cyberphysische Systeme, die eingesetzt werden, um eine transparente und durchgehende Kommunikation zu schaffen, die auch bis zum Menschen weitergeleitet werden soll und das in Echtzeit über das Internet der Dinge. Des Weiteren schafft die neu erworbene Kommunikationsmöglichkeit durch das Internet der Dinge bzw. Dienste (Internet of Things), sowohl die Integration aller beteiligten organisatorischen Instanzen am Wertschöpfungsprozess und dadurch können Entscheidungen dezentral getroffen werden. Dabei richtet

sich die Organisation zunehmend an individuellen Kundenwünschen aus. (Steven 2019)

Näherungsweise lässt sich I4.0 als Kehrtwende der Durchführung von Wertschöpfungsprozessen beschreiben, wobei die damit veränderten Prozesse über alle Instanzen verknüpft sind und auch der Kunde mit in den Wertschöpfungsprozess eingebunden wird (Steven 2019).

Zum Verständnis der Entstehung des Begriffes „Industrie 4.0“ ist auf die Vorgeschichte der Industrie zu schauen und Meilensteine hervorzuheben, die jeweils in jener Zeit die Industrie voranbrachten.

Die Industrialisierung begann mit der mechanischen Unterstützung von Arbeitsprozessen durch Dampfmaschinen o.Ä., die sogenannte „Industrie 1.0“ war geboren. Erst Ende des 19. Jahrhunderts begann das Zeitalter der „Industrie 2.0“. Dieses wurde eingeleitet durch die Entwicklung von elektrischen Antrieben, die vielseitig einsetzbar waren und daher die Industrie weiterentwickelten. Im Zuge dessen, wurde die Massenproduktion entwickelt, die in erster Linie durch die Erfindung des Fließbandes von Henry Ford geprägt wurde. Dabei trennte er einzelne Arbeitsschritte auf, welches zuerst in der Produktion des Ford Model T Anwendung fand. Die Akkordarbeit entstand hierdurch. Zusätzlich wurden Automatisierungsschritte integriert, die die Produktivität wesentlich steigern und Kosten einsparen konnten.

Die voranschreitende Entwicklung der Mikroelektronik leitete die „Industrie 3.0“ ein. Dadurch konnten in der Automatisierung und IT weitere Fortschritte geschaffen werden und zwar durch fortschreitend kleiner werdende Mikroprozessoren und Mikrocontrollern mit verbesserter Leistung. Die schrittweise Automatisierung ersetzte zunehmend die menschliche Arbeitskraft durch Maschinen. Zu den Automatisierungstechniken zählen komplexe Industrieroboter und elektronische Schaltungen z.B. speicherprogrammierbare Steuerungen (SPS), die in der Produktion eingesetzt wurden. Dieser Automatisierungsgrad ist heutzutage Standard in weiten Bereichen der Industrie. Ferner zählen zu den Zielen, die durch die Automatisierung verfolgt werden, eine sichere Produktion, Verbesserung der Qualität wie auch Kostenreduktion. Im Zuge der derzeitigen industriellen Revolution von „Industrie 3.0“ hin zur „Industrie 4.0“ ist ein Trend weg von einer hierarchischen Automatisierungsstruktur hin zu einer modular aufgebauten intelligenten Fabrik. Der Trend wird durch smarte Geräte wie Internet of Things

unterstützt, die untereinander Informationen und sich auch mit dem Menschen austauschen.

Zu den Basistechnologien der I4.0 zählen Big Data, Internet of Things und Cyber Physical Systems (Steven 2019). Auf diese Keywords im Zusammenhang zur I4.0 wird im Folgenden einzeln eingegangen und erläutert, welche Rolle diese Technologie spielt und wie diese miteinander agieren.

Die folgende Abbildung 1 stellt zusammenfassend die wesentlichen erläuterten Komponenten der I4.0 dar und ist angelehnt an die aus der Literatur „Industrie 4.0: Grundlagen-Teilbereiche-Perspektiven“ von M. Steven vorgestellten Komponenten zur I4.0.

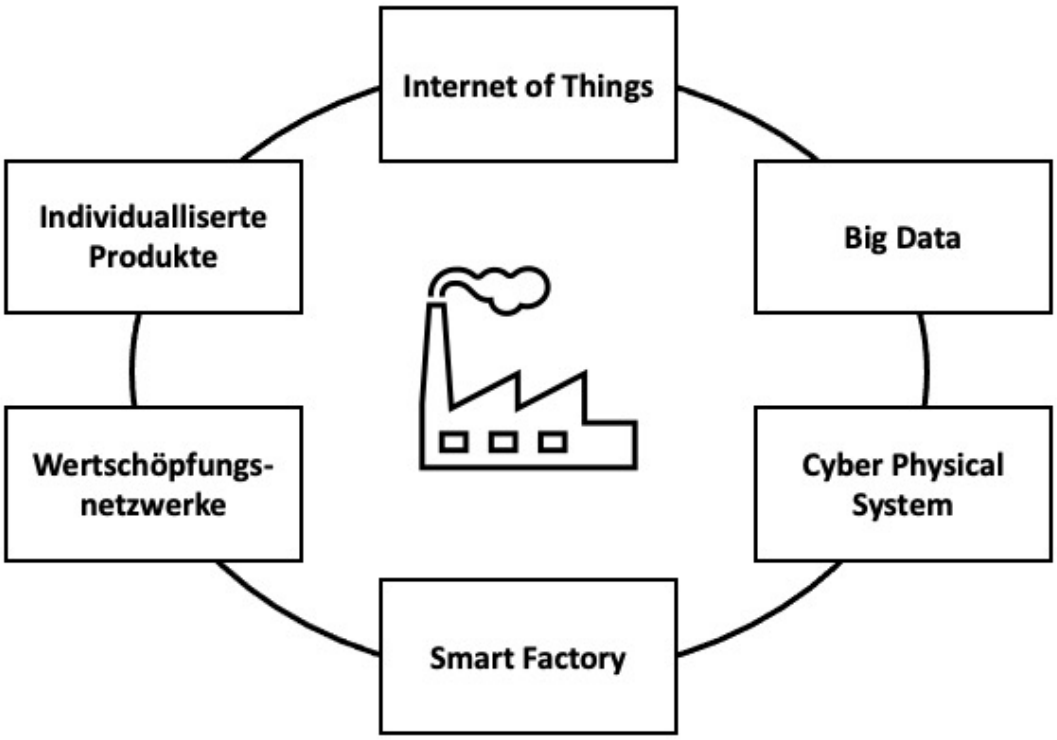


Abbildung 1: Komponenten der Industrie 4.0 (Steven 2019)

Smart Factory

In dieser Abbildung wird zusätzlich als weitere Komponente die „Smarte Fabrik“ abgebildet. Von einer Smarte Fabrik spricht man, wenn mehrere Internet der Dinge und Dienste verknüpfte Systeme in einer Produktion vorhanden sind, die weitgehend in Echtzeit miteinander kommunizieren. Durch den Anschluss der Systeme an das

Internet of Things werden die Daten umfassend zusammengetragen und dadurch wird die Produktionssteuerung und -planung dezentralisiert.

Internet of Things

Im Zuge der Industrie 4.0 ist das Stichwort „Internet of Things“ geläufig, welches in Zusammenhang zur Digitalisierung gestellt wird. Die Entstehung der Internet of Things (IoT) zu Deutsch Internet der Dinge lässt sich daher auch in den Digitalisierungsfortschritt einreihen. Zumeist war die Kommunikation eine rein zwischenmenschliche Angelegenheit, die direkt oder per Kommunikationsmedien wie bei einem Telefon vonstatten ging. Mit dem Aufkommen des Internets in den 70er und 80er Jahren änderte sich dies und Daten und Informationen wurde beispielsweise via E-Mail ausgetauscht. Durch weitere komplexere Dienste über das Internet wie Online-Handel wuchs das World Wide Web weiter, doch Endgeräte wie Computer waren zu dieser Zeit noch im Wesentlichen stationär. Mit dem Aufkommen mobiler Endgeräte wie Smartphones und Tablets änderte sich dies. Durch die Entwicklung leistungsstarker Anwendungen bzw. Apps entstand eine völlig neuartige Kommunikation zwischen den Menschen über sogenannte soziale Medien wie Facebook, Whatsapp, Twitter und Instagram. Im nächsten Schritt wurde die Kommunikation vom Menschen entkoppelt und kann ausschließlich zwischen Geräten stattfinden, welches das IoT umfasst. (Hüning 2019)

Durch die fortschreitende Miniarisierung von Elektronik und Sensoren, werden immer mehr Möglichkeiten im IoT- Bereich geschaffen, dadurch können nicht nur große Maschinen vernetzt werden, sondern auch kleine Dinge wie Mikrocontroller können Daten generieren wie auch smart und vernetzt werden. Aus diesem Grund heraus, werden reale Dinge, virtuelle Objekte sowie Menschen miteinander vernetzt und die Informationen und Daten können übers Internet übertragen werden, welche nicht mehr einschließlich von Menschen generiert werden, stattdessen vermehrt von vernetzten Dingen und Objekten selbst. (Hüning 2019)

Anfangs des Jahrtausends waren die Menschen vernetzt, doch inzwischen übertrifft die Zahl der vernetzten Objekte die menschlichen Internetnutzer weitgehend. Dabei soll die Zahl an vernetzten Objekten im Jahr 2020 bei etwa 30 Milliarden liegen (Hüning 2019) und im Jahr 2025 bei über 75 Milliarden (IoT-Prognosen: 75 Milliarden vernetze

Geräte | SAP News Center). Dazu kommt bei so einer großen Anzahl an vernetzten Geräten eine riesige Menge an generierten Daten zustande, die im Jahr 2025 bei 175 Zettabyte liegen soll, wobei ein Zettabyte 1 Mrd. Terabyte entspricht. (Daten - Volumen der weltweit generierten Daten 2025 | Statista).

Ein weiteres Schlagwort im Kontext von IoT und I4.0 ist **Cloud Computing**. Wodurch die Vernetzung von Komponenten realisiert werden kann. Die Schnittstellen über das Internet ermöglichen den Zugriff auf Informationen von Produktionsmitteln und den Zugriff auf Funktionen der Anlage. Außerdem führt der hohe Vernetzungsgrad zwischen Produktionsmitteln zu Kosteneinsparungen für Instandhaltungsprozesse, da hierdurch das Instandhaltungsmanagement nicht mehr zu jeder instand zusetzenden Anlage vor Ort sein muss. Die Fernwartung ist kein neues, etabliertes Thema in der Industrie, führte aber häufig zu Problemen wie beispielsweise unterschiedliche Konfigurationen, Tools oder verschiedener Zugriffsmöglichkeiten über Mobilfunkanbindungen, da es keine standardisierten Zugänge gab. Zugleich werden Wertschöpfungsprozesse im I4.0-Umfeld durch die Anwendung von Cloud-Architekturen aufgebrochen. Der klassische Wertschöpfungsprozess wird von einer Partei überwacht, hingegen der cloudbasierte Wertschöpfungsprozess auch von unabhängigen Parteien überwacht werden kann. Dabei können die Parteien unabhängig voneinander den Wertschöpfungsprozess betreuen, überwachen und eingreifen. (Vogel-Heuser et al. 2017)

Solche Datenmengen ermöglichen neue Geschäftsmodelle, im Hinblick auf Big Data, unterdessen muss diese Menge an Daten sinnvoll analysiert und ausgewertet, sodass die Nutzung relevanter Informationen gewährleistet werden kann (Hüning 2019).

Big Data

Die ständig erzeugten Daten von Menschen, Maschinen, Anlagen, Geschäftsprozessen, Produkten usw. erschaffen neue Möglichkeiten im Datenmanagement. Durch die Zusammenführung und effiziente Analyse dieser Daten und durch unter anderem Cloudserver können Ressourcennutzungen und Prozesse optimiert werden (Vogel-Heuser et al. 2017). Solche Möglichkeiten werden unter Big Data gefasst.

Beispielsweise können kooperierende Unternehmen eines Wertschöpfungsprozesses mit bestimmten Zugriffsrechten bei Bedarf Daten bzw. Informationen zum Prozess vom

Server abgerufen. So ist eine bedarfsgerechte Einlastung von Aufträgen und eine Echtzeitsteuerung von Produktionsmitteln möglich. Außerdem können Alternativen durchgespielt und Simulationen von Auftragskonfigurationen anhand der Daten durchgeführt werden. (Steven 2019)

Cyber Physical Systems

Eine weitere Basistechnologie der I4.0 sind Cyber Physical Systems (CPS) die im Deutschen als cyberphysische Systeme gleichnamig übersetzt werden. CPS sind durch die neu geschaffenen Möglichkeiten der IoT realisierbar geworden, hierfür sind verbesserte Datenerfassung, -speicherung und -verarbeitung notwendig, die mithilfe der Kommunikationsmöglichkeiten von IoT realisiert werden. Folgend werden drei Definitionen zu einem CPS vorgestellt und Zusammenhänge erläutert.

M. Steven fasst CPSs zusammen als, erweiterte physische Systeme, durch IoT-Module für neue Kommunikationsmöglichkeiten mit integrierten Sensoren und eingebetteten Systemen. Sie sind dazu fähig in einem vorgegebenen Rahmen Entscheidungen zu treffen und können mit ihrer Umwelt interagieren. In der Produktion werden beispielsweise als Basis für solch ein CPS Fertigungsanlagen und -mittel zusammengefasst. In der Logistik dagegen sind es beispielsweise Fahrzeuge und Förder- bzw. Transportmittel. (Steven 2019)

Lechler und Schlechtendahl bezeichnen CPSs als ein Roboter, Maschine oder ein gesamtes Produktionssystem, welche ihre Steuerungsfunktionen in eine Cloud verlagern, um mit anderen weiteren, integrierten Objekten ohne Hardwareschnittstellen Informationen über Services austauschen können. Das schafft die Grundlage, dass jenes System wandelbar und lernend auf ihr Umfeld und Bediener agieren kann. (Lechler/Schlechtendahl 2017)

Nach Mourtzis und Vlachou werden CPSs als Systeme definiert, in denen natürliche und vom Menschen geschaffene Systeme (physischer Raum) eng mit Rechensystemen, Kommunikations- und Steuerungssystemen (Cyberspace) integriert sind. CPSs verbinden die physische mit der virtuellen Welt durch flexibles, kooperatives und interaktives Arbeiten. (Mourtzis/Vlachou 2018)

Zwischen den drei Definitionen gibt es zwei wiederkehrende Aspekte, die von allen aufgegriffen wurden. Zudem ist für ein CPS ein physisches Objekt beispielsweise ein Roboter notwendig und des Weiteren die Fähigkeit des Systems mit der Umwelt interagieren zu können. Zusätzlich finden sich Zusammenhänge in der Verbindung des CPS mit einer Cloud bzw. IoT und oder Kommunikations- und Steuerungssystemen.

In der unteren Abbildung 2 ist abgebildet, wie die Zusammenarbeit und Kombination aus den vorangegangenen Basistechnologien die Entstehung von I4.0 zusammenspielen. Die Abbildung ist angelehnt an die Darstellung des Zusammenspiels der Basistechnologien von I4.0 von M. Steven.

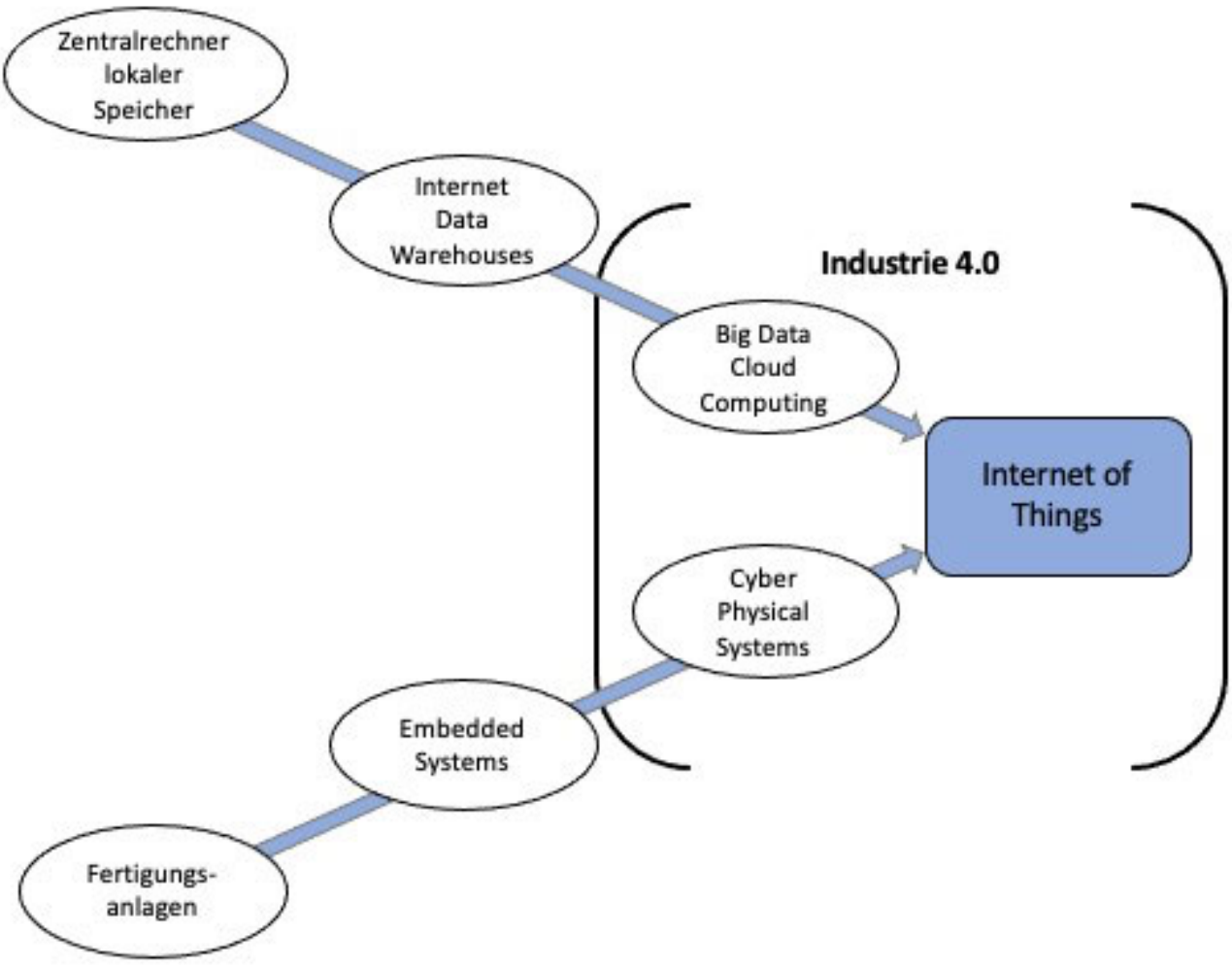


Abbildung 2: Zusammenspiel der Basistechnologien von I4.0 (Steven 2019)

Aus der oberen Linie der Abbildung lässt sich die Entwicklung im Bereich der IT ableiten. Zu Beginn am Zentralrechner mit lokaler Speicherung läuft die Industrie dazu über, die generierten Daten über Data Warehouses zu sammeln und zu managen. Im nächsten Schritt ist der Umgang mit den Data Lakes (Datenseen) und Data Warehouses (Datenbanken) zu bewältigen, dies dazu führte, das Big Data entstand. Um solche Menge zu bewältigen, außerdem wurde für wesentliche Anwendungen Daten in Cloud-Services ausgelagert.

Die untere Linie spiegelt die Entwicklung des Produktionsbereichs bzw. Shopfloor wieder, angefangen von der klassischen Fertigungsanlage, weiter zur optimierten Version durch Embedded Systems, hin zum cyberphysischen System.

Für das weitere Verständnis zur I4.0 ist die Beleuchtung der Einflussfaktoren der Entwicklung und Entstehung notwendig. In der folgenden Abbildung 3 sind einzelne Faktoren, die Einfluss auf die Entwicklung von I4.0 haben, aufgezeigt.

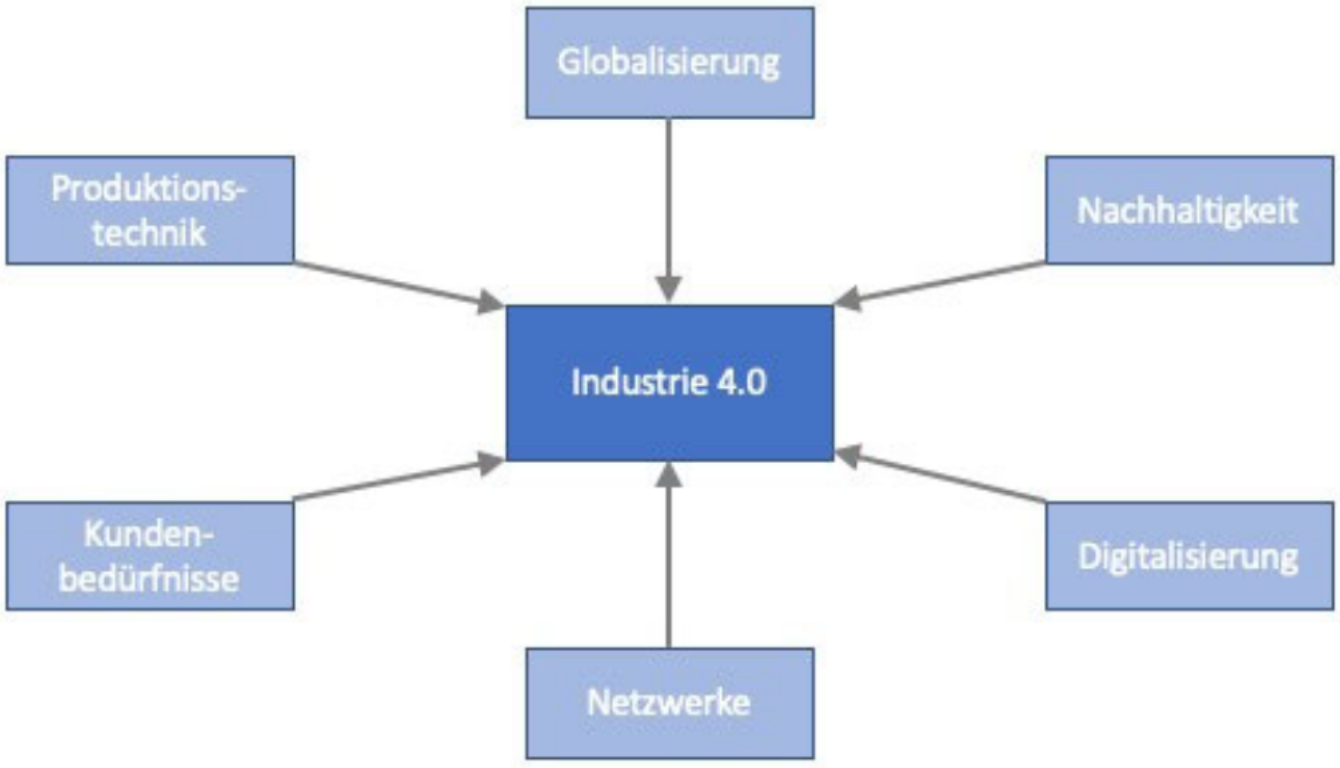


Abbildung 3: Einflussfaktoren der Industrie 4.0 (Steven 2019)

Vor bereits mehreren Jahrzehnten besteht das Streben zur **Globalisierung** im wirtschaftlichen Interesse. Ausgehend vom Bestreben des Erreichens internationaler Märkte ist die Konkurrenz groß und man wird gezwungen das Kostensenkungspotenzial weiter auszuschöpfen. Dafür werden Einkauf weiter ausgeweitet und die Standortauswahl einzelner Produktionsstätten weiter ausgedehnt. Aus diesen Umständen heraus wird es erforderlich, eine neue Steuerung von Supply Chain-Vorgängen einzuführen, welches Einfluss auf die Weiterentwicklung der Industrie vorweist.

Die **Nachhaltigkeit** in der Industrie wird seit Jahrzehnten berücksichtig, gewinnt aber immer mehr an Bedeutung, daher treibt auch hier dieser Aspekt die I4.0 voran, da viele Ressourcen endlich sind.

Durch die fortschreitende **Digitalisierung** auf der Welt werden zunehmend Geschäftsprozesse und Produktionsprozesse davon eingenommen. Durch die daraus einhergehende verbesserte Informations- und Kommunikationstechnik (IKT) wird der Zugriff auf Informationen und Daten dezentralisiert, vereinfacht und beschleunigt, sodass aktuellere und umfassendere Informationen möglich werden. Hierdurch werden neuartige Prozesse, die auch Anwendung in dieser Masterarbeit finden, ermöglicht.

Ein weiterer Trend der heutigen Zeit ist das Outsourcing, d.h. das die Wertschöpfungstiefe der eigenen Produkte aus dem Unternehmen getragen wird und nicht mehr alle Anteile der zugehörigen Wertschöpfung eines Produktes im Unternehmen stattfindet. Daraus folgt, dass es immer speziellere Zulieferer und einzelne bedeutsamere Supplyer daraus entstehen. Mit solchen bedeutsamen Supplyer ist der Zusammenschluss von großer Bedeutung, damit der Ausfall so gering wie möglich gehalten wird. Daher werden sogenannte Supply Chains bzw. **Netzwerke** mit diesen Unternehmen gebunden, um eine bessere Zusammenarbeit zu garantieren. Hierunter werden nicht nur operative Arbeitsprozesse wie Logistikprozesse gefasst, sondern ein Zusammenschluss der einzelnen zugehörigen Wertschöpfungsprozesse jener notwendigen Produkte, die miteinander verbunden sind. Hierdurch wird die Entwicklung der Industrie durch kooperative Netzwerke der I4.0 maßgeblich beeinflusst.

Das Verlangen des Kunden nach einer stetig steigenden Individualisierung ihrer Produkte nimmt immer mehr zu. Ein Extrem wäre die Losgröße eins, dies durch die individuelle Komplexität eines geforderten Produktes entstehen kann. Dabei ist bei einem wiederkehrenden, variierenden Herstellungsprozess hohe Flexibilität gefordert. Die

notwendige Flexibilität bei gleichgleichbleibender Qualität fordert ein hohes Maß an Planung und dadurch neue digitale Geschäftsmodelle im Kontext der I4.0 um jeden **Kundenbedürfnissen** gerecht zu werden.

Auch die Entwicklung neuer Produkte und die daraus resultierenden Produktionsverfahren schreiten voran, der sogenannte technische Fortschritt. Die Konsequenz daraus ist, dass sich der Innovationszyklus jener Produkte verschiebt wie auch der Produktlebenszyklus. Diese Veränderung haben Auswirkungen auf die **Produktionstechnik** sowie die zugehörige Logistikprozesse, die Veränderung in der Industrie hervorrufen.

Zur Realisierung von cyberphysischen Systemen im Rahmen von I4.0 werden unterschiedliche Basistechnologien und Methoden vermehrt und kombiniert eingesetzt. Eine bedeutende Technologie ist der Einsatz von künstlicher Intelligenz (KI) beispielsweise in Form von intelligenten Softwaresystemen. Wie schon in den Definitionen zu CPS erwähnt, wird es möglich sein, situationsabhängige Entscheidungen selbst bzw. automatischen zu treffen. Ein bedeutendes Anwendungsfeld eines solchen CPS ist unter anderem die aktive Prozesssteuerung anhand aktueller Maschinendaten und Prozessparameter und eine vorausschauende Wartung anhand bereits, erhobener Daten von Sensoren beispielsweise einer Fertigungsanlage. (Steven 2019)

Die Rolle des Menschen in einer sich selbst organisierenden, steuernden und kommunizierenden Produktion wird sein, die Prozessstrategie zu entwickeln, am Laufen zu halten und zu überwachen. Somit wird der konventionelle Arbeitsplatz wie ein Leitstand oder Büro zunehmend an Bedeutung verlieren. Die dezentrale Steuerung erlaubt, die Entscheidungsprozesse nicht ortsgebunden durchführen zu können. Resultierend daraus entsteht die Annahme, dass der einzelne Mitarbeiter zukünftig eine verantwortungs- und wirkungsvollere Rolle spielen wird. Zugleich wird der Mensch als eine letzte Instanz innerhalb eines cyberphysischen Gefüges wirken, wenn komplexe Probleme auftreten, wie eine Störung oder potenzielle Optimierungsmethoden zu erschließen sind. Die folgende Abbildung 4 stellt den Menschen in solch einem cyberphysischen Gefüge als letzte Instanz im Entscheidungsprozess dar. Diese Abbildung entstammt der Grundidee als solches, dass der Mensch als Überwacher der Produktionsstrategie und letzte Instanz im Entscheidungsprozess fungieren soll, von Vogel-Heuser et. al. (Vogel-Heuser et al. 2017)

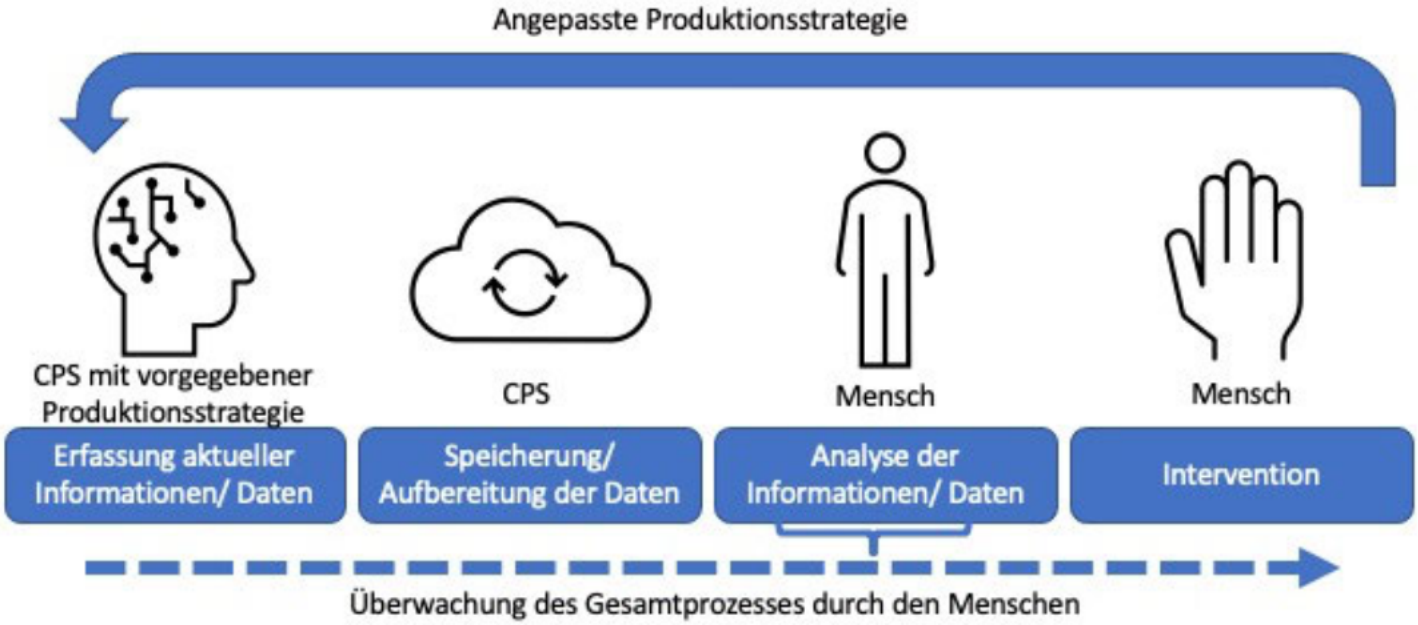


Abbildung 4: Mensch als letzte Instanz im cyberphysischen Gefüge

2.1.2 Beispiele von Konzepten zur Industrie 4.0

Die vierte industrielle Revolution treibt die Industrie zu immer neuen Möglichkeiten durch Einflussfaktoren wie beispielsweise die Digitalisierung (vgl. Kap. 2.1.1). Der Einsatz von Basistechnologien im Kontext der I4.0 ist notwendig, um Möglichkeiten wie mehr Transparenz im Datenfluss zu schaffen (vgl. Kap. 1.3), oder eine vorausschauende Instandhaltung zu ermöglichen. Das Thema IoT spielt hierbei eine große Rolle, da erst durch diese Technologie die Möglichkeit der dezentralen Datennutzung ermöglicht wird. Hinsichtlich des Datentransfers von Produktionsmaschinen hin zur dezentralen Datenspeicherung und Nutzung ist es erforderlich, wesentlichen Sensoren von Produktionsmitteln (Maschine, Anlage, usw.) IoT-fähig zu gestalten (vgl. Kap. 2.1.1).

Es existieren bereits Konzepte im Bereich von Cloud Computing und Maintenance im Zusammenhang mit IoT-Anwendungen. Zwei existierende Anwendungsbeispiele werden im Folgenden vorgestellt, um die Möglichkeiten von I4.0 darzustellen.

Erstes Anwendungsbeispiel stammt von D. Mourtzis und E. Vlachou und wurde im Journal of Manufacturing Systems im Jahr 2018 veröffentlicht.

Ziel dieses Konzeptes ist es, die traditionelle Produktion aufzubrechen und zu modernen, digitalisierten Systemen weiterzuentwickeln. In dieser Veröffentlichung wird ein cloudbasiertes cyberphysisches System zur Produktionsplanung und zustandsorientierten Instandhaltung entwickelt. Dafür soll die Integration von Internet of Things in die Produktion effektive und anpassungsfähige Planungs- und Steuerungssysteme schaffen.

Der Hauptbeitrag des vorgeschlagenen Systems lässt sich wie folgt zusammenfassen. Es handelt sich dabei um ein kostengünstiges Überwachungssystem, welches in der Lage ist, Daten von verschiedenen Quellen zu empfangen und über ein kabelloses Sensornetzwerk und Kommunikationsprotokoll weiterzuleiten. Weiterhin wurde ein Algorithmus von unterschiedlichsten Kriterien zur Findung von Entscheidungen für eine adaptive Produktionsplanung anhand verarbeiteter Echtzeitdaten erarbeitet. Also ein cyberphysisches System, das aus mehreren Modulen besteht, welche untereinander kommunizieren können und in einer Cloud-Architektur entwickelt wurde. Unterstützt durch Technologien der Industrie 4.0, der Digitalisierung und Internet of Things. Insgesamt ein System, dass sich leicht integrieren lassen soll und welches in diesen Fall auch schon in einer Formgebungsindustrie erfolgreich angewendet wurde.

Die untere Abbildung 5 entstammt der Veröffentlichung und stellt das Zusammenspiel der einzelnen Module des vorgestellten Konzeptes dar. (Mourtzis/Vlachou 2018)

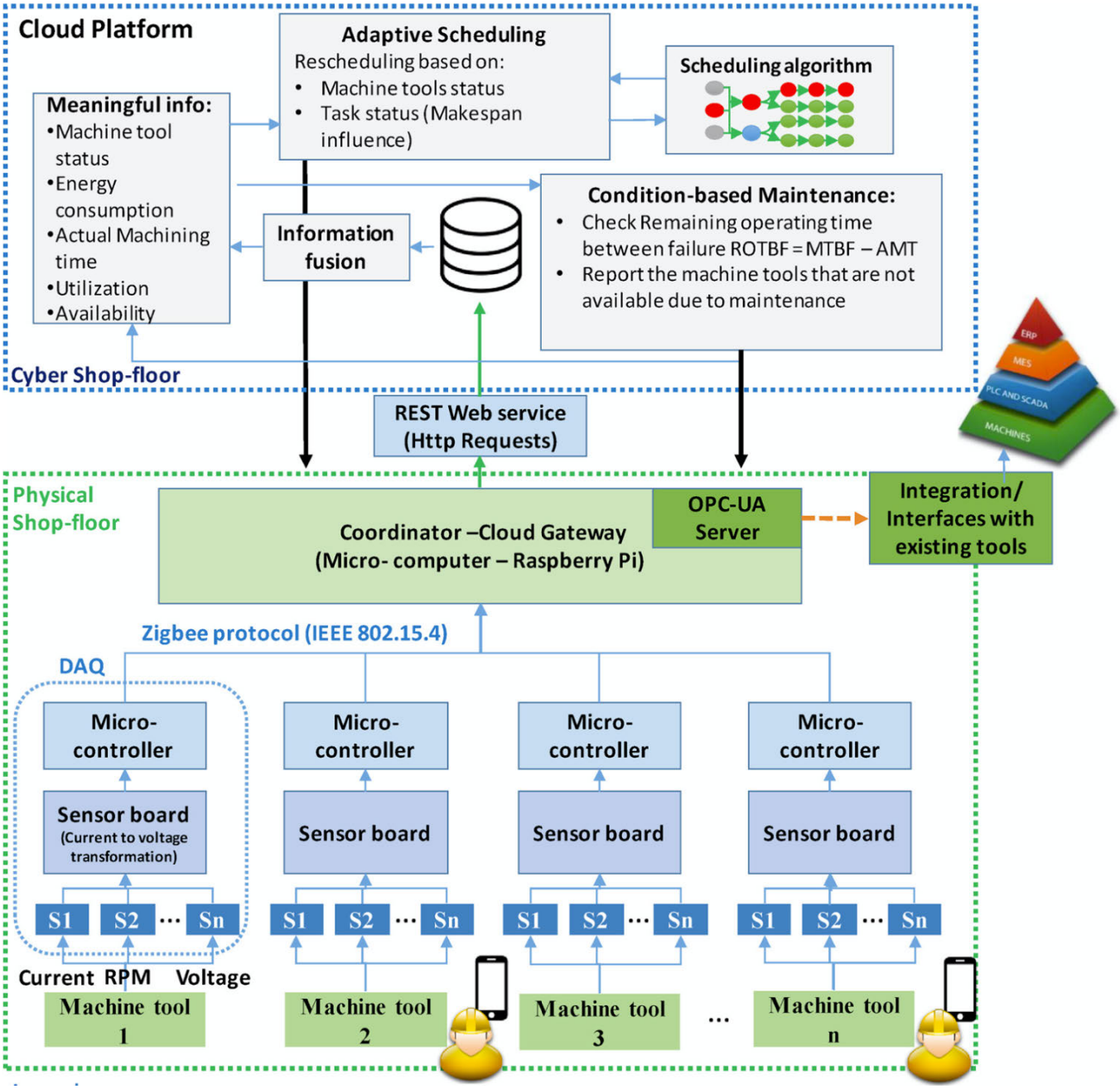


Abbildung 5: Cloudbasiertes cyberphysisches System (Mourtzis/Vlachou 2018)

Das zweite Anwendungsbeispiel stammt von P. Clausen et al. und wurde publiziert im Wireless Personal Communications Journal aus dem Jahr 2020.

Titel dieses Konzeptes lautet „Smart Manufacturing Through Digital Shop Floor Management Boards“. Das Schlagwort „Smart Manufacturing“ wurde im Kontext der I4.0 entworfen und umfasst die Zukunft von produzierenden Industrien. Smart Manufacturing führt die Digitalisierung im Shopfloor (vgl. 2.2.1) bzw. Produktionshalle voran, welches automatisiert, computergestützt und komplex abläuft. Zweck dieser Publikation ist es, das Verständnis von Shopfloor Management Board Meetings im Rahmen der I4.0 zu verbessern und hin zur Nutzung solcher Boards zu motivieren.

Big Data (vgl. Kap. 2.1.1) spielt eine große Rolle im Zusammenhang zur voranschreitenden Digitalisierung der Produktionsumgebung, dabei treibt die I4.0 die Verwendung dieser Daten auf intelligente Weise durch smarte Technologien voran. Zur Umsetzung eines digitalen SFM Boards sollten die Stärken, die ein solches Board hervorbringt, kurz zusammengetragen werden. Grundlage der Stärken eines solchen Systems sind auch hier die Daten bzw. Big Data. Zu den Stärken zählen unter anderem bessere Datentransparenz, Zuverlässige Daten in Echtzeit, bessere Datenzugänglichkeit wie auch frühe Problemerkennung und eine datengestützte Entscheidungsfindung. Kräfte, die aus dieser Publikation hervorgehen, die gegen ein solches SFM sprechen, sind unter anderem anspruchsvolle Ressourcen, mögliche Datenblindheit, geringes Kompetenzniveau, unstrukturierte Datenspeicherung wie auch zeitaufwendige Umsetzung.

Erkenntnisse dieser Publikation zu Umfragen an Unternehmen zu der Nutzung von digitalen SFM Boards in Meeting zu den gerade erläuterten Stärken sind, die Abschaffung von Informationssilos und zeitaufwendiger Informationssammlung. Weiterhin soll der einfachere, geschaffene Informationsfluss zu verbesserten Kommunikationen zwischen unterschiedlichen Organisationen führen. Kräfte, die gegen eine solche Nutzung von SFM Boards wirken, sind unteranderem unausgereifte IT-Technologien, die zu Ausfällen führen und geringes Engagement für Veränderungen hin zu smart Manufacturing bzw. digitales SFM. (Clausen et al. 2020)

2.2 Technische Grundlagen

In diesem Kapitel werden technische Grundlagen erläutert, die im Rahmen dieser Masterarbeit im Kontext zu dem theoretischen Konzept und der erarbeiteten Fallstudie stehen und zum Verständnis des Konzeptes helfen.

2.2.1 Shopfloor Management

Die Industrie strebt nach Globalisierung und zwingt Unternehmen stetig produktiver und effizienter zu werden, um dem globalen Wettbewerb standzuhalten. Damit Unternehmen solche Leistungssteigerungen erreichen können, wird häufig Prozessoptimierung herangezogen, welche mittels Methoden und Werkzeugen umgesetzt werden. In diesem Fall hat die Produktion bzw. der sogenannte Shopfloor eine große Bedeutung. Denn gerade hier, werden Werte erschaffen, die produzierende Unternehmen seinen Erfolg sicherstellen. Um diesen Erfolg nachhaltig zu erreichen, ist das Managen des Shopfloors essenziell.

Die Verbindung des Shopfloor mit der oberen Managementebene führt häufig zu erheblichen Problemen, daher wird das Thema Shopfloor Management (SFM) bedeutsamer. Der tayloristische Ansatz wird heutzutage noch mehrfach verfolgt und führt dazu, dass die Managementebene keinen direkten Kontakt zur Produktion hat. Da die obere und mittlere Managementebene weitesgehend von der Produktion entkoppelt sind. Aus diesem Grund findet deren Tagesgeschäft nur innerhalb der Managementebene statt und haben nicht genügend Zeit in Kommunikation mit der Produktion zu treten.

Durch SFM kooperieren verschiedene Organisationsebenen miteinander und schaffen die Möglichkeit schneller, flexibler und zielorientierter Probleme zu lösen. Außerdem fungiert SFM als Instrument eine Lernbereitschaft im Unternehmen zu schaffen, um den Lean-Gedanken weiterzuführen.

Mangelnde Transparenz über Vorgänge zur Lösungsfindung, unzureichende Kommunikation über Organisationsebenen hinaus sowie fehlende Standards, die unterstützen sollten, führen dazu, dass kleine Probleme in Produktion und administrativen Bereichen mit sehr großem Aufwand bewältigt werden müssen. Dabei steht der Aufwand

der Problemlösung nicht in Relation zur Größe des Problems, hierbei soll SFM eingreifen, um Lösungen vor Ort schnell und effizient zu finden.

Das transparente und standardisierte Führen am Ort des Geschehens (Shopfloor) beschriebt SFM mit Zuhilfenahme von Werkzeugen und Optimierungsmethoden zur nachhaltigen Prozessverbesserung über alle Hierarchieebenen hinweg. Solche zu implementierenden Managementsysteme wie SFM oder Lean Management werden in vier Kernelemente unterteilt. Dazu zählen Transparenz schaffen, Optimierung und Standardisierung von Prozessen, Kennzahlen zum Führen und Kultur und Organisation auf dem Shopfloor. In der folgenden Abbildung 6 sind diese Elemente dargestellt. Diese Abbildung ist abgeleitet aus der von B. Leyendecker und P. Pötters stammenden Literatur „Shopfloor Management: Führen am Ort des Geschehens“ zum Thema Definition und Nutzung von SFM.

Abbildung 6: Kernelemente von Shopfloor Management

Der Schwerpunkt von SFM umfasst, dass direkte Lösen am Ort des Geschehens in Zusammenarbeit aller Führungskräfte und eingebundenen Mitarbeitern, um gemeinsam zu analysieren, was für ein Problem vorliegt und wo der Ursprung liegen könnte. Hierbei ist ein respektvoller Umgang Voraussetzung, wodurch Hierarchieebenen aufgebrochen werden und alle Mitwirkenden mehr Verantwortung zugesprochen werden.

Die Visualisierung ist ein Erfolgsfaktor für die Kernelemente Standardisierung und Transparenz und spielt daher eine große Rolle im SFM. Dabei ist es erforderlich, den Informationsfluss zwischen Prozessen im Shopfloor hin zu indirekten Prozessen im Management erfolgreich sicherzustellen.

Definierte Standards werden auch als Voraussetzung für Verbesserungen gesehen und schaffen daher Möglichkeiten wie Optimierungen von Prozessen. Schon seit Beginn des Qualitätsmanagements nach William Edwards Deming sind eindeutige Standards erfolgsversprechend, da ansonsten vermehrt Probleme verschleiert bzw. nicht entdeckt werden können.

Auch vor SFM macht die Digitalisierung kein Halt und treibt die Weiterentwicklung voran. Potenziale, die sich hierbei für das SFM ergeben, sind unter anderem analoge Prozesse und Informationen digital zu gestalten wie beispielsweise durch Softwareanwendungen. Der Einsatz von IKT ermöglicht die Umgestaltung aktueller Geschäftsprozesse mit dem Fokus, Vernetzungen hierarchieübergreifend zu schaffen, Schnittstellenanzahl zu reduzieren und Prozesse effizienter zu gestalten. Eine weitere geschaffene Möglichkeit durch das digitale SFM ist die orts- und zeitunabhängige Datenerfassung und -verarbeitung (vgl. Kap. 2.1.1), die Entscheidungsfindungen (Decission-Making-Process) unterstützen soll. Dadurch wird der Zugang zu Datenflüssen im Unternehmen begünstigt bzw. die Datenverfügbarkeit erhöht, wodurch weitere relevante Analysen in das SFM integriert werden können. In der Produktion können spontan auftretende Störungen von Mitarbeitern digital erfasst werden, wie zum Beispiel über Tablets, welche anschließend in Echtzeit visualisiert werden können. Solche Störungen können dann in aktuellen Besprechungen gemeinsam analysiert und Problemlösungen gefunden werden mithilfe von digitalen Shopfloor Boards. Dabei besteht das Potenzial, durch die ortübergreifende Einbindung von Experten, die Gruppen zu erweitern.

Zusammengefasst schafft die Digitalisierung des Shop Floor Managements Transparenz, schnellere Reaktionswege und eine verbesserte Effizienz und Effektivität im Kontext des SFM. (Leyendecker/Pötters 2021)

Die folgende Tabelle 1 stellt das klassische SFM und das digitale SFM gegenüber.

Tabelle 1: klassisches SFM gegenüber digitalem SFM (Leyendecker/Pötters 2021)

	Klassisches SFM	**Digitales SFM**
Visualisierung	Analog/ papiergebunden über Shopfloor Boards (Whiteboards)	Digital/ papierlos übers digitale Shopfloor Board (interaktiv)
Optimierung	Reaktionsfähigkeit bei einer Störung analog weiterzu-geben	Störung digital erfasst und in Echtzeit auf Shopfloor Board visualisiert
Datenaufzeichnung	Manuell über Flipcharts	Digital per Tablet/ PC
Datenaufbereitung	Manuell durch die Führungskraft	Automatisiert durch IT-Applikationen
Aktualisierung des Shopfloor Boards	Regelmäßig durch die Führungskraft	In Echtzeit über IT-Systeme
Transparenz und Zusammenarbeit	Lokal und werksintern	Mobil und standortübergreifend möglich

2.2.2 Kommunikationsprotokoll MQTT

Das Kommunikationsprotokoll „MQTT“ wurde von dem Unternehmen IBM entwickelt und hatten den Zweck der Überwachung von Ölpipelines. Seit dem Jahr 2010 ist das M2M-Protokoll (Machine to Machine) frei verfügbar. Das Protokoll ermöglicht, Nachrichten über unzuverlässige Satellitenverbindungen zuverlässig zu verschicken, ohne dabei eine große Bandbreite und viele Ressourcen in Anspruch zu nehmen. (Hüwe/Hüwe 2019)(Senft 2019)

MQTT ist eine Abkürzung für „Message Queue Telemetry Transport“ und steht für ein leichtgewichtiges Warteschlangen-und Transportprotokoll von, wie der Name schon verrät, Telemetriedaten (Egli 2013), dieses baut auf TCP/ IP-Protokollen auf (ICIDCA (Conference) et al. 2020).

Zur Realisierung der M2M-Kommunikation ist ein sogenannter **MQTT-Broker** notwendig, der als Server fungiert (Dietrich 2021). MQTT verwendet eine Publish/ Subscribe-Architektur. Darunter ist gemeint, dass Meldungen von Geräten an den Broker gesendet werden, sogenanntes Publizieren ihrer Nachrichten. Wenn ein Gerät bzw. Client eine Nachricht empfangen möchte, muss dieser Client den Server subskribieren. Nicht der Server selbst fungiert hierbei als Adresspunkt der Nachrichten, sondern ein erzeugtes Topic eines Servers. Hierüber werden Nachrichten gefiltert, d.h. wenn ein Client ein bestimmtes Topic subskribiert bzw. abonniert, empfängt dieser Nachrichten im Push-Verfahren des adressierten Topics vom Broker. Im Grunde bildet der Broker das Zentrum des Protokolls und empfängt alle Nachrichten, filtert diese und versendet diese an interessierte Clients. Voraussetzung eines solchen Clients ist, dass ein TCP-Stack und das MQTT-Protokoll implementiert sind und außerdem eine Netzwerkverbindung zum jeweiligen MQTT-Broker vorhanden sind. Weiterhin beachtet der Broker die abgestufte Qualitätssicherung einer Nachricht und garantiert für eine konkrete Anlieferung der Messages, welches zu den Aufgaben eines Brokers zählen. Die untere Abbildung 7 stellt die beschriebene Publish-Subscriber-Architektur von MQTT dar. (Dietrich 2021)(Trojan 2017)

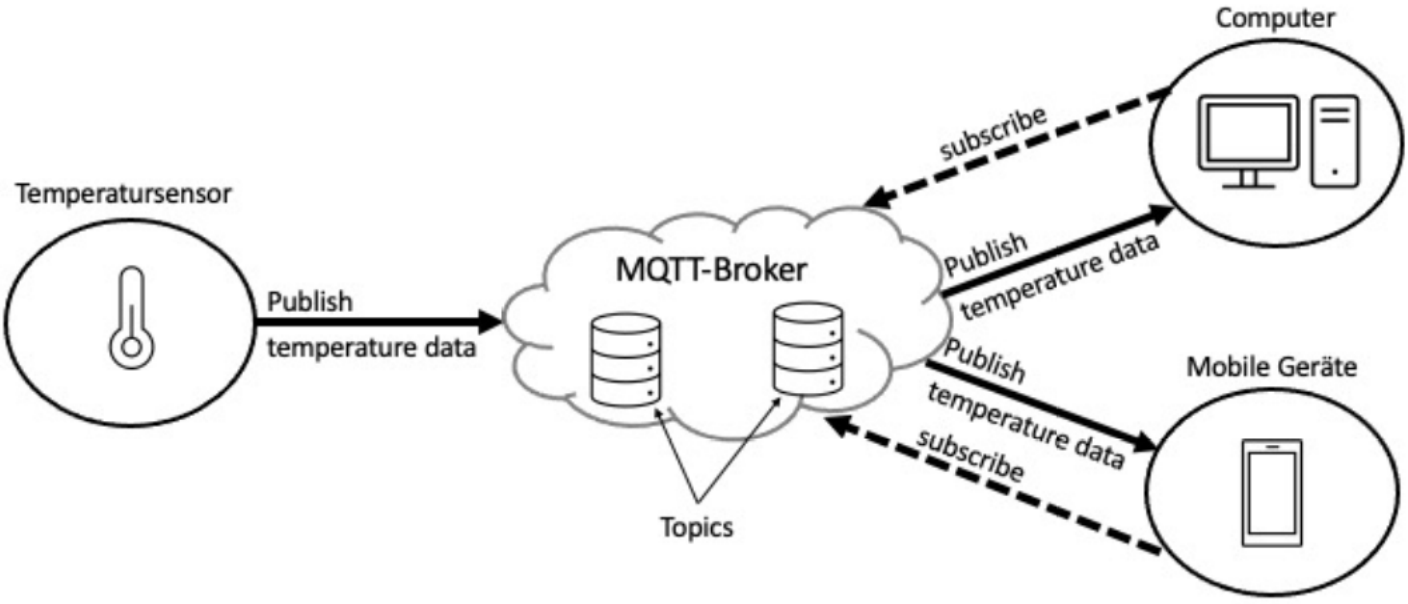


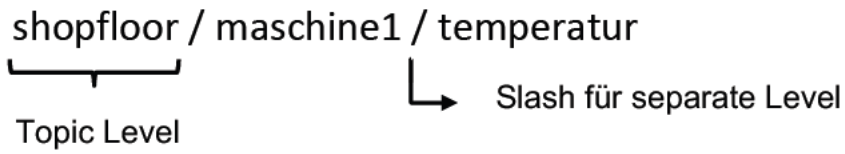
Abbildung 7: MQTT Publish-Subscriber-Modell

Im Folgenden werden Vorteile einer Publish/ Subscribe-Architektur wie von MQTT vorgestellt: (Trojan 2017)

- Weder Publisher oder Subscriber müssen sich kennen, nur der zentrale Broker muss bekannt sein für jeden Client
- Sender und Empfänger müssen nicht gleichzeitig online sein, um die versendete Nachricht zu empfangen, d.h. schaltet sich der Empfänger erst später ein, kann er trotzdem die letzte Nachricht erhalten
- Wartezeiten binnen der Publizierung/Subskribierung entfallen auf beiden Komponenten

Die Filterung der Nachrichten wird durch **Topics** durchgeführt, hierbei werden Nachrichten an Topics adressiert. Ein solches Topic. Ist ein String, welches als Betreff einer Nachricht dient. Dazu werden zum Topic noch die Nutzdaten (Payload) übermittelt. MQTT ist datenunabhängig, somit können binäre Zahlen, Texte wie auch XML- oder JSON-Strukturen übermittelt werden. Ein Topic kann aus mehreren **Hierarchiestufen bzw. Leveln** bestehen, dabei dienen die Level der Organisation.

Jede Hierarchiestufe ist mit einem Slash getrennt, hierbei ist darauf zu achten, dass MQTT die Groß- und Kleinschreibung beachtet, daher ist hier ein besonderes Augenmerk nötig, da ansonsten das gewünschte Topic nicht erreicht werden kann. Das folgende Beispiel stellt ein Topic bestehend aus mehreren Leveln dar:



Diese Adresse sagt aus, dass die Temperaturdaten aus dem Shopfloor der Maschine 1 empfangen werden sollen.

Zur vereinfachten Filterung von Topcis lassen sich zwei Arten von **Wildcards** verwenden. Die erste ist die Ein-Level-Wildcard und sieht wie folgt aus:

shopfloor / + / temperatur

Mit dieser **Ein-Level-Wildcard** werden alle Elemente eines Levels empfangen. Somit ist es möglich, mit diesem Filter durch ein „+“ für das Level Maschinen die Temperatur aller Maschinen aus dem Shopfloor herausfiltern. Die zweite ist die **Mehr-Level-Wildcard** und wird wie folgt angewendet:

shopfloor / maschine1 / #

Durch die eingefügte Raute „#“ werden alle Daten der „maschine1“ herausgefiltert. Solch eine Wildcard darf immer nur am Ende einer Adresse ausgeführt werden.

Eine weitere Aufgabe eines MQTT-Brokers ist **Quality-of-Service (QoS)**. MQTT besitzt drei Qualitätsstufen einer Nachricht. Diese Auswahl kann sowohl beim Empfangen oder Verschicken einer Nachricht verwendet werden.

Erste Stufe ist QoS 0

Diese Minimalstufe entspricht der Qualität von TCP/ IP und beinhaltet daher auch die geringste Netzbelastung. Bei dieser QoS Stufe spricht man von „fire and forget“, d.h. versendet und weg mit der Nachricht. Dabei erfolgt keine zusätzliche Bestätigung des Empfangens oder Zwischenspeicherung des Sendens.

Zweite Stufe ist QoS 1

Durch diese Qualitätsstufe wird die Message mindestens einmal dem Empfänger zugesendet. Hierbei wird die Nachricht vom Publisher nach dem Veröffentlichen zwischengespeichert, falls der Empfänger keine Rückmeldung in einer bestimmten Zeitspanne gibt, wird die Nachricht erneut übermittelt.

Dritte Stufe ist QoS 2

Quality of Service 2 sorgt dafür, dass jene Meldung nur garantiert einmal verschickt wird. Hierbei wird die empfangene Nachricht einmal dem Sender bestätigt und verarbeitet die Meldung, wenn die Bestätigung beim Sender angekommen ist, schaltet dieser eine mögliche Wiederholung ab und quittiert die Bestätigung. Durch die Bestätigung der Bestätigung des Empfangs vollendet der Empfänger seine Meldung und löscht die gespeicherten Informationen. Anwendungsbeispiel dieser Qualitätsstufe wäre ein Empfänger, der exakt eine Meldung benötigt.

Retain-Flag

Nachrichten werden im Wesentlichen vom Broker nicht gespeichert und sofort gelöscht, ausgenommen von dieser Regel sind Mitteilungen, an denen Retain-Flags gesetzt sind. Durch solch eine Retain-Flag wird die Mitteilung von dem Broker unter dem adressierten Topic zwischengespeichert und an jeden neuen eingewählten Empfänger des Topics neu zugeschickt, bis sich das Payload ändert.

Letzter Wille

Eine weitere Möglichkeit von MQTT die angewendet werden kann, ist der letzte Wille eines Clients, wobei der Client mit seinem Ableben ein Testament hinterlassen. Dies kann durch einfaches Abschalten oder auch einem Netzwerkverlust erfolgen. Der letzte Wille wird erst verschickt, wenn der Client in einem bestimmten Intervall nicht auf sein Ableben reagieren kann. Im Anschluss würde der Broker den letzten Willen des Clients übermitteln. (Trojan 2017)

Im IoT-Bereich ist die Sicherheit ein bedeutendes Thema, da die Zahl der Nutzungen von intelligenten Dingen steil ansteigt (vgl. Kap. 2.1.1). Beim Einsatz von MQTT ist die Sicherheit auf wenige eigene Optionen begrenzt, doch verfügbare Standardlösungen wie TLS/ SSL (Transport Layer Security/ Secure Sockets Layer) können zur

gegebenen Sicherheit kombiniert werden (Lauzi 2019). Auf der Netzwerkebene erbringt die Kopplung der IoT-Komponenten an eine WLAN-Verbindung eine Verschlüsselung der Kommunikation zum Router. Ein VPN (virtuelles Privates Netzwerk) schafft darüber hinaus eine sichere Verbindung bis zum Broker. TLS wird, wie der Name schon sagt, auf der Transportebene angewendet. TLS ist der Nachfolger von SSL. Zusätzlich können die eben erwähnten Hilfsmittel zur Sicherheit von MQTT selbst ergänzt werden. Dabei handelt es sich um Authentisierung, die mittels Benutzername und Passwort gesteuert wird. Beide Token (Benutzername, Passwort) lassen sich für jeden Client und oder auch für Benutzergruppen definieren. Diese Token müssen bei jedem Login der Knoten bzw. Schnittstellen dem Broker übermittelt werden.

In der unteren Tabelle werden die Protokolle MQTT und HTTP gegenübergestellt, um deutliche Unterschiede zu einem ausgereiften, bestehenden Kommunikationsprotokoll und dessen Vorteile zu veranschaulichen.

Tabelle 2: Unterschiede zwischen MQTT- und HTTP-Protokollen

MQTT	**HTTPS**
Noch geringere Verbreitung, trotzdem OASIS Standard	Etablierte Protokoll, milliardenfach etabliert
Publish/Subscribe-Architektur mit hoher Servicequalität, gut für Push-Anwendungen	Nicht gut für Push-Anwendungen und Servicequalität, aber dafür gut bei Request-/Response-Lösungen
Geringer Ressourcenbedarf von Clients, wodurch Batterieverbrauch reduziert wird	Höhere CPU-Leistung nötig, dadurch erhöhte Batteriekapazität erforderlich
Binäre Nachrichtenübertragung, Header benötigt nur 2 Bytes	Textbasiert, benötigt daher eine höhere Bandbreite

2.2.3 Mikrocontroller, Sensoren und Arduino

Mikrocontroller haben das Internet der Dinge geprägt und deutlich vorangetrieben. Sie bieten im Wesentlichen die Möglichkeit z.B. elektrische Geräte „smart“ zu gestalten, durch integrierte WLAN-Module. Solch ein Mikrocontroller dient als Hardwareplattform (auch Board genannt). Auf dem Markt ist eine große Spannbreite von Boards zu finden, von kleinen und günstigen dazu gehört z.B. Arduino und große und leistungsfähige wie ein Raspberry Pi. Dadurch lässt sich für jede Situation ein geeignetes Board finden und integrieren. (Hüwe, P. & S., S. 63)

Herz eines eingebetteten Systems (vgl. Kap. 2.2.4) sind Mikrocontroller, es gibt aber auch eingebettete Systeme aufgebaut auf Mikroprozessoren. Moderne Systeme basieren im Allgemeinen aber auf Mikrocontrollern. Ein Mikrocontroller besteht generell aus einer zentralen Rechen-, Steuereinheit (CPU), Speicher und anderen Peripheriegeräten, alles zusammen in einem einzigen Schaltkreis. Ein Mikroprozessor umfasst dagegen nur einen CPU in einem Schaltkreis.

Mikroprozessoren werden meist in Computern verwendet und haben eine hohe Rechenleistung und können zahlreiche, verschiedene Aufgaben erledigen. Daher ist der Stromverbrauch im Allgemeinen höher, wodurch oft auch ein externes Kühlungssystem gefordert ist. Mikrocontroller sind für Steuerungsanwendungen konzipiert worden und werden wie erwähnt in der Regel in eingebettete Systeme integriert. (Xiao 2018)

In diesem Projekt wird ein „ESP8266 WEMOS D1 R2 Wifi“ verwendet und ist in der folgenden Abbildung 8 dargestellt. Bei einem „ESP8266-Board“ handelt es sich um einen programmierbaren Mikrocontroller, welcher 2014 auf den Markt gebracht wurde und daher auch schon bei vielen Herstellern zu finden ist. (Hüwe, P. & S., S.43).

Abbildung 8: Abbildung von WeMos D1 R2 Wifi ESP8266

Durch Sensoren können technische Systeme wie Lebewesen wichtige Umgebungsgrößen und Parameter erheben bzw. messen und beobachten. Menschen nutzen dazu ihre Sinnesorgane bzw. fünf Sinne wie Sehen, Hören, Schmecken, Riechen und Fühlen. Einige Größen wie Radioaktivität oder magnetische Felder kann der Mensch mit seinen Sinnen nicht wahrnehmen, andere Lebewesen wie beispielsweise Vögel können magnetische Felder erfassen und orientieren sich daran. Technische Sensoren messen dedizierte technische und physikalische Größen durch geeignete physikalische oder chemische Effekte. Dabei wandelt er die eingehende, zu messende Größe in eine primäre elektrische Größe um. (Hüning 2019)

Der Trend von eingebetteten Systemen führt dazu, dass die Sensorik immer kleiner werden muss (Hüning 2019), da der zur Verfügung stehende Bauraum eins eingebetteten Systems eingeschränkt ist.

In dieser Masterarbeit wird der **Sensor DHT22** verwendet, daher wird im Nachfolgenden dieser Sensor vorgestellt. Der DHT22 ist ein einfacher, kostengünstiger digitaler Temperatur- und Luftfeuchtigkeitssensor und in der nebenstehenden Abbildung 9 dargestellt. Der Sensor verwendet einen kapazitiven Feuchtigkeitssensor und ein Thermistor, um die Umgebungsluft zu messen und gibt das Messergebnis über den Daten-Pin aus, hierbei ist kein analoger Eingang erforderlich. Die Bedienung ist simpel, doch dabei ist zu beachten, dass nur alle zwei Sekunden neue Daten weitergereicht werden. (Adafruit Industries, Unique & fun DIY electronics and kits)

Abbildung 9: Abbildung eines DHT22

In der Welt der IoT ist Arduino nicht mehr wegzudenken. Das Unternehmen gehört zu den prägendsten Unternehmen im IoT-Bereich und den Mikrocontrollern. Der Begriff Arduino umfasst dabei Arduino-Boards (Mikrocontroller), die -Entwicklungsumgebung, die -Sprache und die Arduino-Bibliothek. Zum Verständnis dieser Mastarbeit ist nur die nähere Betrachtung der Arduino-Entwicklungsumgebung bzw. die Arduino-IDE relevant.

Die Arduino-IDE ist zur vereinfachten Programmerstellung entwickelt worden, dabei werden diverse Funktionen und Bibliotheken dem Nutzer bereitgestellt. Das Flashen der Programme ist durch zwei Button „Überprüfen“ und „Hochladen“ einfach realisiert worden. Bei der Arduino-Sprache handelt es sich um die übliche Programmiersprache C und C++ und wird zusätzlich um die diversen Funktionen und Bibliotheken ergänzt. Diese Bibliotheken sollen dem Nutzer die Vielzahl an Möglichkeiten im Rahmen von Arduino bewusst machen und Starterfunktionen liefern, um komplexe Funktionen unterschiedlichster Anwendungen zu erläutern. Weiterhin werden diverse Softwarebibliotheken bereitgestellt. Dazu gehört z.B. die Pub/Sub-Bibliothek, um per MQTT zu kommunizieren. (Hüwe/Hüwe 2019)

Abbildung 10: Sketchaufbau Arduino IDE

Im direkten Programmierumfeld von Arduino wird mit Sketch gearbeitet, die die eigentlichen Programmdateien umfassen. Nach erfolgreicher Überprüfung können diese Sketches direkt auf einen Mikrocontroller hochgeladen werden und werden mit angeschlossener Stromversorgen am Mikrocontroller sofort gestartet.

2.2.4 Embedded Systems

Eingebettete Systeme im Englischen Embedded Systems spielen in der I4.0 in Zusammenhang zu CPS eine große Rolle, da sie inzwischen durch immer bessere Rechenleistungen zur Basis von cyberphysischen Systemen werden (Hüning 2019). Unter Embedded System wird im Allgemeinen ein kleinskaliertes Computersystem gemeint, das in ein Hardwaregerät eingebettet ist. Häufig ist es ein Teil einer Maschine oder eines größeren elektrischen oder mechanischen Systems, dass ausschließlich für bestimmte Aufgaben wie z.B. Echtzeit-Systeme entworfen wurde. Hierbei sind Mikrocontroller das Gehirn der eingebetteten Systeme, welche die bestimmten Operationen steuern. (Xiao 2018)

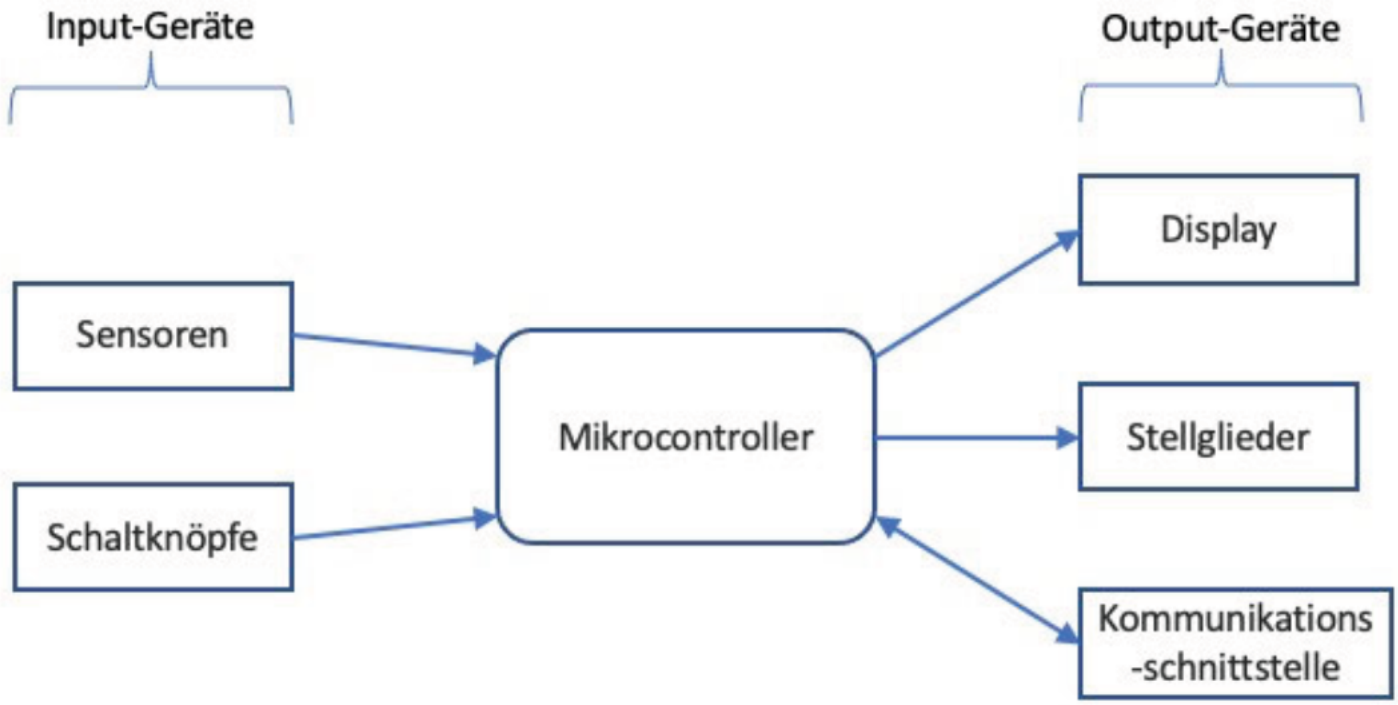


Abbildung 11: Schematischer Aufbau eines Embedded System (Xiao 2018)

Zu den Bestandteilen eines eingebetteten Systems gehören in der Regel ein Rechenhardware, Software und mechanische und oder mechatronische Komponenten. Durch solche Komponenten ist ein eingebettetes System in der Lage Regelungs-, Steuerungs- und Datenverarbeitungsaufgaben durchzuführen (Steven 2019). Die Interaktion mit der Umgebung des Embedded Systems wird durch Sensoren Inputs (Sensoren)

und Outputs (Stellglieder) und realisiert (Hüning 2019). In der obigen Abbildung 11 ist ein schematischer Aufbau eines typischen Embedded System dargestellt. Grundidee dieser Abbildung stammt von P. Xiao.

3 Theoretisches Konzept einer cloudbasierten Produktionssteuerung

In diesem Abschnitt der Arbeit wird das theoretisch erarbeitete Konzept einer cloudbasierten Produktionssteuerung vorgestellt und erläutert. Weiterhin werden die verfolgten Ziele dieses Konzept beschrieben und abschließend Sicherheitsaspekte einer solchen Steuerung im Rahmen von IoT aufgegriffen und erläutert.

3.1 Aufbau und Funktion des theoretischen Konzeptes

Dieses Unterkapitel soll einen Überblick über den theoretischen Hintergrund der praktischen Fallstudie dieser Masterarbeit geben. Bei der Fallstudie handelt es sich um ein Konzept einer cloudbasierten Kommunikationssteuerung im produzierenden Umfeld, das beispielhaft für eine cloudbasierte Produktionssteuerung stehen und auf ein Praxisszenario transferiert werden kann.

Das magische Dreieck lässt sich hierdurch um zwei Ziele erweitern (vgl. Kap. 1.2). Bei den Zielen handelt es sich um die Reaktionsfähigkeit im produzierenden Umfeld und die Transparenz von Daten- und Informationsflüssen im Unternehmen. Dabei unterstützt die I4.0, um solche Möglichkeiten zu schaffen, Daten transparent zur Verfügung zu stellen und daraus schnellere Reaktionsmöglichkeiten zu schaffen. Einer der Umsetzungsmöglichkeiten ist hierbei die Anbindung der wesentlichen Produktionsmittel an einen Cloudserver. So sollen die Sensoren der Produktionsmittel die jeweiligen Daten an den eingebundenen Cloudserver übermitteln, sodass die Daten direkt über den Server zu erreichen sind, wodurch schnellere Analysen durchgeführt bzw. Entscheidungen getroffen werden können.

Die Kernelemente des Shopfloor Managements (vgl. Kap. 2.1.1) werden durch diese Funktionen des Konzeptes unterstützt und weiterentwickelt. Zur Einordnung der Funktionsziele ins SFM sind die unterstützten Kernelemente in der unteren Abbildung 12 der Kernelemente des SFM hervorgehoben.

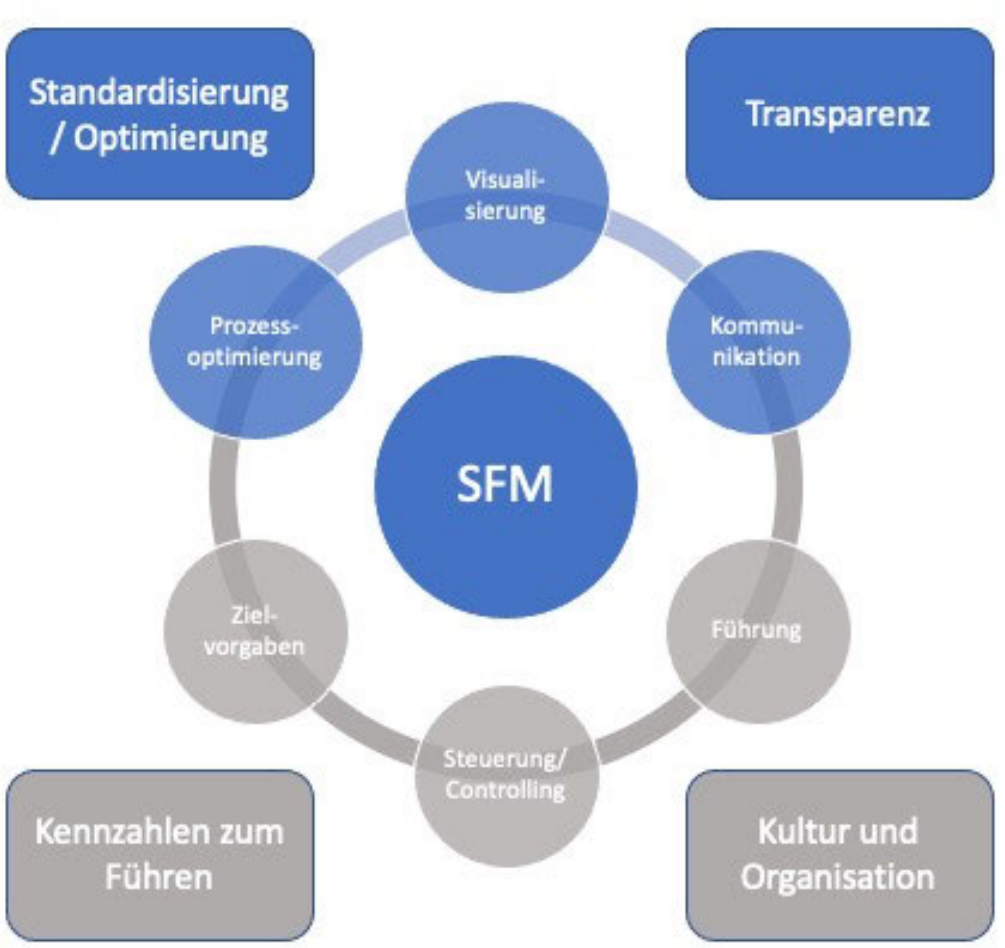


Abbildung 12: Hervorgehobene Kernelemente des SFM

Zu den hervorgehobenen Elementen gehören die Transparenz durch Visualisierung und Kommunikation, sowie die Standardisierung und Optimierung im Bereich der Prozessoptimierung und Visualisierung.

Im weiteren Verlauf des Kapitels werden weitere Ziele des Konzeptes spezifiziert, anschließend wird erläutert, in welchem Kontext das Konzept im produzierenden Umfeld angewendet werden kann, welche notwendigen Faktoren zur Umsetzung nötig sind und welche Schnittstellen zur Umsetzung erforderlich sind.

Die Ziele, die zusätzlich durch dieses Konzept geschaffen werden sollen, sind Standardisierungsmöglichkeiten speziell im Instandhaltungsmanagement wie auch im SFM auf Maschinenstillstände zu reagieren (vgl. Kap. 2.2.1) und des Weiteren auch

Möglichkeiten zur Anbindung an weitere Systeme wie beispielsweise das Supply Chain Management.

Zur theoretischen Umsetzung des Konzeptes ist der Einsatz von Mikrocontroller erforderlich, da sich im Shopfloor Cloud Computing durchsetzen soll, sodass die wesentlichen Sensoren der zu überwachenden Produktionsmittel ihre Daten in ein Cloudsystem übermitteln können. In der folgenden Abbildung stellt die unterste Ebene eine solche Verbindung zwischen den ausgewählten Produktionsmitteln und Mikrocontroller dar, die des Weiteren an einen Cloudserver verbunden sind.

Die internetfähigen Mikrocontroller machen den Datentransfer über einen Cloudserver möglich. Solch ein Cloudserver kann intern betrieben, wie in dieser Abbildung dargestellt, wie auch extern angebunden werden. Dadurch würde sich der Cloudserver aus dem Produktionsbereich verschieben und außerhalb der Grenzen dargestellt werden. Hierbei ist darauf zu achten, dass die übermittelten Daten extern gelagert werden und daher Lizenzverträge mit solch einem Unternehmen von Vorteil sind, um eine klare Datenschutzrichtlinie als Grundlage zur Sicherung der Produktionsdaten und Informationen zu schaffen.

Einer der möglichen Verwendungszwecke bzw. Anschlusspunkte dieses Konzeptes ist z.B. das SFM. Hier soll das Konzept in der Organisation des Maintenance Management unterstützen und Transparenz schaffen wie auch Entscheidungskompetenzen im Bereich von Maschinenstillstandzeiten und Wartungsintervallen und/ oder Umrüstungen von Produktionsmitteln fördern.

Zu den erforderlichen Komponenten eines solchen Konzeptes zählen Mikrocontroller, (die M2M-Kommunikationsprotokoll fähig sind) die Sensoren der wesentlichen Produktionsmittel, dann ein Cloudserver und eine Cloudplattform für das Datenmanagement des Konzeptes. Der theoretische Aufbau und Schnittstellen zwischen den Komponenten sind in der folgenden Abbildung 13 dargestellt. Die Abbildung ist abgeleitet von der Abbildung 5 des vorgestellten Systems eines cloudbasierten cyberphysischen Systems zur adaptiven Produktionssteuerung und Planung (vgl. Kapitel 2.1.2).

In der untersten Ebene der Produktionshalle sind die Mikrocontroller dargestellt und mit den jeweiligen Sensoren verbunden und schicken über eine Wifi-Verbindung die aufgenommenen Daten der Sensoren an den in diesem Fall internen Cloudserver.

Über eine Internetverbindung gelangen die Daten auf eine Cloudplattform und werden in dieser gespeichert und in unterschiedlichen Kategorien organisiert, sodass unterschiedliche Informationen nicht in einem riesigen „Data Lake“ untergehen (vgl. Kap. 2.1.1). Durch die Anbindung ans Internet ist eine weitere Vernetzung außerhalb des Unternehmens möglich. Dadurch können unter anderem externe Instandhaltungsfirmen für z.B. spezielle Produktionsmittel, Zugangsrechte zugewiesen bekommen. Hierdurch können die externen Firmen die nötigen Informationen zum Instandhaltungsauftrag direkt erhalten und dadurch auch schon spezielle Maschinendaten erlangen.

Das entwickelte Konzept kann als ein cyberphysisches System begriffen werden (vgl. Kap 2.1.1), da die Produktionsmittel um physische Systeme ergänzt werden. Dabei werden IoT-Module zur erweiterten Kommunikation von Sensordaten verwendet. Die angeschlossenen Mikrocontroller sollen in diesem Fall als Embedded System der Produktionsmittel verstanden werden.

Hierbei ist zu beachten, dass der Mensch in letzter Instanz als Entscheidungsfinder im cyberphysischen Gefüge dieses Konzeptes agieren sollte (vgl. Kap. 1.2.1). Das Konzept zielt darauf ab, dass eine befugte Person wie beispielsweise ein Shopfloor Manager in letzter Instanz den Prozess des CPS bestätigen oder verwerfen soll.

Als M2M Kommunikationsprotokoll für dieses Konzept eignet sich MQTT, da sich in diesem Konzept weder Nachrichtensender und -empfänger nicht zwangsläufig kennen müssen, sondern nur den Cloudserver bzw. zentralen Broker als Gateway. Weiterer Vorteil dieser Publish/ Subscribe-Architektur ist, dass Sender und Empfänger nicht zur selben Zeit online sein müssen, um Nachrichten zu empfangen. Des Weiteren liefert das MQTT-Protokoll eine hohe Servicequalität und eignet sich gut für Push-Anwendungen wie Sensordaten. Zudem ist der Ressourcenaufwand von MQTT von Clients im Gegensatz zum http-Protokoll geringer und dadurch der Stromverbrauch reduziert, welches den Einsatz von Embedded Systems erleichtert.

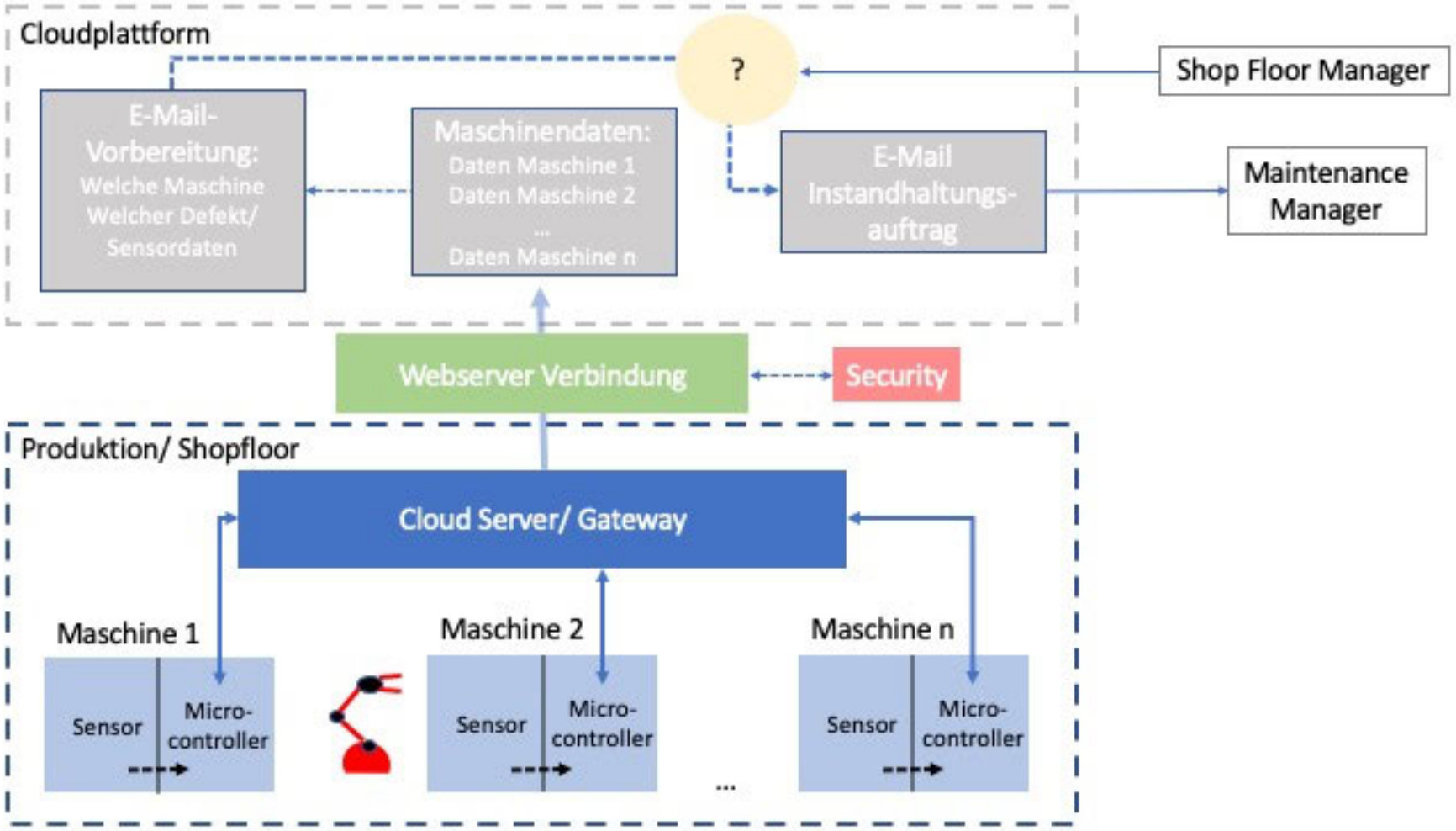


Abbildung 13: Theoretischer Systemaufbau, Schnittstellen und Kommunikationswege des Konzeptes

Die nachfolgende Abbildung soll den Ablauf einer Störung erläutern. Beginnend vom Mikrocontroller über den Broker hin zur Einleitung eines Instandhaltungsprozess, wobei der Prozess in letzter Instanz vom Menschen in diesem Fall eines Shopfloor Manger bestätigt werden muss.

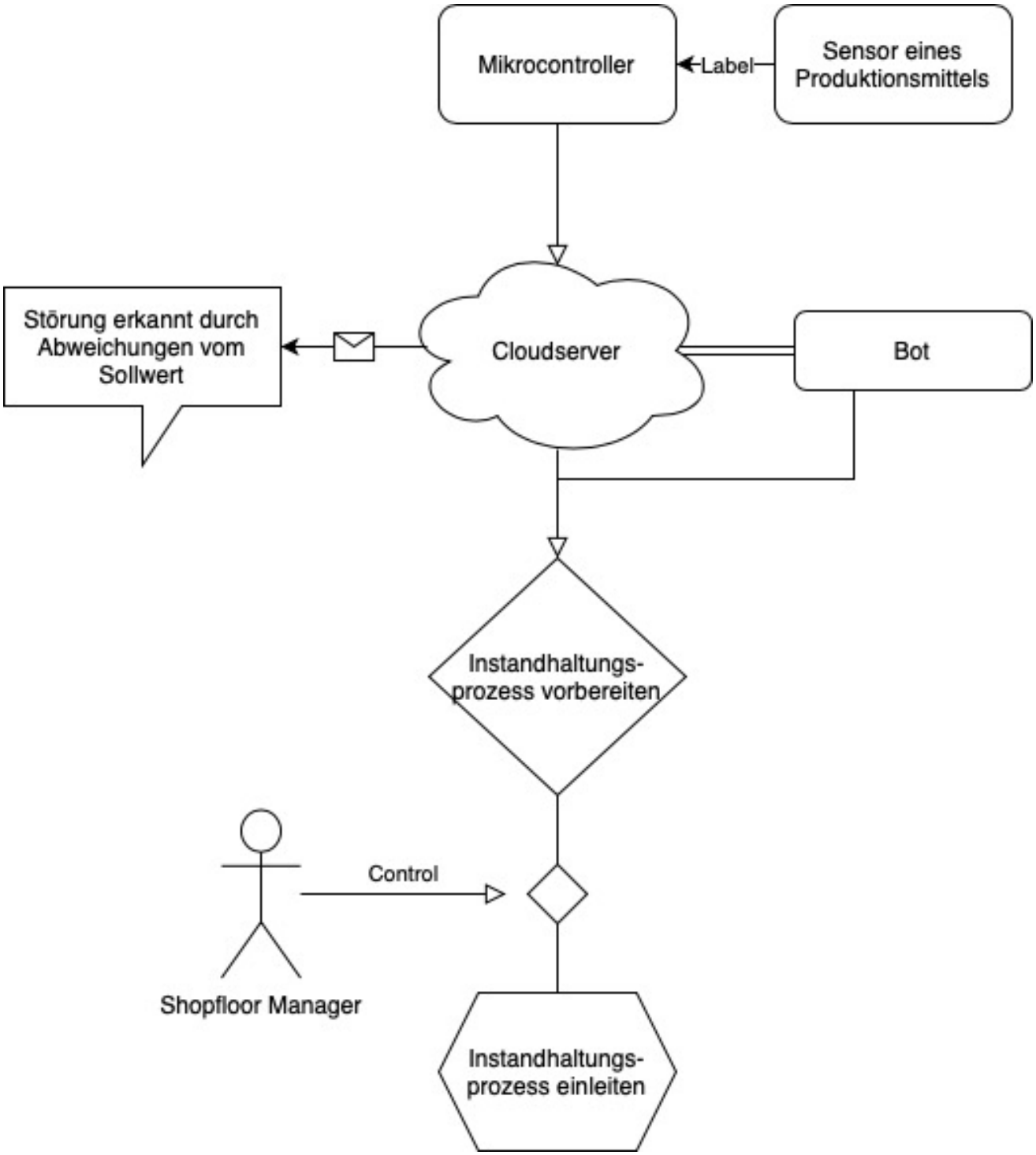


Abbildung 14: Flowchart vom Ablauf des Konzeptes

Um nun abschließend das entwickelte Konzept in Kontext von bestehenden Industrie 4.0 Systeme zu verstehen, wird im Folgenden dieses Konzept zu bestehenden Konzepten (vgl. Kap. 2.1.2) abgegrenzt.

Bei diesem entwickelten Konzept steht der aktive Kommunikationsweg im Vordergrund, da die erfassten Daten der wesentlichen Produktionssensoren direkt in eine Cloud gespeist und abrufbereit bzw. live überwacht werden können. Wenn durch die Überwachung, die durch ein Bot durchgeführt wird, klare Abweichungen (vermehrte Wiederholungen) vom Sollwert erkennbar werden, soll eine Nachricht generiert werden. Diese Nachricht kann beispielsweise per E-Mail verfasst und abgeschickt werden. Als Empfänger soll hier das Maintenance Management erreicht werden, welches intern oder auch extern agieren kann. Bei einer notwendigen externen Instandhaltung ist die Weitergabe von sensiblen Daten zu prüfen. Des Weiteren ist das Konzept auch eine Unterstützung für den Decission Making Process im Shop Floor Management für bspw. Instandhaltungsintervalle und Produktionsanpassungen. Genauso kann dieses Konzept auch in der Anpassung und Einführung von neuen Produkten in der Fertigung unterstützen, da durch das Konzept die Daten transparent dargestellt und schnellere Reaktionswege geschaffen werden können.

3.2 Sicherheit von Cloud Computing und MQTT

In der zunehmenden Vernetzung von Komponenten im Shopfloor spielt Cloud Computing eine wesentliche Rolle (vgl. Kap. 2.1.1). Die Cloud-Services ermöglichen eine zentrale Speicherung von Daten, deren Auswertung und weiterhin den Zugriff auf die Komponenten, wodurch sich aber auch die Cyberrisiken erhöhen. Hierbei ist es erforderlich, die Risiken zu unterscheiden. Erstens ist der Zugang von externen Unternehmen zur Cloud-Architektur ein Risiko, des Weiteren spielen auch die Anbindungen beliebiger Dienste zur Cloud ein Risiko (Dietrich 2021). Daher ist es notwendig, alle Zugänge zu überwachen.

Ziele, die durch Sicherheitsaspekte verfolgt werden sollte, sind die Verfügbarkeit der Dienste und Daten der Cloud Services aufrecht zu erhalten, die Vollständigkeit der Daten zu realisieren und die Geheimhaltung von vertraulichen Daten zu gewährleisten (Vogel-Heuser).

Folgende Aspekte, die zur Sicherheit beitragen, sollten berücksichtigt werden. Dazu zählen nur sichere und überwachte Zugänge bereitzustellen, Zugriffe von Fremdfirmen überprüfen, Cloudanbindungen nur über eine reduzierte Anzahl von Servern ermöglichen, Sicherheitsanforderungen schon in der Entwicklung erstellen und Sicherheitsaudits durchführen wie auch eigene Subnetze für den Shopfloor realisieren (Dietrich 2021).

Folgende technologische und organisatorische Lösungsansätze, die zur Sicherheit im Cloud Computing im Kontext von I4.0 beitragen und zur Umsetzung der Ziele herangezogen werden, werden im Folgenden drei näher vorgestellt.

Einer dieser Lösungen ist der Ansatz eines rollenbasierten Zugriffssystems innerhalb der Cloudplattform (Vogel-Heuser et al. 2017) . Wie es schon aus dem Namen hervorgeht, werden den Nutzern definierte Rollen entsprechend ihres Aufgabenfeldes und der Zuständigkeit zugeteilt. Über die entsprechende Rolle werden den Nutzern die Zugriffsrechte erteilt und müssen nicht speziell auf jeden einzelnen Nutzer zugeschnitten werden.

Einen ähnlichen Ansatz zum Schutz der Daten ist ein attributbasiertes Verschlüsselungssystem. In diesem System werden die Daten verschlüsselt abgelegt und können

nur von Nutzern mit einem Schlüssel, der eine gleiche Anzahl von Attributen aufweist, die Daten entschlüsseln (Vogel-Heuser et al. 2017). Vorteil dieses Ansatzes ist es, dass die Zugriffsrechte aus einem komplexen Softwaresystem in die verschlüsselten Daten selbst verlegt werden und überhaupt eine Verschlüsselung der eigentlichen Daten vorliegt. Nachteil dabei ist, dass die Umsetzung aufwendiger ist.

Auch die Datenübertragung kann durch etablierte Verfahren wie VPNs verschlüsselt werden. Durch dieses Verfahren können einzelne Verbindungen zwischen Komponenten oder gesamte Netzwerkverbindungen verschlüsselt realisiert werden (vgl. Kap 2.2.2). Hierdurch können bestehende Kommunikationsprotokolle getunnelt werden, wodurch keine neuen, konzipierten Protokolle entstehen müssen, sondern bestehende Protokolle über abgesicherte Verbindungen weiterverwendet werden können (Vogel-Heuser et al. 2017).

Weitere Lösungsansätze, die weiter erforscht und ausgearbeitet werden lauten (Vogel-Heuser et al. 2017):

- Datenintegrität durch sichere Hardware-Module (Beispiele sind NEA und IMA)
- Produkt- und Know-how-Schutz (Beispiele hierfür sind PUFs)
- Erhöhung der Verfügbarkeit und Integrität von Daten in der Cloud (Cloud-Datensystem Iris)

Eine weitere Vertiefung dieser Ansätze ist nicht vorgesehen, da dies den Rahmen der Arbeit überschreiten würde, aber trotzdem in diesem Kontext Erwähnung finden sollten.

Im Hinblick auf den Einsatz von MQTT als Kommunikationsprotokoll in diesem Konzept, bietet dies folgende Möglichkeiten im Kontext von Sicherheit (vgl. Kap. 2.2.2). Über das Kommunikationsprotokoll kann eine eigene Authentifizierung von Nutzern realisiert werden und weiterhin ist es möglich, dass Kommunikationsprotokoll MQTT um Standards wie TLS beispielsweise zu erweitern, damit die Kommunikation weiter verschlüsselt wird.

4 Praktische Fallstudie des theoretischen Konzeptes

Dieses Kapitel umfasst die entwickelte Fallstudie zum vorangegangenen Konzept. Diese Fallstudie soll die Umsetzung vereinfacht darstellen und weitere Aspekte, die zur Umsetzung beachtet werden sollten verdeutlichen.

4.1 Ausarbeitung der praktischen Fallstudie

Dieses Kapitel gibt dem Leser einen Einstieg zur praktischen Umsetzung des theoretischen Konzeptes im Rahmen einer Fallstudie und erläuterte die praktische Umsetzung in drei Schritten.

4.1.1 Einstieg und Erläuterung zum bestehenden Konzept

Das Konzept dieser Masterarbeit soll zur Unterstützung im Shopfloor Management dienen. Ziel hierbei ist es, die Transparenz und Verfügbarkeit von Maschinendaten, Informationsflüssen und Entscheidungsmöglichkeiten in einer Produktion zu steigern. Damit hierdurch Stillstandzeiten von beispielsweise Maschinen schneller erkannt und behoben werden können.

Grundidee dieses Konzeptes ist es, die geschaffenen Möglichkeiten durch die I4.0 zu nutzen, dazu zählen wesentliche Bereiche wie IoT und auch Cloudmanagement. Indes sollen Maschinendaten durch Cloudanbindungen offengelegt werden und müssen nicht mehr vor Ort an jeweils jeder Maschine einzeln ausgelesen werden.

Für dieses Konzept soll eine Fallstudie entworfen werden, in der wesentliche Sensordaten in einen MQTT-Broker gespeist werden. In diesem Broker sollen unterschiedliche Topics zur Struktur der Daten genutzt werden (vgl. Kap. 2.2.2). In diesem Konzept werden im Wesentlichen zwei Topics verwendet. Das erste Topic gilt der Sensordaten der Produktionsclients, die in diesem System überwacht werden sollen. Hier sollen alle Daten über jeden Client zu finden sein, dabei sollen sich die Daten im Wesentlichen in drei Punkten unterscheiden. Nämlich dem jeweiligen Client, also z. B. um welche

Maschine es sich handelt, dann um was für Daten es sich handelt, z. B. Temperatur, Drehzahl o. Ä. und zuletzt zu welcher Zeit und von welchem Ort die Daten kommen.

Die folgende Abbildung 15 verdeutlicht das Konzept am Beispiel eines Produktionsszenarios. Hierzu werden zwei Ebenen herangezogen, die Produktionsebene und die Cloudplattform. Auf der Produktionsebene wird hierfür ein Produktionsclient bzw. Produktionsmittel wie z. B. eine Maschine eingesetzt und davon werden wesentliche Sensoren mit Mikrocontrollern verbunden. Über zwei Kanäle sind die Mikrocontroller mit dem Broker verbunden, um Daten zu veröffentlichen und zu empfangen. Jeder dieser Mikrocontroller published seine empfangenen Daten der Sensoren im Topic „…“

Subscriber dieses Topics soll in diesem Konzept ein „Bot“ sein, der die Daten überprüft und auf Abweichungen zu Soll-Zuständen achtet. Bei einer deutlichen Abweichung zu einem vorgegebenen Soll-Wert soll der Client eine automatische E-Mail bzw. Instandhaltungsauftrag erzeugen. Dieser Auftrag soll dann an eine befugte Person wie beispielsweise einem Shopfloor Manager weitergeleitet und überprüft werden. Nach erfolgreicher Überprüfung wird der Instandhaltungsvertrag in der Cloud gespeichert und die erforderliche Instandhaltungsfirma kontaktiert, die entweder intern oder auch extern ablaufen kann.

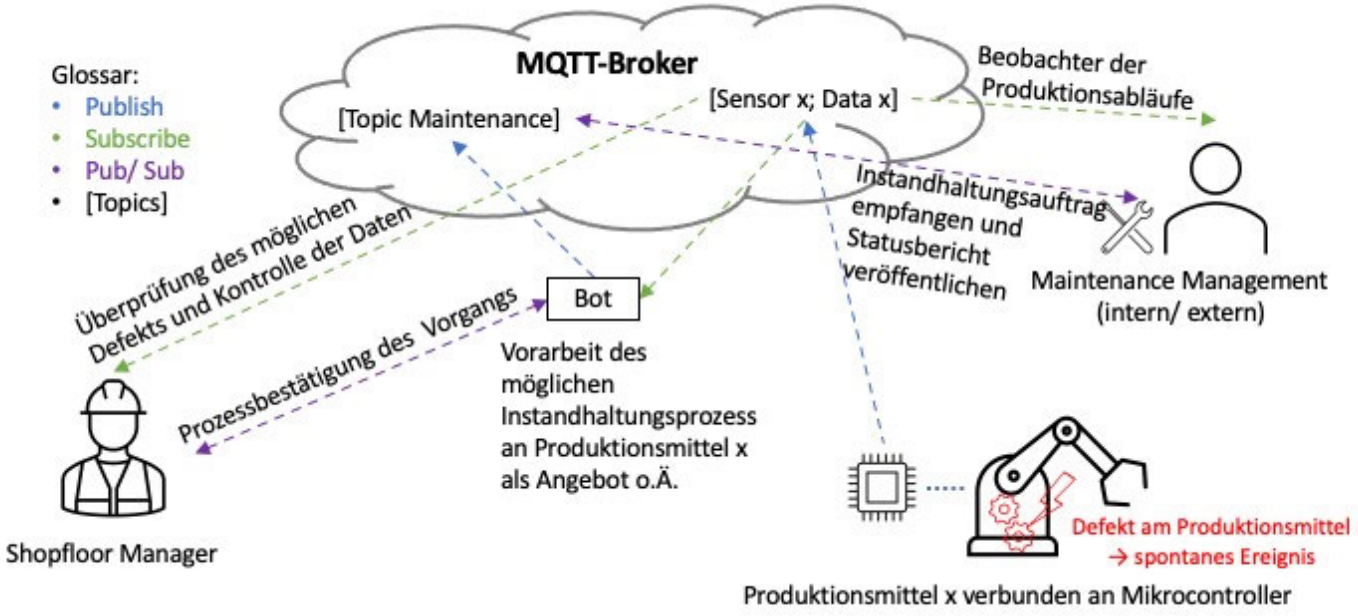


Abbildung 15: Theoretisches Produktionsszenario des Konzeptes

Die untere Abbildung 16 soll zusätzlich den Ablauf der Fallstudie vereinfacht visualisieren, dabei ist der Ablauf in einem Flowchart dargestellt und angelehnt an die Abbildung 15 Flowchart vom Ablauf des Konzeptes.

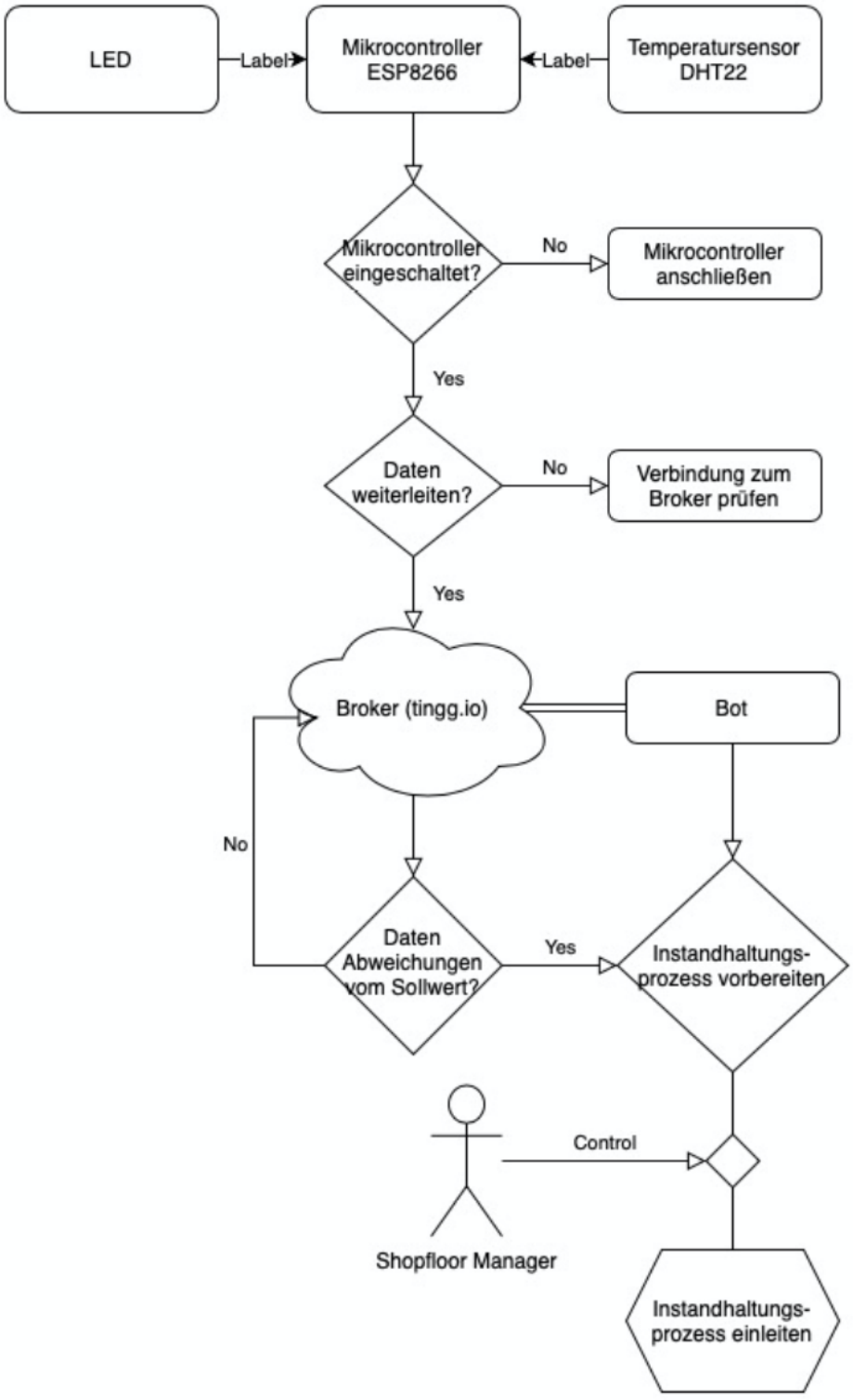


Abbildung 16: Funktionsablauf der Fallstudie als Flowchart

Da MQTT eine hohe Servicequalität liefert, ist der Einsatz dieser Service of Quality Stufen der Nachrichten ein gutes Mittel zur Sicherung des Datenverkehrs (vgl. Kap. 2.2.2). Hierbei können alle QoS Stufen von 0 – 2 in diesem Konzept eingesetzt werden.

Die QoS 0 Stufe lässt sich auf die Sensordaten anwenden, da in diesem Fall eine ausschließliche Push-Anwendung vorliegt, bei der es nicht erforderlich ist, dass jede einzelne Nachricht (Sensordaten) zuverlässig den Empfänger ohne Lücken erreichen. Eine sogenannte „fire and forget“ Übertragung, bei der der Empfang einer Nachricht keine Bestätigung erfordert. Ein weiterer Vorteil dieser QoS 0 Stufe ist es, dass hierbei die geringste Netzbelastung entsteht und sich daher gut für Push-Anwendungen wie Sensordaten eignet.

Die zweite Stufe QoS 1 ist für die Funktionen des Bot geeignet, da diese Nachrichten zugestellt werden müssen, da ansonsten keine Störmeldung erfasst bzw. erkannt wird durch das CPS. Die letzte und höchste Stufe QoS 2 ist im Falle einer wichtigen exakten einmaligen Zustellung einer Nachricht anzuwenden, wie beispielsweise bei einem Client wie ein mobiles Endgerät, worüber eine Nachricht im Mobilfunk verschickt werden soll. Wobei auch in vielen Fällen QoS 1 ausreicht, um Nachrichten mit Sicherheit beim Empfänger anzubringen, falls Störungen und Unterbrechungen im Mobilfunknetz auftreten können.

An das theoretische Produktionsszenario ist der praktische Teil dieser Masterarbeit angeknüpft, nämlich die zu entwickelnde Fallstudie. Diese soll die mögliche Umsetzung des Konzeptes praktisch verdeutlichen, und auf das vorgestellt theoretische Produktionsszenario übertragen werden können.

Die Umsetzung dieses Konzeptes ist durch die Fallstudie in drei Einzelschritte aufgeteilt, um das Ergebnis einer gesamten Vernetzung und Kommunikation vom MQTT-Broker übers Web bis hin zum Sensor am Mikrocontroller, das den Datentransfer über eine Cloud ermöglicht und eine Steuerung und Prüfung der Daten und Informationen ermöglicht.

4.1.2 Umsetzung der Fallstudie in drei Schritten

Dieses Unterkapitel beschreibt die Umsetzung der Fallstudie in drei Schritten hin zu dem cyberphysischen System.

Erster Schritt der Kommunikation eines Clients lokal via Internetverbindung mit einem Broker:

Im ersten Schritt ist es nötig, eine Verbindung zwischen einem Client und einem MQTT-Broker zu schaffen. In dieser Fallstudie ist der verwendete Client ein ESP8266 Mikrocontroller und als Cloudserver wird der Public MQTT-Broker von „hivemq.com“[1] verwendet. Auf der Webseite sind die Brokerdaten und die eigene Client-ID hinterlegt, um die Verbindung zum Server herzustellen. In der folgenden Abbildung 17 ist der Browser Client des Public HiveMQ MQTT Broker dargestellt.

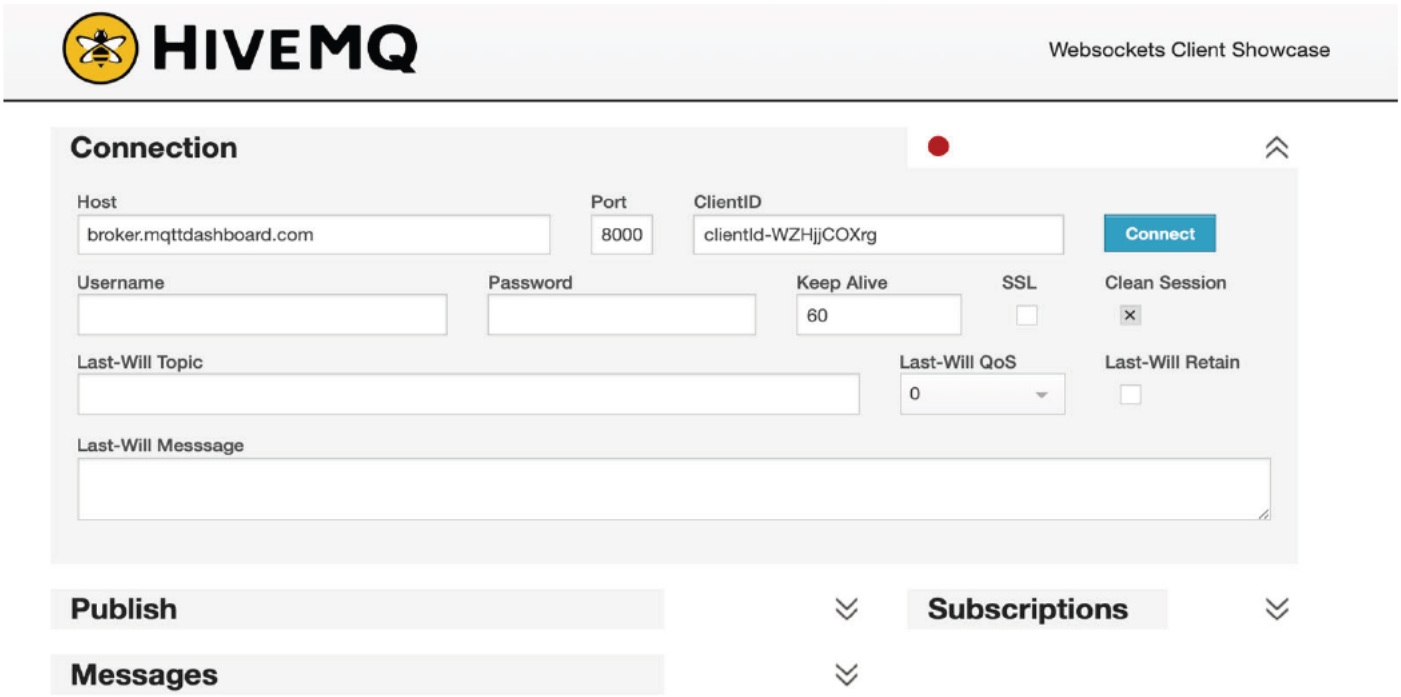


Abbildung 17: Ausschnitt des Browser Clients „Public HiveMQ MQTT Broker“ (MQTT Websocket Client)

[1] Webadresse dieses Public MQTT-Brokers lautet: http://www.hivemq.com/demos/websocket-client/

Ein Topic kann über den Browser Client unter dem Reiter „Subscriptions“ erzeugt und mit dem Mikrocontroller an beide Pub- und Sub-Kanäle angebunden werden, um Daten zu empfangen und zu verschicken. Über den Reiter „Publish“ können Nachrichten verschickt und über den „Massage“ Reiter Nachrichten empfangen bzw. kontrolliert werden (MQTT Websocket Client).

Auf den ESP8266 Client ist das Programm zur Kopplung an einen MQTT Broker für den ersten Schritt aufgespielt worden. Die Übertragung des Programms erfolgt über Arduino. Das Programm ist im Anhang A dargestellt. Dabei ist der Client an das Topic des Public MQTT-Brokers von „hivemq“ gebunden, empfängt und veröffentlicht Daten. Der Client veröffentlicht Einsen oder Nullen im vorgegebenen Intervall, dabei spielt das Empfangen von Daten eine große Rolle für diesen Client. Bei dem jeweiligen Empfangen einer Eins und bei dem dauernden Senden einer Null springt der Client um und versendet im vorgegebenen Intervall Einsen. Dies ist jeweils auch andersherum möglich.

Im dritten Schritt wird das Programm zur Umsetzung der Fallstudie vollständig vorgestellt und erläutert. Jenes Programm weist auch Teile des ersten Schrittes auf.

Zweiter Schritt Sensor Verbindung mit Mikrocontroller und Messdaten erfassen:

Der zweite Schritt umfasst das Programm für den Mikrocontroller zur Verbindung eines Sensors und Schaltung einer LED. Die Schaltung soll über das Empfangen von einer Eins eingeschaltet werden und durch eine Null ausgeschaltet werden. Das Programm für diesen Schritt ist im Anhang B hinterlegt. Dieser Schritt ist notwendig, um die Kommunikation zwischen Mikrocontroller und Sensoren zu schaffen. Dieser Schritt ist beispielhaft für die „untere Ebene“ in der Produktionshalle, welches in der Abbildung des theoretischen Systemaufbaus des Konzeptes dargestellt ist.

In der Fallstudie wird ein Temperatursensor der Bezeichnung DHT22 verwendet und die eingebaute LED auf dem Board verwendet. Der Temperatursensor wird am ESP8266 über die Pins GND, Data und VCC angeschlossen. Im Programm ist die Bezeichnung des verwendete Data-Pins zu hinterlegen, sodass die Sensordaten erfasst werden können. Die Darstellung der Sensordaten erfolgt in diesem Schritt über den Monitor der Support Software Arduino.

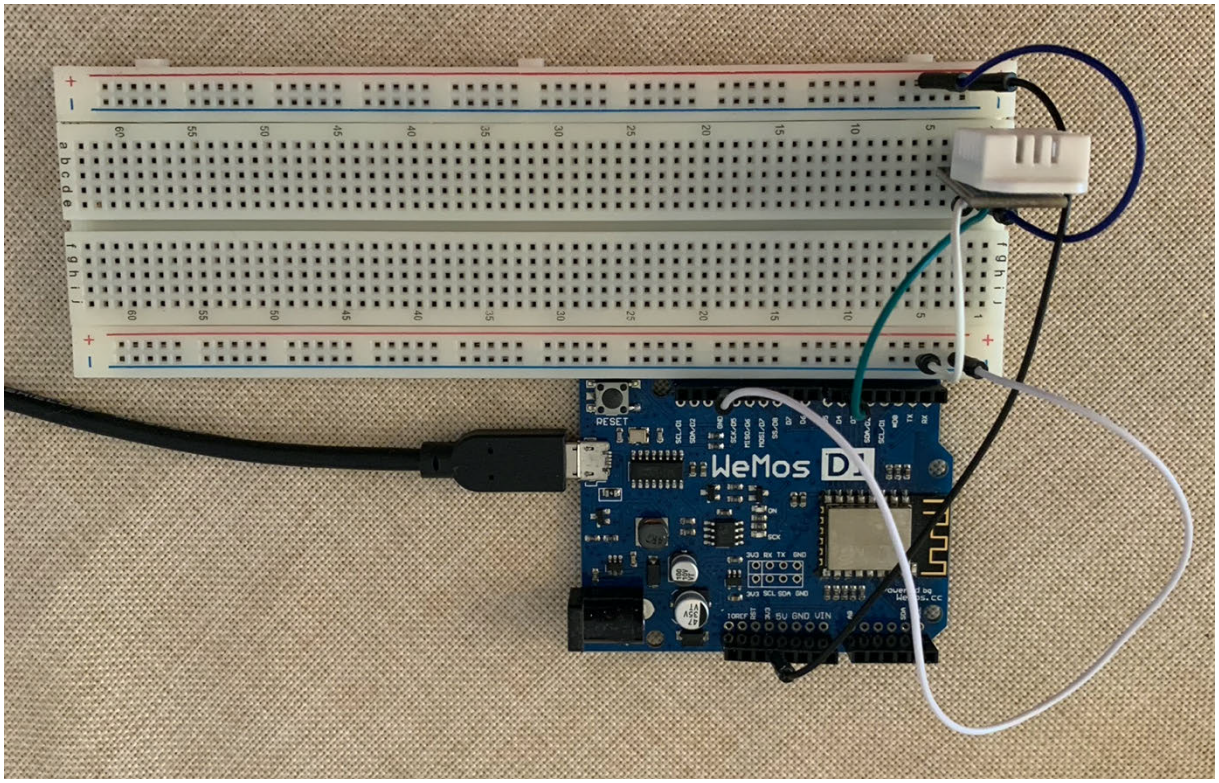


Abbildung 18: Esp8266 mit DHT22 und Steckbrett

Dritter Schritt Sensor-Mikrocontroller Verbindung Broker und unterschiedlichen Topics zur Unterscheidung von Pub/ Sub (Steuerung/ Kontrolle der Sensoren):

Im dritten Schritt werden die ersten beiden Schritte verknüpft und stellt die eigentliche praktische Fallstudie zum theoretischen Konzept des entwickelten Produktionsszenarios dar. Hierbei werden die Clients direkt mit einem MQTT-Broker und den passenden Topics verbunden, um Daten zu versenden und zu empfangen. In dieser Fallstudie wird der Broker von „tingg.io“ verwendet, hierüber werden die Topics erzeugt, gesteuert und überwacht.

Diese Fallstudie soll das Produktionsszenario aus dem vorangegangenen Kapitel aufgreifen und vereinfacht simulieren und darstellen. Für diese Fallstudie werden zwei ESP8266 Mikrocontroller verwendet, an die jeweils eine LED und ein Temperatur- und Luftfeuchtigkeitssensor angeschlossen sind. Des Weiteren ein Smartphone als Überwachungstool durch eine Internetverbindung an den Broker und eine Wifi-Verbindung zur Anbindung ans Internet. Im Broker werden vier Topics erzeugt. Dabei werden zur beispielhaften möglichen Steuerung eines Produktionssensors eine LED verwendet, die an und ausgeschaltet werden kann und laufend ihren Status über den gleichen Client sendet. Das dritte und vierte Topic dienen der aktiven Sensordatenübermittlung. Über diese Topics werden die Sensordaten des DHT22 live erfasst. Diese sollen beispielhaft für wesentliche Produktionssensoren stehen, die überwacht bzw. welche Sensordaten dokumentiert werden sollen. Um im Fall der Fälle einen Stillstand oder auch kritische Zustände eines Produktionsmittels feststellen zu können.

So erschafft diese Fallstudie vereinfach eine Transparenz von wesentlichen Sensordaten, die jeder Zeit abrufbereit sind und von CPSs überwacht werden können. Dazu bietet es die Möglichkeit, ein geeignetes CPS so zu nutzen, dass die gesammelten Daten es so auswerten, dass für zukünftige Ausfälle frühere Prognosen über mögliche Ausfälle getroffen werden können. Ein solcher Fall wäre Beispielsfall möglich, wenn nach einem immer wiederkehrenden Zeitintervall Ausfälle entstehen, wie beispielsweise nach einer gewissen Umstellung des Produktionszyklus oder nach einer Umstellung auf ein neues Produkt.

Der „tingg console“ MQTT-Broker bietet die Vorteile, dass die Daten auch in Charts abrufbereit und überwacht werden können. Dadurch können Sprünge im Datenfluss

deutlich gemacht werden und auch Abhängigkeiten zwischen den unterschiedlichen Daten erkennbar sind. Im Hinblick auf Sicherheit in der Welt von IoT ist die Anmeldung bzw. Authentifizierung über einen Account und Passwort von diesem Broker gefordert. Weitere Aspekte zur Sicherheit wurden im Rahmen dieses Konzeptes nicht berücksichtigt, da dies den Rahmen dieser Arbeit übersteigen würde.

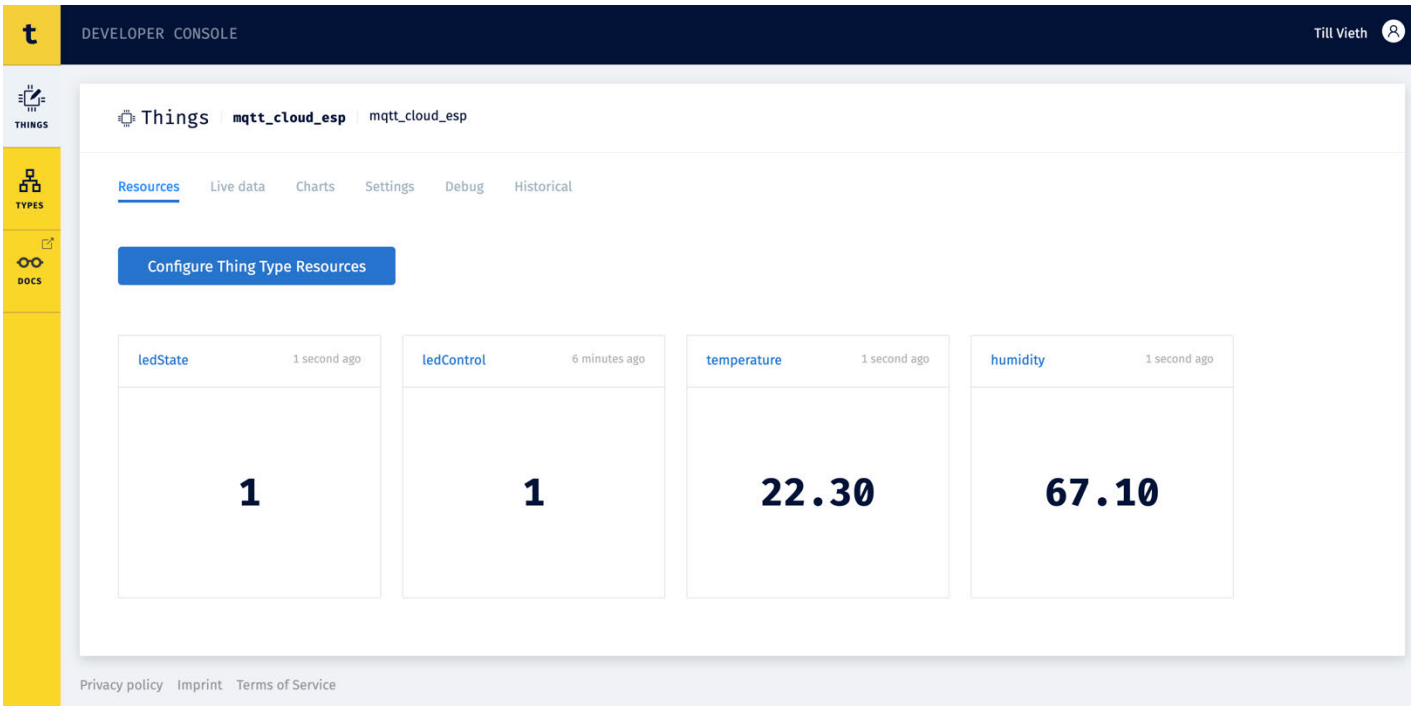


Abbildung 19: Ausschnitt des Brokers Developer Console von tingg.io

In der obigen Abbildung 18 ist der genutzten Broker „Developer Console“ von tingg.io dargestellt. Dargestellt sind die vier Topics dieses Konzeptes, die unter den Ressourcen des Brokers zu finden sind, die auch in der Abbildung zum schematischen Aufbau der Fallstudie dargestellt sind. Dieser Broker von tingg.io visualisiert zusätzlich die Daten in Echtzeit und aufgezeichnete Daten in Charts. Die nächste Abbildung 19 stellt die visualisierten Daten über tingg.io dar. In den Charts werden die unterschiedlichen Sensordaten und Informationen durch Farben unterschieden.

Abbildung 20: Daten-Charts über tingg.io

Die nachfolgende Abbildung 20 stellt den schematischen Aufbau der Fallstudie und deren Sub-/Pub-Architektur zu dem MQTT-Broker dar. Dabei werden zwei Mikrocontroller eingesetzt. Ein Client veröffentlicht (published) die Daten des Temperatur- bzw. Luftfeuchtigkeitssensors an den Broker bzw. das jeweilige Topic der Sensordaten (Temperatur und Luftfeuchtigkeit). Der andere Client empfängt (Subscriber) Daten aus dem Topic „ledControl“ und published den Status der LED im Intervall von 5 Sekunden, ob die LED ein- oder ausgeschaltet ist. Über das Topic „ledControl“ kann die LED ein und ausgeschaltet werden, hierbei schaltet der Mikrocontroller die LED ein, wenn die Ziffer „1“ im Topic veröffentlicht wird und schaltet die LED aus. Wenn die Ziffer „0“ erscheint. Das Smartphone fungiert in diesem Fall auch als Client und kann zur Überwachung der einzelnen Topics bzw. Sensoren herangezogen werden. Zusätzlich ist es möglich, dass über diesen Client die LED geschaltet werden kann, indem die Ziffern „1“ oder „0“ im Topic „ledControl“ veröffentlicht werden.

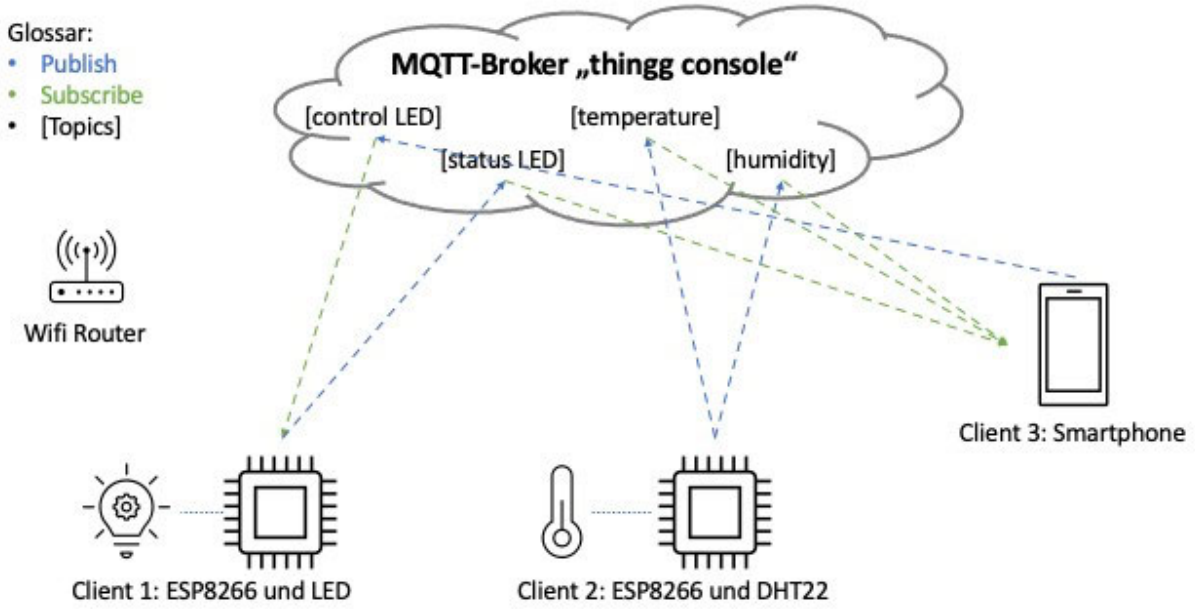


Abbildung 21: Schematischer Aufbau der praktischen Fallstudie

Um die beispielhaften Parallelen dieser Fallstudie zum theoretischen Konzept einordnen zu können, folgt die Split-Darstellung der vorangegangenen Abbildungen der schematischen Darstellungen zum theoretischen Konzept und der Fallstudie. Die unterste Ebene siegelt die Produktionshalle wieder. Hier werden Mikrocontroller mit den jeweiligen ausgewählten Sensoren verbunden, um diese Sensordaten an die Cloud anbinden zu können. Dies ist In der Fallstudie durch zwei ESP8266 Mikrocontroller als Clients verdeutlicht und dem daran angeschlossenen DHT22 Sensor und einer LED. Beide angekoppelten Geräte stehen beispielhaft für Produktionsmittel. Der Wifi Router schafft die Internetverbindung zum MQTT-Broker der Clients.

Der „thingg.io Developer Console“ Broker selbst ist die Ebene der Cloudplattform aus dem theoretischen Konzept, das daran angekoppelte Smartphone soll als Shopfloor Manager agieren, die die Überwachungsschnittstellen im theoretischen Konzept übernehmen sollen.

In den folgenden vier Abbildungen 21-24 ist das Programm zur praktischen Fallstudie dargestellt, dazu wird im Folgenden Teile dieses Programms erläutert.

In der nebenstehenden „while“-Schleife wird die Verbindung zu dem verwendeten Wifi aufgebaut bzw. versucht zu verbinden. Nach erfolgreicher Verbindung soll eine Bestätigung „Wifi connected“ erfolgen und die Daten der Wifi-Adresse ausgegeben werden.

In der darauffolgenden „void callback“-Routine wird die Verbindung zu den Topics bzw. zur Ausgabe über den seriellen Monitor geknüpft. Des Weiteren wird der Status der LED an das Topic „ledState“ gesendet durch die Routine „client.publish“.

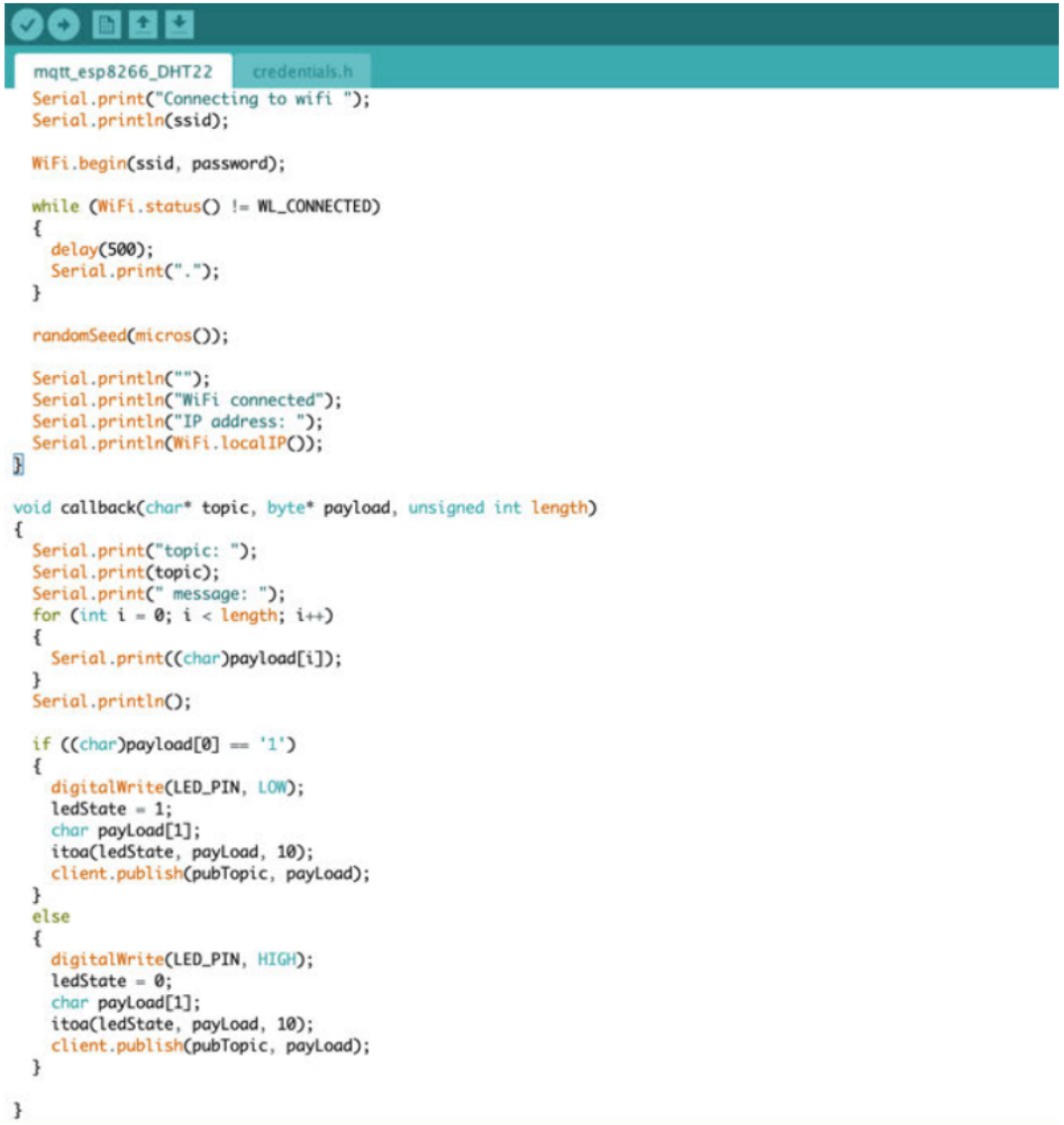


Abbildung 22: Fallstudien-Programm erster Abschnitt

Die „void reconnect“- Schleife, die am Anfang der Abbildung zu finden ist, greift bei dem Verlust der Verbindung zum Broker ein und versucht die Verbindung alle 5 Sekunden wiederherzustellen.


```
void reconnect()
{
  while (!client.connected())
  {
    Serial.print("connecting to mqtt.tingg.io...");

    if (client.connect(mqttDeviceId, mqttUsername, mqttPassword))
    {
      Serial.println("connected.");
      client.subscribe(subTopic);
    } else
    {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}

void setup()
{
  pinMode(LED_PIN, OUTPUT);
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqttServer, 1883);
  client.setCallback(callback);

  dht.begin();
  // Print temperature sensor details.
  sensor_t sensor;
  dht.temperature().getSensor(&sensor);
  Serial.println(F("------------------------------------"));
  Serial.println(F("Temperature Sensor"));
  Serial.print  (F("Sensor Type: ")); Serial.println(sensor.name);
  Serial.print  (F("Driver Ver:  ")); Serial.println(sensor.version);
  Serial.print  (F("Unique ID:   ")); Serial.println(sensor.sensor_id);
  Serial.print  (F("Max Value:   ")); Serial.print(sensor.max_value); Serial.println(F("°C"));
  Serial.print  (F("Min Value:   ")); Serial.print(sensor.min_value); Serial.println(F("°C"));
  Serial.print  (F("Resolution:  ")); Serial.print(sensor.resolution); Serial.println(F("°C"));
  Serial.println(F("------------------------------------"));
  // Print humidity sensor details.
  dht.humidity().getSensor(&sensor);
  Serial.println(F("Humidity Sensor"));
  Serial.print  (F("Sensor Type: ")); Serial.println(sensor.name);
  Serial.print  (F("Driver Ver:  ")); Serial.println(sensor.version);
  Serial.print  (F("Unique ID:   ")); Serial.println(sensor.sensor_id);
  Serial.print  (F("Max Value:   ")); Serial.print(sensor.max_value); Serial.println(F("%"));
```

Abbildung 23: Fallstudien-Programm zweiter Abschnitt

Durch die „void setup“ Routine am Anfang des Programms bzw. „client.setServer“ wird die Verbindung zum Broker an den Port 1883 aufgebaut und durch „client.setCallback(callback)“ wird die callback-Funktion aufgerufen, wenn Nachrichten empfangen werden.

Die nebenstehende „dht.begin“ Routine steuert die Ausgabe der Sensordaten wie in diesem Fall der Temperatur und der Luftfeuchtigkeit, dazu wird jeweils der Name des Sensors ausgegeben, in diesem Fall DHT22.

mqtt_esp8266_DHT22 | credentials.h

```
void setup()
{
  pinMode(LED_PIN, OUTPUT);
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqttServer, 1883);
  client.setCallback(callback);

  dht.begin();
  // Print temperature sensor details.
  sensor_t sensor;
  dht.temperature().getSensor(&sensor);
  Serial.println(F("------------------------------------"));
  Serial.println(F("Temperature Sensor"));
  Serial.print  (F("Sensor Type: ")); Serial.println(sensor.name);
  Serial.print  (F("Driver Ver:  ")); Serial.println(sensor.version);
  Serial.print  (F("Unique ID:   ")); Serial.println(sensor.sensor_id);
  Serial.print  (F("Max Value:   ")); Serial.print(sensor.max_value); Serial.println(F("°C"));
  Serial.print  (F("Min Value:   ")); Serial.print(sensor.min_value); Serial.println(F("°C"));
  Serial.print  (F("Resolution:  ")); Serial.print(sensor.resolution); Serial.println(F("°C"));
  Serial.println(F("------------------------------------"));
  // Print humidity sensor details.
  dht.humidity().getSensor(&sensor);
  Serial.println(F("Humidity Sensor"));
  Serial.print  (F("Sensor Type: ")); Serial.println(sensor.name);
  Serial.print  (F("Driver Ver:  ")); Serial.println(sensor.version);
  Serial.print  (F("Unique ID:   ")); Serial.println(sensor.sensor_id);
  Serial.print  (F("Max Value:   ")); Serial.print(sensor.max_value); Serial.println(F("%"));
  Serial.print  (F("Min Value:   ")); Serial.print(sensor.min_value); Serial.println(F("%"));
  Serial.print  (F("Resolution:  ")); Serial.print(sensor.resolution); Serial.println(F("%"));
  Serial.println(F("------------------------------------"));
}

void loop()
{
  if (!client.connected())
  {
    reconnect();
  }
  client.loop();

  long now = millis();
  if (now - lastMsg > 5000)
  {
    lastMsg = now;
    char payLoad[1];
    itoa(ledState, payLoad, 10);
```

Abbildung 24: Fallstudien-Programm dritter Abschnitt

In dieser loop() Schleife wird die Verbindung zum Broker überprüft und im Falle keiner Verbindung wird versucht die Verbindung wieder aufzubauen, dies geschieht über „if (!client-connected())“. Über client.publish(topic, payload) werden die entsprechenden Sensordaten entweder Temperatur oder Luftfeuchtigkeit publiziert. Dazu wird entweder ein Error bei einem Fehler oder die eigentlichen Daten ausgegeben.

Ähnliche Programme zu dem Thema MQTT gesteuerte IoT-Anwendungen sind im GitHub zu finden. GitHub ist eine Internetplattform für Softwareentwickler und -projekte, die über diesen Dienst ausgetauscht werden können.

Abbildung 25: Fallstudien-Programm vierter Abschnitt

4.2 Bewertung und Handlungsempfehlungen zur Fallstudie

Im Folgenden werden drei Fragen herangezogen, die bei der zusammenfassenden Bewertung der Fallstudie zur Umsetzung des entwickelten theoretischen Konzeptes beitragen sollen und welche möglichen Hinweise zur weiteren Realisierung in der Praxis betrachtet werden sollten.

Lässt sich das entwickelte Konzept exemplarisch auf die Fallstudie anwenden?

Die Fallstudie wurde entwickelt, um das theoretische Konzept exemplarisch darzustellen und eine mögliche Umsetzung in die Praxis zu verdeutlichen. Zur Umsetzung des Konzeptes wurde in der Fallstudie ein ESP8266-Board und ein das MQTT-Protokoll verwendet. Als zentrale Schnittstelle in der Fallstudie wurde der MQTT-Broker von Tingg verwendet. Durch die Verwendung des tingg.io Brokers werden die Daten direkt visualisiert. Die Visualisierung wird dabei über Charts durchgeführt, wie in der vorherigen Abbildung 19 dargestellt.

Die Umsetzung empfangener Daten via Internet bzw. über die Bereitstellung eines Cloud-Services ist exemplarisch durch die praktische Fallstudie gelungen. Die Daten werden über einen angeschlossenen Mikrocontroller und durch die Verwendung des MQTT-Kommunikationsprotokoll ausgelesen und weitergeleitet. Durch die IoT-Erweiterung des Mikrocontrollers ist das Cloud Computing in dieser Fallstudie realisiert worden und die Daten vom Mikrocontroller an den tingg.io Broker weitergeleitet.

Außerdem schafft die Einbindung des Brokers eine dezentrale Produktionssteuerung, da der Zugriff auf die Daten nicht ortsgebunden ist, sondern via Cloud-Service über eine Internetverbindung abrufbereit sind.

Welche Ziele werden mit diesem Konzept durch die praktische Fallstudie verfolgt und umgesetzt?

In diesem Konzept ist die Anwendung von IoT-Geräten und Cloud Computing exemplarisch durch ein ESP8266-Board, das via Wifi verbunden ist, realisiert worden. An das Board sind eine LED und ein Temperatursensor (DHT22) angeschlossen, die

beispielhaft für ein Sensor und Schalt-Aktor eines Produktionsmittels stehen sollen. Durch die Verbindung des Mikrocontrollers an den genutzten tingg.io Broker werden die Sensordaten und Statusdaten der LED transparent visualisiert. Hierbei wird eine Echtzeitübertragung in diesem Fall vereinfacht angenommen, wobei die Echtzeitdaten durch den DHT22 begrenzt werden. Da dieser nur alle zwei Sekunden neue Daten liefert. Die hierdurch geschaffene Transparenz von Produktionsdaten gilt als neues verfolgtes Ziel von Unternehmen im Kontext der I4.0.

Des Weiteren ist es möglich, die LED, die beispielhaft für einen Schalter eines Produktionsmittels stehen soll, über den Broker zu steuern. Daraus lässt sich ein weiteres Motiv auf den der Einsatz von IoT-Anwendungen im Kontext von I4.0 ableiten, hierbei handelt es sich um die Verkürzung von Reaktionswegen im Umfeld der Produktion (vgl. Kap. 1.2). Dazu zählen unteranderem schnellere Reaktionswege im Falle einer Störung bzw. Stillstand eines Produktionsmittels durch transparente Echtzeitdaten. Zusätzlich wird durch den konzeptionellen Bot, der zur Überwachung und Hilfe von Instandsetzungsaufträgen eingesetzt werden soll, ein CPS geschaffen. Das CPS soll im Rahmen des Shopfloor Managements im Falle einer Störung von ausgewählten Produktionsmitteln unterstützen, indem es die Sensordaten überprüft und auf Abweichungen zum Sollwert reagiert. Der Einsatz dieses CPS soll die Reaktionswege im Shopfloor Management verkürzen und für weitere Verläufe standardisieren.

Welche weiteren Funktionen sollten für eine weitere Umsetzung hin zur Realisierung in der Praxis betrachtet werden?

Im Rahmen dieser Masterarbeit wurde zu dem theoretischen Konzept eine abgestimmte Fallstudie entworfen und auf Basis eines Mikrocontrollers mithilfe von Arduino umgesetzt. Dazu wurde ein vorhandenes Cloud-Architektur von tingg.io verwendet, da die Nutzung kostenfrei ist, wie auch eine vorhandene Visualisierung von Daten bereitstellt wird und einen Sicherheitsaspekt der Authentifizierung durch einen registrierten Account und Passwort fordert.

Für weitere Umsetzungen hin zu realen Produktionsszenarien ist hierbei die Anwendung des Embedded Systems auf Basis von Arduino und MQTT als Schnittstelle für ausgewählte Produktionsmittel zu überprüfen. Eine notwendige Umstellung auf ein

anderes System als Datenaustauschstandard, damit das Embedded System mit den Produktionsmitteln gekoppelt werden kann, ist für eine weitere Umsetzung hin zu potenziellen Produktionsszenarien zu überprüfen. In diesem Fall ist es vorstellbar, UPC UA zu verwenden, da es sich hierbei um ein Datenaustauschstandard für die industrielle Kommunikation handelt und über diverse Schnittstellenstandards verfügt. Hierbei kann OPC UA mit einem Publish-Subscriber-Modell wie MQTT zusammengeführt werden, um Daten im Kontext der I4.0 zu Nutzen wie Big Data usw.

Für die weitere Realisierung des CPS bzw. der Aufgaben des Bot ist es erforderlich, den Bot in das System zu integrieren. Eine mögliche Realisierung des Bots und die damit einhergehenden Aufgaben lassen sich über neuartige Entwicklungsplattformen wie Node-Red beispielweise umsetzen und mit einem MQTT-Broker verknüpfen.

5 Zusammenfassung und Ausblick

Einflussfaktoren wie die Digitalisierung und Globalisierung treiben die Industrie immer weiter voran, auch im Kontext der vierten industriellen Revolution habe diese Faktoren einen großen Anteil daran. Im Rahmen von I4.0 ist die Entwicklung von neuen Technologien im Bereich der IKT ein entsprechender Treiber neuer Perspektiven und im Umfeld von IoT für Unternehmen. Daraus lässt sich die Motivation dieser Masterarbeit ableiten, indessen im Kontext von Industrie 4.0 ein Konzept mit dem Titel „cloudbasierten Produktionssteuerung“ entworfen wurde.

Im Rahmen der konzeptionellen Ausarbeitung dieser Masterarbeit wurde ein Konzept entwickelt, bei dem es sich um ein cyberphysisches System handelt. Das CPS wird im Umfeld des Shopfloor Managements angewendet und soll dieses unterstützen. Ziel hierbei ist es, durch den Einsatz von Cloud Computing Daten und Informationen von wesentlichen zu überwachenden Produktionsmitteln diese transparent zur Verfügung zu stellen. Weiterhin sollen Reaktionswege in Bezug auf Störfälle im Prozess durch eine transparente Produktionssteuerung verkürzt und zusätzlich der Instandhaltungsprozess eingeleitet werden.

Zur Realisierung des Konzeptes wurde eine praktische Fallstudie entwickelt, welche das vorgestellte Produktionsszenario aus dem Konzept beispielhaft realisieren soll. In dieser Fallstudie wurde ein Mikrocontroller verwendet, auf Basis von Arduino programmiert und Sensoren angekoppelt. Die Sensordaten werden vom Mikrocontroller durch die Kommunikation über MQTT an einen Broker bzw. Cloudserver (tingg.io) weitergeleitet. An den Broker gekoppelte mobile Endgeräte können diese Daten überwachen.

Für den weiteren Verlauf einer möglichen Umsetzung des Konzeptes auf ein reales Produktionsszenario sind weitere Anforderungen zu beachten, die die Kompatibilität der Kommunikation zwischen dem CPS und den zu überwachenden Produktionsmitteln erlauben. Weiterhin birgt die Nutzung von Internet of Things-Anwendungen und Cloud Computing Risiken, die durch Sicherheitsaspekte minimiert werden sollten.

Abschließend ist zu erwähnen, dass dieses Konzept dieser Masterarbeit die realisierten Ziele von Industrie 4.0 aufgegriffen und in Teilen umgesetzt werden konnten, indem unteranderem Sensordaten transparent verfügbar gemacht wurden.

Literaturverzeichnis

Clausen, P.; Mathiasen, J. B.; Nielsen, J. S. (2020): Smart Manufacturing Through Digital Shop Floor Management Boards. In: Wireless Personal Communications 115, S. 3261–3274.

Daten - Volumen der weltweit generierten Daten 2025 | Statista. https://de.statista.com/statistik/daten/studie/267974/umfrage/prognose-zum-weltweit-generierten-datenvolumen/ (Abruf 17.8.2021).

DHT22 temperature-humidity sensor + extras : ID 385 : $9.95 : Adafruit Industries, Unique & fun DIY electronics and kits. https://www.adafruit.com/product/385 (Abruf 29.8.2021).

Dietrich, M. (2021): Digitales Shopfloor Management in SAP-Systemumgebungen: Roadmap und Lösungsalternativen für die Umsetzung. Wiesbaden: Springer Vieweg.

Egli, P.: MQTT - MQ Telemetry Transport for Message Queueing. https://www.slideshare.net/PeterREgli/mq-telemetry-transport (Abruf 28.8.2021).

Hüning, F. (2019): Embedded Systems für IoT. Berlin [Heidelberg]: Springer Vieweg.

Hüwe, P.; Hüwe, S. (2019): IoT at Home: Smart Gadgets mit Arduino, Raspberry Pi, ESP8266 und Calliope entwickeln. München: Hanser. (= #makers do it).

ICIDCA (Conference); Raj, J. S.; Bashar, A.; Ramson, S. R. J. (2020): Innovative Data Communication Technologies and Application: ICIDCA 2019.

IoT-Prognosen: 75 Milliarden vernetze Geräte | SAP News Center. https://news.sap.com/germany/2019/10/iot-chance-moeglichkeiten/ (Abruf 17.8.2021).

Kletti, J.; Schumacher, J. (2014): Die Anforderungen an die perfekte Produktion. In: Die perfekte Produktion. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 1–7.

Lauzi, M. (2019): Smart City: technische Fundamente und erfolgreiche Anwendungen. München: fv, Fachbuchverlag Leipzig im Carl Hanser Verlag.

Lechler, A.; Schlechtendahl, J. (2017): Steuerung aus der Cloud. In: Vogel-Heuser, B.; Bauernhansl, T.; ten Hompel, M. (Hrsg.): Handbuch Industrie 4.0 Bd.1. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 61–74.

Leyendecker, B.; Pötters, P. (2021): Shopfloor Management: führen am Ort des Geschehens. 2. Auflage. München: Hanser. (= Pocket Power 075).

Mourtzis, D.; Vlachou, E. (2018): A cloud-based cyber-physical system for adaptive shop-floor scheduling and condition-based maintenance. In: Journal of Manufacturing Systems 47, S. 179–198.

MQTT Websocket Client. http://www.hivemq.com/demos/websocket-client/ (Abruf 6.9.2021).

Reinheimer, S. (Hrsg.) (2017): Industrie 4.0: Herausforderungen, Konzepte und Praxisbeispiele. Wiesbaden: Springer Fachmedien Wiesbaden. (= Edition HMD).

Senft, F. (2019): Kommunikationstechnische Optimierung eines energieautarken funkbasierten Sensorkonzepts.

Steven, M. (2019): Industrie 4.0: Grundlagen - Teilbereiche - Perspektiven. 1. Auflage. Stuttgart: Verlag W. Kohlhammer. (= Moderne Produktion).

Trojan, W. (2017): Das MQTT-Praxisbuch. 1. Auflage. Aachen: Elektor-Verlag GmbH. (= An Elector Publication).

Vogel-Heuser, B.; Bauernhansl, T.; Ten Hompel, M. (Hrsg.) (2017): Handbuch Industrie 4.0. Bd. 4: Allgemeine Grundlagen / Birgit Vogel-Heuser, Thomas Bauernhansl, Michael ten Hompel. 2. Auflage. Berlin; [Heidelberg]: Springer Vieweg. (= Springer Reference Technik).

Xiao, P. (2018): Designing embedded systems and the internet of things (IoT) with the ARM Mbed. First edition. Hoboken, NJ: John Wiley & Sons, Inc.

Anhang

Anhang A:

Arduino IDE-Programm zur Steuerung einer Buildin LED eines ESP8266 über MQTT

mqtt_esp8266_V1 §

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

const char* ssid = "...";
const char* password = "...";
const char* mqtt_server = "broker.mqtt-dashboard.com";

WiFiClient espClient;
PubSubClient client(espClient);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE (50)
char msg[MSG_BUFFER_SIZE];
int value = 0;

void setup_wifi() {

  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  randomSeed(micros());

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void callback(char* topic, byte* payload, unsigned int length) {
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();
```

mqtt_esp8266_V1 §

```
  }
  Serial.println();

  // Switch on the LED if an 1 was received as first character
  if ((char)payload[0] == '1') {
    digitalWrite(BUILTIN_LED, LOW);   // Turn the LED on (Note that LOW is the voltage level
    // but actually the LED is on; this is because
    // it is active low on the ESP-01)
  } else {
    digitalWrite(BUILTIN_LED, HIGH);  // Turn the LED off by making the voltage HIGH
  }

}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP8266PubSub";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      client.publish("outTopic", "ESP8266PubSub connected");
      // ... and resubscribe
      client.subscribe("test33");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

void setup() {
  pinMode(BUILTIN_LED, OUTPUT);     // Initialize the BUILTIN_LED pin as an output
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

void loop() {
```

mqtt_esp8266_V1 §

```
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "ESP8266PubSub";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      client.publish("outTopic", "ESP8266PubSub connected");
      // ... and resubscribe
      client.subscribe("test33");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

void setup() {
  pinMode(BUILTIN_LED, OUTPUT);     // Initialize the BUILTIN_LED pin as an output
  Serial.begin(115200);
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

void loop() {

  if (!client.connected()) {
    reconnect();
  }
  client.loop();

  unsigned long now = millis();
  if (now - lastMsg > 7000) {
    lastMsg = now;
    ++value;
    snprintf (msg, MSG_BUFFER_SIZE, "%ld", value %2);
    Serial.print("Publish message: Hier ist ESP8266PubSub ist hier ;)");
    Serial.println(msg);
    client.publish("test33", msg);
  }
}
```

Anhang B:

Arduino IDE Beispielprogramm zur DHT-Sensorsteuerung

DHT_Unified_Sensor

```
// DHT Temperature & Humidity Sensor
// Unified Sensor Library Example
// Written by Tony DiCola for Adafruit Industries
// Released under an MIT license.

// REQUIRES the following Arduino libraries:
// - DHT Sensor Library: https://github.com/adafruit/DHT-sensor-library
// - Adafruit Unified Sensor Lib: https://github.com/adafruit/Adafruit_Sensor

#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

#define DHTPIN 2     // Digital pin connected to the DHT sensor
// Feather HUZZAH ESP8266 note: use pins 3, 4, 5, 12, 13 or 14 --
// Pin 15 can work but DHT must be disconnected during program upload.

// Uncomment the type of sensor in use:
//#define DHTTYPE    DHT11     // DHT 11
#define DHTTYPE    DHT22     // DHT 22 (AM2302)
//#define DHTTYPE    DHT21     // DHT 21 (AM2301)

// See guide for details on sensor wiring and usage:
//   https://learn.adafruit.com/dht/overview

DHT_Unified dht(DHTPIN, DHTTYPE);

uint32_t delayMS;

void setup() {
  Serial.begin(9600);
  // Initialize device.
  dht.begin();
  Serial.println(F("DHTxx Unified Sensor Example"));
  // Print temperature sensor details.
  sensor_t sensor;
  dht.temperature().getSensor(&sensor);
  Serial.println(F("------------------------------------"));
  Serial.println(F("Temperature Sensor"));
  Serial.print  (F("Sensor Type: ")); Serial.println(sensor.name);
  Serial.print  (F("Driver Ver:  ")); Serial.println(sensor.version);
  Serial.print  (F("Unique ID:   ")); Serial.println(sensor.sensor_id);
  Serial.print  (F("Max Value:   ")); Serial.print(sensor.max_value); Serial.println(F("°C"));
  Serial.print  (F("Min Value:   ")); Serial.print(sensor.min_value); Serial.println(F("°C"));
  Serial.print  (F("Resolution:  ")); Serial.print(sensor.resolution); Serial.println(F("°C"));
  Serial.println(F("------------------------------------"));
  // Print humidity sensor details.
  dht.humidity().getSensor(&sensor);
```

DHT_Unified_Sensor

```
  Serial.println(F("------------------------------------"));
  Serial.println(F("Temperature Sensor"));
  Serial.print  (F("Sensor Type: ")); Serial.println(sensor.name);
  Serial.print  (F("Driver Ver:  ")); Serial.println(sensor.version);
  Serial.print  (F("Unique ID:   ")); Serial.println(sensor.sensor_id);
  Serial.print  (F("Max Value:   ")); Serial.print(sensor.max_value); Serial.println(F("°C"));
  Serial.print  (F("Min Value:   ")); Serial.print(sensor.min_value); Serial.println(F("°C"));
  Serial.print  (F("Resolution:  ")); Serial.print(sensor.resolution); Serial.println(F("°C"));
  Serial.println(F("------------------------------------"));
  // Print humidity sensor details.
  dht.humidity().getSensor(&sensor);
  Serial.println(F("Humidity Sensor"));
  Serial.print  (F("Sensor Type: ")); Serial.println(sensor.name);
  Serial.print  (F("Driver Ver:  ")); Serial.println(sensor.version);
  Serial.print  (F("Unique ID:   ")); Serial.println(sensor.sensor_id);
  Serial.print  (F("Max Value:   ")); Serial.print(sensor.max_value); Serial.println(F("%"));
  Serial.print  (F("Min Value:   ")); Serial.print(sensor.min_value); Serial.println(F("%"));
  Serial.print  (F("Resolution:  ")); Serial.print(sensor.resolution); Serial.println(F("%"));
  Serial.println(F("------------------------------------"));
  // Set delay between sensor readings based on sensor details.
  delayMS = sensor.min_delay / 1000;
}

void loop() {
  // Delay between measurements.
  delay(delayMS);
  // Get temperature event and print its value.
  sensors_event_t event;
  dht.temperature().getEvent(&event);
  if (isnan(event.temperature)) {
    Serial.println(F("Error reading temperature!"));
  }
  else {
    Serial.print(F("Temperature: "));
    Serial.print(event.temperature);
    Serial.println(F("°C"));
  }
  // Get humidity event and print its value.
  dht.humidity().getEvent(&event);
  if (isnan(event.relative_humidity)) {
    Serial.println(F("Error reading humidity!"));
  }
  else {
    Serial.print(F("Humidity: "));
    Serial.print(event.relative_humidity);
    Serial.println(F("%"));
  }
}
```

Eidesstattliche Erklärung

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Erklärung zur selbstständigen Bearbeitung einer Abschlussarbeit

Gemäß der Allgemeinen Prüfungs- und Studienordnung ist zusammen mit der Abschlussarbeit eine schriftliche Erklärung abzugeben, in der der Studierende bestätigt, dass die Abschlussarbeit „– bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit [(§ 18 Abs. 1 APSO-TI-BM bzw. § 21 Abs. 1 APSO-INGI)] – ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt wurden. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich zu machen."

Quelle: § 16 Abs. 5 APSO-TI-BM bzw. § 15 Abs. 6 APSO-INGI

Dieses Blatt, mit der folgenden Erklärung, ist nach Fertigstellung der Abschlussarbeit durch den Studierenden auszufüllen und jeweils mit Originalunterschrift als letztes Blatt in das Prüfungsexemplar der Abschlussarbeit einzubinden.
Eine unrichtig abgegebene Erklärung kann -auch nachträglich- zur Ungültigkeit des Studienabschlusses führen.

Erklärung zur selbstständigen Bearbeitung der Arbeit

Hiermit versichere ich,

Name: Vieth

Vorname: Till

dass ich die vorliegende Masterarbeit bzw. bei einer Gruppenarbeit die entsprechend gekennzeichneten Teile der Arbeit – mit dem Thema:

Cloudbasierte Produktionssteuerung

ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

- die folgende Aussage ist bei Gruppenarbeiten auszufüllen und entfällt bei Einzelarbeiten -

Die Kennzeichnung der von mir erstellten und verantworteten Teile der -bitte auswählen- ist erfolgt durch:

Hamburg	10.09.2021	[redacted]
Ort	Datum	Unterschrift im Original