Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Aike Banse

Entwicklung eines automatisierten, datenumfangsoptimierten, Cloud-basierten Systems für die vorbeugende Instandhaltung einer Fassreinigungsanlage

Fakultät Technik und Informatik
Department Fahrzeugtechnik und Flugzeugbau

Faculty of Engineering and Computer Science
Department of Automotive and Aeronautical Engineering

Aike Banse

Entwicklung eines automatisierten, datenumfangsoptimierten, Cloud-basierten Systems für die vorbeugende Instandhaltung einer Fassreinigungsanlage

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Mechatronik
am Department Fahrzeugtechnik und Flugzeugbau
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

in Zusammenarbeit mit:
TIG Automation GmbH
Automatisierung
Bötelkamp 38
22529 Hamburg

Erstprüfer: Prof. Dr. Rasmus Rettig
Zweitprüfer: Prof. Dr. Henner Gärtner

Abgabedatum: 02.12.2021

Zusammenfassung

Name des Studierenden
Aike Banse

Thema der Bachelorthesis
Entwicklung eines automatisierten, datenumfangsoptimierten, Cloud-basierten Systems für die vorbeugende Instandhaltung einer Fassreinigungsanlage

Stichworte
SPS, Software-Engineering, Cloud, Automatisierung, IoT, HMI, Vorbeugende Instandhaltung

Kurzzusammenfassung
Diese Arbeit umfasst den Versuch der Konzeption und Entwicklung eines automatisierten, datenumfangsoptimierten, Cloud-basierten Systems zur Übertragung und Verarbeitung von Prozessdaten einer Fassreinigungsanlage. Für das Projekt wird zunächst die Fassreinigungsanlage analysiert und die für das System notwendigen Anforderungen zusammengefasst. In den darauffolgenden Phasen wird ein geeignetes Konzept entwickelt und methodisch ausgewählt. Hierzu gehören sowohl die Auswahl der Komponenten als auch der Aufbau der Cloud-Datenhaltung, das Programmieren der Übertragungen und der SPS-Bausteine sowie die Verlaufsanalysen der Prozessdaten zur vorbeugenden Instandhaltung. Abschließend wird das Gesamtsystem mit einer Funktionsanalyse auf seine Funktionen überprüft. Die Analyse der Prozessdaten zur vorbeugenden Instandhaltung der Fassreinigungsanlage, wird durch den gescheiterten Verbindungsaufbau der Analyseplattform mit der Datenbank nicht durchgeführt.

Name of Student
Aike Banse

Title of the paper
Development of an automated, data volume-optimized, cloud-based system for the preventive maintenance of a barrel cleaning system

Keywords
PLC, software engineering, cloud, Automation, IoT, HMI, preventive maintenance

Abstract
This thesis includes the attempt of the conception and development of an automated, data volume-optimized, cloud-based system for the transmission and processing of process data of a barrel cleaning system. First the barrel cleaning system will be analyzed and then the necessary requirements for the system will be summarized. In the following phases, a suitable concept is developed and methodically selected. This includes the selection of the components as well as the structure of the cloud data management, the programming of the transmissions and the PLC modules as well as the process data analysis for preventive maintenance. Finally, the overall system is checked for its functions with a functional analysis. The analysis of the process data for the preventive maintenance of the barrel cleaning system is not carried out due to the failed connection establishment between the analysis platform and the database.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

SPS *Speicherprogrammierbare Steuerung*
HMI *Human Machine Interface*
SCADA *Supervisory Control and Data Acquisition*
IIoT *Industrial Internet of Things*
M2M *Maschine-zu-Maschine-Kommunikation*
MQTT *Message Queuing Telemetry Transport*
SQL *Structured Query Language*
UML *Unified Modelling Language*
VPS *Verbindungsprogrammierbare Steuerung*
EVA *Eingabe Verarbeitung Ausgabe*
CPU *Central Processing Unit*
WAN *Wide Area Network*
LAN *Local Area Network*
QOS *Quality of Service*
PAE *Prozessabbild der Eingänge*
PAA *Prozessabbild der Ausgänge*
ARN *Amazon Ressource Number*
VPC *Virtual Private Cloud*
FUP *Funktionsplan*
ST *strukturierter Text*
BASIC *Beginner's All-purpose Symbolic Instruction Code*
HTTPS *HyperText Transfer Protocol Secure*
LoRaWAN *Long Range Wide Area Network*

1. Einleitung

Im Rahmen der vorliegenden Bachelorarbeit soll die vorbeugende Instandhaltung einer Fassreinigungsanlage mit der Cloud-Analyseplattform Qlik Sense realisiert werden [1]. Hierzu sollen Daten automatisiert und durch eine Vorauswahl umfangsoptimiert von der speicherprogrammierbaren Steuerung (SPS) in eine Cloud übermittelt und für Qlik Sense bereitgestellt werden. Folgend werden die Konzeption und Entwicklung dieses Projektes erläutert.

1.1 Motivation

Aufgrund von Bauteilverschleiß und daraus resultierenden unvorhergesehenen Anlagenausfällen entstehen für produzierende Firmen hohe wirtschaftliche Verluste. Um diese Verluste vermeiden zu können, wird vorbeugende Instandhaltung betrieben, indem Prozessdatenverläufe erstellt und analysiert werden. Mit diesen Verlaufsanalysen ist es möglich einen bevorstehenden Ausfall von Bauteilen frühzeitig zu erkennen und eine notwendige Wartung bei Prozessstillstand ohne Produktionsausfall durchzuführen [2]. Damit die Arbeitslast für das Personal so gering wie möglich bleibt, soll der Übertragungs- und Analyseprozess der Daten automatisiert erfolgen. Ebenfalls soll die Datenlast so gering wie möglich sein, um die Zykluszeiten der SPS so gering wie möglich zu halten und damit die Anlagenprozesse nicht zu verlangsamen. Durch die Cloud-basierte Lösung ist eine Einsehbarkeit der Prozessdaten in der Verwaltungsabteilung der Firma möglich, um so Prozesse und deren Kosten besser kalkulieren zu können. Anlagen- und Verwaltungsnetzwerke werden oft getrennt, da die Anlagen keinen Zugang zum Internet besitzen sollen, um Eingriffe von außen zu vermeiden. Die Maschinenkommunikation kann mit einem Local Area Network (LAN), einem lokalen Netzwerk, umgesetzt werden. Verwaltungsnetzwerke hingegen benötigen Zugang zum Internet, auch Wide Area Network (WAN) genannt, um in Kontakt mit Zulieferern oder Dienstleistern zu treten. Mit einer gesicherten Übertragung an eine Cloud-Datenhaltung ist es möglich die Prozessdaten der Verwaltung zur Einsicht bereit zu stellen, wodurch eine zeiteffizientere Arbeit durch bessere Planbarkeit der Produktion möglich ist.

1.2 Ziel

Das Ziel dieser Arbeit ist es, ein Cloud-basiertes System zur sicheren Datenübertragung von Prozessdaten, zu entwickeln und mit den Prozessdaten der Fassreinigungsanlage eine Verlaufsanalyse durchzuführen, um eine vorbeugende Instandhaltung zu ermöglichen. Die Übertragung der Prozessdaten von der SPS oder dem Human Machine Interface (HMI) soll automatisch und umfangsoptimiert an eine Cloud-Datenhaltung übergeben werden, um dort die Daten der Analyseplattform Qlik Sense für Verlaufsanalysen zur vorbeugenden Instandhaltung bereit zu stellen.

1.3 Vorgehen

Um das Projekt erfolgreich durchführen zu können, werden nacheinander verschiedene Arbeitsphasen nach dem Top-Down-Prinzip durchlaufen [3]. Zu Beginn wird der aktuelle Stand der Technik recherchiert und die Ausgangssituation geschildert. Nachfolgend wird der Markt für das Produkt untersucht und die Anforderungen an das Projekt in einer Anforderungsliste festgehalten. Anschließend werden in der Konzeptionsphase Lösungskonzepte anhand der Anforderungen entwickelt und miteinander verglichen. Ebenfalls werden Analysen zur vorbeugenden Instandhaltung konzipiert und die zur Durchführung notwendigen Prozessdaten ermittelt. Im nächsten Schritt, der Entwicklungsphase, werden die ausgewählten Konzepte der Software, der Cloud-Struktur und der Übertragung der Prozessdaten geplant. Im Anschluss an die Entwicklung werden die ausgewählten Lösungen realisiert, implementiert und getestet. Nachdem die korrekte Übertragung der Anlagendaten sichergestellt wurde, werden diese an Qlik Sense

übergeben. Mit Qlik Sense und den Prozessdaten der Fassreinigungsanlage werden, die in der Konzeptionsphase entwickelten, Analysen zur vorbeugenden Instandhaltung durchgeführt. Abschließend wird ein Fazit zum Verlauf des Projektes und der Einhaltung der Anforderungen gezogen.

2. Stand der Technik, Grundlagen und Ausgangssituation

In diesem Abschnitt der Arbeit wird zunächst der aktuelle Stand der Technik und im Anschluss die verwendeten Grundlagen erläutert. Anschließend wird die bestehende Ausgangssituation dargestellt und analysiert.

2.1 Stand der Technik

Um den Stand der Technik aufzuzeigen, werden für diese Arbeit relevante Themengebiete in Hinblick auf den aktuellen Forschungsstand erläutert. Nachfolgend werden die Themengebiete benannt und die Anwendung erörtert. Ebenfalls wird eine verkleinerte Marktanalyse durchgeführt, um das Potenzial und mögliche Wettbewerbslösungen abschätzen zu können.

2.1.1 Vorbeugende Instandhaltung

Die Instandhaltung wird in der DIN 31051 definiert und ist die Kombination der technischen und verwaltungsmäßigen Maßnahmen während der Lebensdauer der Anlagenkomponenten zum Erhalt oder zur Wiederherstellung der Funktionalität [4].

Die Instandhaltung lässt sich in mehrere Instandhaltungsstrategien einordnen (siehe Abbildung 1). Für diese Instandhaltungsstrategien gibt es zwei übergeordnete Gruppen, die reaktive und die präventive Instandhaltung. Die präventive Instandhaltung, welche auch als vorbeugende Instandhaltung bezeichnet wird, kann in weitere Instandhaltungsstrategien aufgeteilt werden. Diese weitere Unterscheidung der präventiven Instandhaltung in eine periodisch vorbeugende, eine zustandsabhängige oder eine vorrausschauende Instandhaltungsstrategie ist ausschlaggebend für die Zuverlässigkeit der Anlage und die anfallenden Instandhaltungskosten. Die Grundlagen der Instandhaltung sind in der DIN 31051 festgehalten. Der Nachweis der Kostenreduzierung durch Änderung der Instandhaltungsstrategie ist nur längerfristig feststellbar, jedoch können die Ausfallzeiten der technischen Anlagen und die Ausnutzung des Abnutzungsvorrates der Anlagenkomponenten in Bezug auf die gewählte Strategie nachgewiesen werden. Der Abnutzungsvorrat beschreibt hierbei die vom Hersteller für die Weiterverwendung zulässige Abnutzung des Bauteils, welcher zur Kostenminimierung ausgereizt werden sollte. [4]

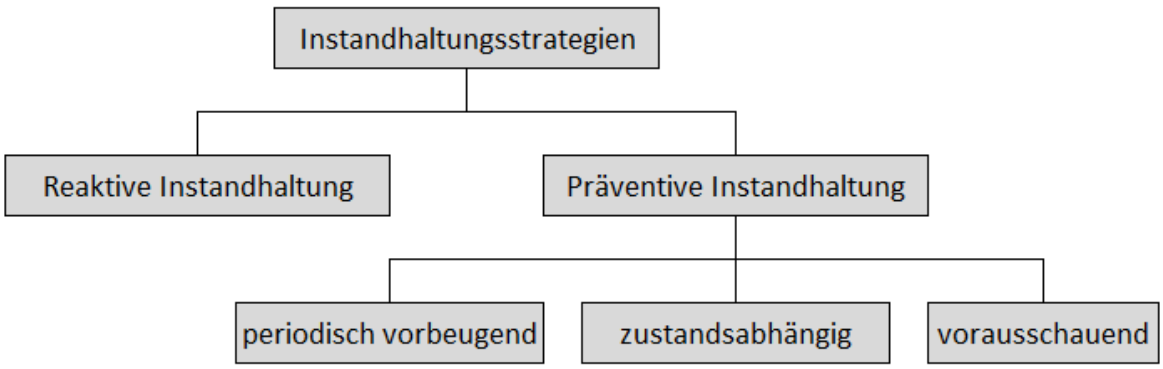


Abbildung 1: Instandhaltungsstrategien

Bei der periodisch vorbeugenden Instandhaltung werden Anlagenkomponenten unabhängig vom Abnutzungsvorrat und vorherrschendem Verschleiß in einem festgelegten Intervall ausgetauscht (siehe Abbildung 2). Die Intervalle können zeit- oder ereignisbezogen geplant werden. Da in dieser Instandhaltungsstrategie der Abnutzungsvorrat der Komponenten nicht ausgenutzt wird, sind diese Anlagen sehr ausfallsicher. Diese Ausfallsicherheit eignet sich besonders für Anlagen mit hohen Sicherheits- oder

Umweltanforderungen. Die periodisch vorbeugende Instandhaltung lässt sich anhand der Intervalle gut planen und benötigte Instandhaltungsressourcen für die Maßnahmen bereitstellen. Ebenfalls kann die Durchführung der Instandhaltung in einem Zeitraum geplant werden, in dem die Anlage nicht betrieben wird. Bei der Planung der Instandhaltung während des Anlagenstillstandes werden keine extra Ausfallzeiten und dadurch resultierende zusätzliche Kosten generiert. Höhere Kosten entstehen bei der periodisch vorbeugenden Strategie durch die häufig durchzuführenden Instandhaltungsmaßnahmen und dem hohen Verbrauch von Ersatzteilen. Schwierig bei der periodisch vorbeugenden Instandhaltungsstrategie ist die Festlegung der Intervalle, da die Intervalle für jede einzelne Anlagenkomponente durch einen unterschiedlichen Abnutzungsvorrat und eine unterschiedliche Lebensdauer individuell festgelegt werden müssen. Für die Planung der individuellen Intervalle sind gute Schadensdokumentation vergangener Ausfälle nötig, welche mit statistischen Verfahren analysiert werden müssen. [2]

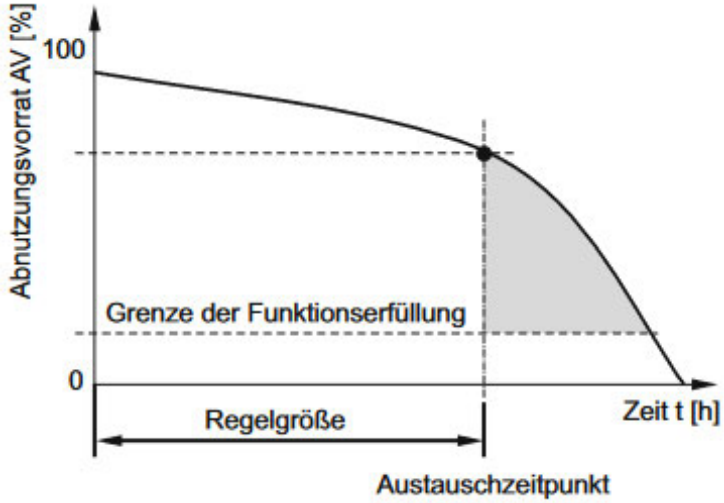


Abbildung 2: periodisch vorbeugende Instandhaltung

Die zustandsabhängige Instandhaltung hat die geringste Stillstandzeit der Anlage bei guter Ausnutzung des Abnutzungsvorrates. Dies kommt zustande durch die Verwendung des individuellen Abnutzungsvorrates des Bauteils, der als Regelgröße für die Durchführung der Instandhaltungsmaßnahmen verwendet wird (siehe Abbildung 3). So ist es möglich die Komponenten optimal zu verwenden und diese ebenfalls zum idealen Zeitpunkt auszutauschen. Um die zustandsabhängige Instandhaltungsstrategie anwenden zu können, braucht es die Möglichkeit den aktuellen Zustand der Komponenten abfragen zu können und Änderungen dieser zu erkennen. Die Intervalle, in der die Zustandsabfrage der Komponenten stattfinden müssen, werden auf Grundlage des Änderungsintervalls des Abnutzungsvorrats erstellt. Die Überwachung des Zustandes kann sowohl vom Menschen, durch regelmäßige Inspektion, als auch maschinell durch zyklische Überwachung von Komponentensignalen realisiert werden. Die zustandsabhängige Instandhaltung kann nur angewendet werden, wenn die Veränderung des Abnutzungsvorrates

messbar ist und Schwellwerte für den Abnutzungsvorrat durch Herstellerinformation gegeben sind oder durch vergangene Ausfälle erzeugt werden können. [2]

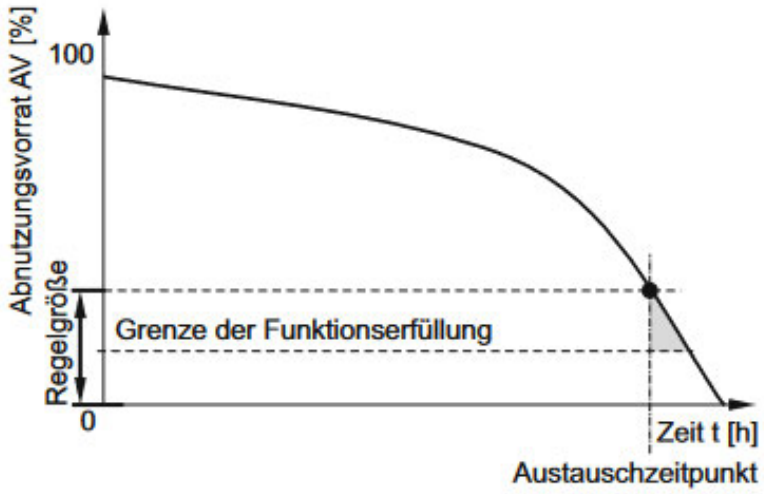


Abbildung 3: zustandsabhängige Instandhaltung

Die vorrausschauende Instandhaltungsstrategie ist eine Weiterentwicklung der zustandsabhängigen Instandhaltungsstrategie. Im Gegensatz zur zustandsabhängigen Instandhaltung wird hierbei nicht der Zustand betrachtet, sondern die bei den Zustandsänderungen potenziell möglichen Störungen, um spätere Weiterentwicklungen dieser Störungen zu verhindern. Bei der vorrausschauenden Instandhaltungsstrategie ist es essenziell die Funktionen und möglichen Funktionsstörungen der Anlagen zu definieren. Die Funktionen werden in primäre, sekundäre und überflüssige Funktionen eingeteilt. Die primären Funktionen werden aus den hauptsächlichen Investitionsgründen abgeleitet und stehen somit für den Anwendungszweck der Anlage. Die sekundären Funktionen spiegeln Nebenfunktionen der Anlagen wider. Sekundäre Funktionen sind zum Beispiel die Betriebssicherheit, der Umweltschutz, die Datenspeicherung, das Aussehen der Anlagen oder die Wirtschaftlichkeit. Überflüssige Funktionen sind Funktionen, die für die Funktionserfüllung der Anlage unwichtig sind. Jedoch haben überflüssige Funktionen Auswirkungen auf die Zuverlässigkeit der Gesamtanlage und damit indirekt Einfluss auf primäre oder sekundäre Funktionen. Ein Beispiel für eine überflüssige Funktion ist die Visualisierung von Anlagenprozessen, da diese keinen direkten Einfluss auf die Anlage hat oder relevant für die Sicherheit ist. Anhand der Visualisierung können jedoch frühzeitig Fehler erkannt und somit die Primärfunktion erhalten werden. Funktionsstörungen werden in sicherheitsrelevante, umweltrelevante, betriebsrelevante und betriebsunabhängige Störungen eingeteilt. Nach der Definition der Funktionen und Störungen wird überprüft, ob es Maßnahmen zur Vermeidung dieser Störungen gibt. [2]

Die reaktive Instandhaltung ist ausfallorientiert und wird erst nach dem Versagen eines Bauteils durchgeführt. Diese Instandhaltungsstrategie wird daher auch als Feuerwehrstrategie oder störungsbedingte Instandsetzung bezeichnet. Bei dieser Strategie wird auf jegliche Wartungen und Inspektionen der Anlage verzichtet. Ebenfalls erfordert die Strategie bei Ausfall von Anlagenteilen eine spontane und schnelle Reaktion vom Instandhalter, was fachspezifische Fähigkeiten in Hinsicht auf Wahrnehmung und Beurteilung der Anlage voraussetzt. Die, zur Vorbeugung des Ausfalls geleistete, Planung der reaktiven Instandhaltung gibt es nicht, wodurch Störungen meist plötzlich und spontan auftreten. Diese Störungen sorgen für einen teilweisen oder kompletten Ausfall von Anlagenteilen, was zu hohen wirtschaftlichen Kosten führen kann. Ebenso hat die reaktive Instandhaltung durch den unvorhergesehenen Ausfall von Komponenten ein hohes Gefährdungspotential für das Personal oder andere Anlagenkomponenten, da unvorhergesehene Reaktionen oder Überbelastungen in der Anlage ausgelöst werden können. Die Anwendung der reaktiven Instandhaltungsstrategie sollte nur bei sekundären und untergeordneten

Komponenten und Anlagen des Betriebs Verwendung finden, welche nicht sicherheitsrelevant sind oder zum kompletten Stillstand der Produktion führen können. [2]

Aus den vier oben genannten Grundstrategien lassen sich sehr viele weitere Strategien ableiten, die meist eine Kombination der Grundstrategien sind. So kann man bei der vorbeugenden Instandhaltung einer Anlage für kostengünstige und einfach auszutauschende Bauteile eine periodische Instandhaltung implementieren, um die Datenlast gering zu halten. [2]

Im Verlauf der Konzeption und der Planung der Analysen zur vorbeugenden Instandhaltung, wird eine Instandhaltungsstrategie entwickelt und erläutert.

2.1.2 Industrial Internet of Things (IIoT)

Das „Industrial Internet of Things“ (IIoT) oder auf Deutsch das „Industrielle Internet der Dinge“ beschreibt die Bereitstellung von Prozessdaten über das Internet, um Prozesse überwachen, steuern, analysieren oder verknüpfen zu können. Das IIoT erfordert eine horizontale und vertikale Integration der Komponenten, sowie die Möglichkeit Daten ins Internet zu übermitteln. Die horizontale Integration beschreibt die Vernetzung von Teilnehmern auf derselben Leitebene und die damit verbundene Kommunikation der Teilnehmer. Beispiele für die horizontale Integration sind die Maschine-zu-Maschine-Kommunikation (M2M) auf der Prozessleitebene oder die Vernetzung von Kunden und Lieferanten in der Unternehmensleitebene. Bei der vertikalen Integration hingegen werden die verschiedenen Leitebenen miteinander verknüpft, um einen Datenaustausch zu ermöglichen. Die Verknüpfung der Leitebenen in der vertikalen Integration, wird meist durch Gateways hergestellt. Gateways sind Teilnehmer des Netzwerkes, die die verschiedenen Leitebenen miteinander verbinden. Gateways haben eine eingebaute Sicherheit und dienen als Übersetzer zwischen zwei Ebenen und deren Übertragungsstandards. Die vertikale und horizontale Integration sowie die verschiedenen im Unternehmen vorkommenden Leitebenen, werden in der Automatisierungspyramide beschrieben (siehe Abbildung 4). [5]

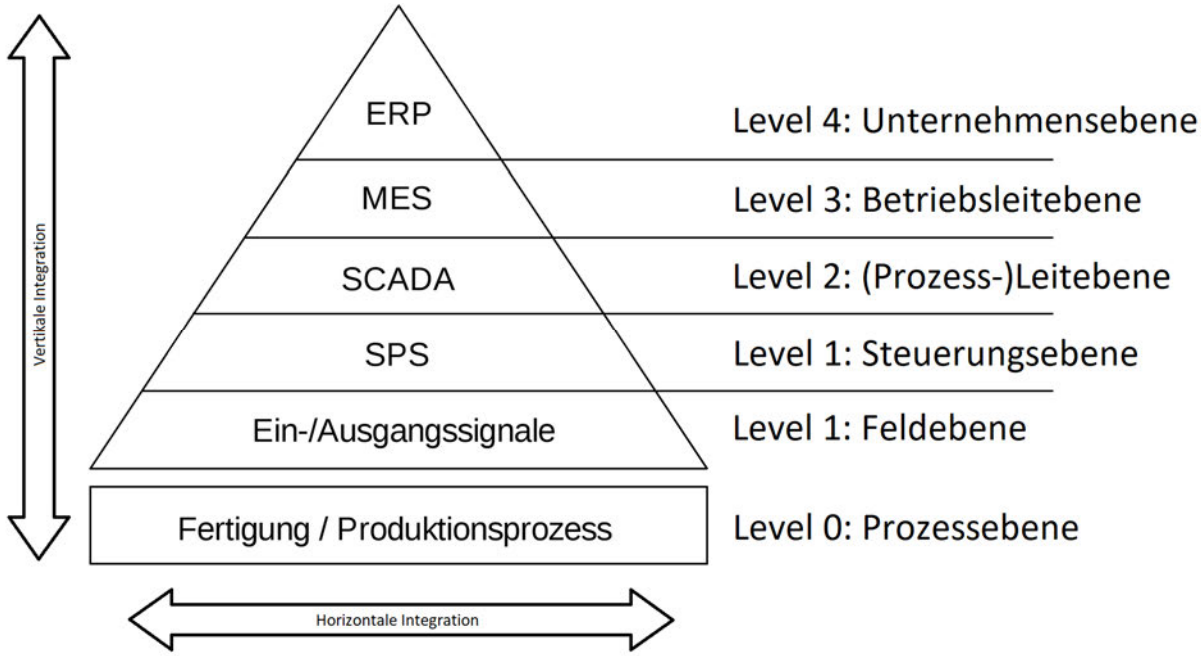


Abbildung 4: Automatisierungspyramide

Beim IIoT erhalten die verschiedenen Leitebenen Zugang zum Internet, um die zu übermittelnden Daten aus den Leitebenen mit speziellen Protokollen an einen Broker weiterzugeben. Der Broker dient als

Schnittstelle und verteilt die ankommenden Daten an die richtigen Teilnehmer. Diese Teilnehmer können beispielsweise Datenhaltungen wie SQL-Server oder Verarbeitungs- und Analysesoftware wie Qlik Sense sein [5].

2.1.3 Marktanalyse

In diesem Abschnitt wird der Markt des Produktes analysiert, um eine Aussage zum Umsatzpotenzial des Produktes im Zielmarkt treffen zu können. Es wird der Zielmarkt und dessen Größe ermittelt, sowie die Marktentwicklung betrachtet. Ebenfalls werden deutsche Firmen mit Lösungen zur vorbeugenden Instandhaltung in Hinblick auf ihren Marktanteil analysiert. Des Weiteren werden die Kundenbranchen für das Produkt erfasst, um abschließend das Marktpotenzial des Zielmarktes ermitteln zu können. Die Marktanalyse wird für das Produkt zur vorbeugenden Instandhaltung ohne das Anwendungsbeispiel der Fassreinigungsanlage betrachtet. Ebenfalls wird die Marktanalyse aus Sicht der Firma TIG Automation GmbH erstellt, da diese das Produkt vermarkten wird. [11]

2.1.3.1 Beschreibung des Zielmarktes

Der Zielmarkt, in dem das Produkt vertrieben werden soll, sind Unternehmen mit bereits automatisierten Anlagen. Ebenfalls müssen die Komponenten die Möglichkeit bieten einen Zustand auszugeben. Die Größen der zu erweiternden Anlagen ist nicht von Bedeutung, da sowohl die Cloud-Struktur, sowie die Analyseplattform Qlik Sense in der Größe skalierbar sind. Durch das weltweite Agieren der TIG Automation GmbH kann das Produkt international vertrieben und angeboten werden, womit der Standort der Anlage ebenfalls irrelevant ist.

2.1.3.2 Analyse der Marktgröße

Um die Marktgröße für ein Produkt zur vorbeugenden Instandhaltung einschätzen zu können, wird die Statistik zum Umsatz der deutschen Unternehmen für industrielle Instandhaltung betrachtet. Der Gesamtumsatz, der durch die 15 wichtigsten deutschen Unternehmen in dieser Branche 2019 erwirtschaftet wurde, beträgt ca. 11,122 Milliarden Euro. [12] In einer statistischen Umfrage zur vorbeugenden Instandhaltung wurden 323 deutsche Unternehmen zum Zweck der innerbetrieblichen Nutzung von vorbeugender Instandhaltung befragt. 28% der Unternehmen verwenden die vorbeugende Instandhaltung zum rechtzeitigen und automatisierten Erkennen vom Verschleiß. Somit ist der Markt zu ca. 28% für dieses Produkt abgedeckt, was im Umkehrschluss ca. 72% der Unternehmen als potenzielle Kunden aufzeigt. [13]

2.1.3.3 Analyse der Marktentwicklung und Trends

Der Markt der Industrie 4.0, zu dem sowohl die vorbeugende Instandhaltung wie auch das IIoT gehört, wächst stetig. Dieses Wachstum des Marktes ist auf unterschiedliche Trends in der Industrie zurückzuführen. Einer dieser Trends ist die bereits erwähnte Industrie 4.0 und die damit verbundene Anlagenautomatisierung und Anlagenverknüpfung zur Qualitätsverbesserung des Endproduktes. Ebenfalls wird zur Zeit der Coronapandemie der Trend der Digitalisierung immer relevanter. Die Digitalisierung von Geschäftsabläufen oder Anlagenprozessen sorgt für eine hohe Flexibilität, bessere Skalierbarkeit und höhere Kosteneffizienz. Das Wachstum des Digitalisierungstrends hat zur Folge, dass die IT-Sicherheit als Trend ebenfalls immer bedeutsamer wird, um die Geschäftsdaten vor Fremdzugriff zu schützen. Ein weiterer Trend ist die Nachhaltigkeit und die damit verbundene Material- und Energieersparnis. Die Nachhaltigkeit und der flexible, schneller werdende Markt bedingen den Trend zur vorbeugenden Instandhaltung, damit ungeplanten Stillstandzeiten von Produktionsanlagen minimiert werden können. [14]

2.1.3.4 Analyse der Kundenbranchen

Die Kunden, denen das Produkt angeboten werden kann, müssen prozessautomatisierte Anlagen besitzen, somit ist jede Firma mit einer automatisierten Anlage ein potenzieller Kunde. Als Vereinfachung und Spezifizierung werden die Branchen der TIG Automation GmbH betrachtet. Die Kundenbranchen der TIG Automation GmbH liegen im Wesentlichen im Maschinenbau, in der Fördertechnik, in der Feinchemie-, Chemie- und Pharmaindustrie, in der Klima-, Lüftungs- und Absaugtechnik und in der Abfallwirtschaft.

2.1.3.5 Analyse des Marktpotenzials des Zielmarktes

Um das Marktpotenzial des Zielmarktes ermitteln zu können, wird die Formel für den Marksättigungsgrad umgestellt. Wie im Abschnitt zur Analyse der Marktgröße beschrieben, liegt der Marktsättigungsgrad bei 28% und das Marktvolumen bei 11,122 Milliarden Euro. Die umgestellte Formel für das Marktpotenzial lautet:

$$\text{Marktpotenzial} = \frac{\text{Marktvolumen}}{\text{Marktsättigungsgrad}} \cdot 100\% = \frac{11{,}122 \cdot 10^9 €}{28\%} \cdot 100\% = 39{,}721 \cdot 10^9 €$$

Das Marktpotenzial beträgt 39,721 Milliarden Euro für ein in Deutschland ansässiges international agierendes Unternehmen. Dieses Potenzial entspricht dem von deutschen Unternehmen nicht abgedeckten Markt und ist damit nur ein Indikator für die Wirtschaftlichkeit des Produktes. Die TIG Automation GmbH wird nur einen Bruchteil des Marktpotenzials abdecken können, jedoch wird durch das bestehende Marktpotenzial die Relevanz des Produkts ersichtlich. [16]

2.2 Technische Grundlagen

In diesem Kapitel werden Grundlagen zur technischen Umsetzung erläutert, die in dieser Arbeit verwendet werden. Es werden Übertragungsstandards, Methodiken zur Entwicklung von Software und Grundlagen der Automatisierungstechnik erläutert.

2.2.1 Message Queuing Telemetry Transport (MQTT)

Das Message Queuing Telemetry Transport (MQTT) ist ein, in dem IIoT standardisiertes, Netzwerkprotokoll für die Maschine-zu-Maschine-Kommunikation und wird in zwei Netzwerkspezifikationen unterteilt. Die Unterteilung der Spezifikation wird in TCP/IP- und Nicht-TCP/IP-Netzwerke vorgenommen. Das MQTT ist ein Client-Server-Protokoll mit geringer Bandbreite, wobei der Server als Broker fungiert und die gesendeten Daten des Clients anderen Clients zur Verfügung stellt. Dadurch das alle Clients nur mit dem Server verbunden werden und somit eine Sternstruktur vorliegt, benötigen die Clients untereinander keine direkte Verbindung. Der Austausch der Daten wird mit einem Publish-Subscribe-Prinzip realisiert, bei dem der Client zu einem spezifizierten Thema, dem Topic, periodisch die Daten an den Broker sendet, der diese an alle weiteren Clients, die dieses Topic abonniert haben, weiterreicht. Die Clients, die die Nachricht erhalten, können diese beispielsweise verarbeiten, verändern und unter einem anderen Topic veröffentlichen. Das MQTT-Protokoll besitzt eine Einstellung für die Übertragungssicherheit, dem Quality of Service (QOS). Der QOS kann in die Stufen Null, Eins und Zwei eingestellt werden und legt damit fest wie sicher die Nachricht übertragen werden soll. Die Stufe Null ist eine einfache Übertragung, bei der nicht auf den Verlust der Daten geachtet wird. Die Nachricht wird einmal gesendet ohne Erwartung einer Rückmeldung des Brokers, ob diese korrekt übertragen wurde. Die Stufe Null ist somit die schnellste jedoch am wenigsten sicherste Sicherheitsstufe. Die Stufe Eins hingegen sendet so lange periodisch die Nachricht, bis der Broker die Rückmeldung des Erhalts der korrekten Nachricht

sendet. Hierbei sind durch das periodische Übertragen der Nachricht Mehrfacheintragungen möglich. Mit der Stufe Eins wird die Übertragung sichergestellt, jedoch ist es möglich das andere Clients mehrfach die gleiche Nachricht erhalten, was zu Mehrfacheintragungen führen kann. Die Stufe Zwei stellt sicher, dass die Nachricht nur exakt einmal korrekt versendet wird. Hierbei schickt der Client die Nachricht und wartet mit dem nächsten Sendeprozess auf die Rückmeldung des Brokers. Bei Fehlern in der Übertragung gibt der Broker einen Befehl zum erneuten Senden der Daten. Fehler bei der Stufe Zwei des QOS werden durch einen vierfachen Abgleich erreicht, indem die kommunizierenden Teilnehmer das erfolgreiche Erhalten der Daten gegenseitig zurückmelden. Der schematische Client-Server-Aufbau in Sternstruktur mit den Funktionen des Veröffentlichens (Publish) und des Abonnierens (Subscribe) ist in der nachfolgenden Abbildung 5 dargestellt. [17]

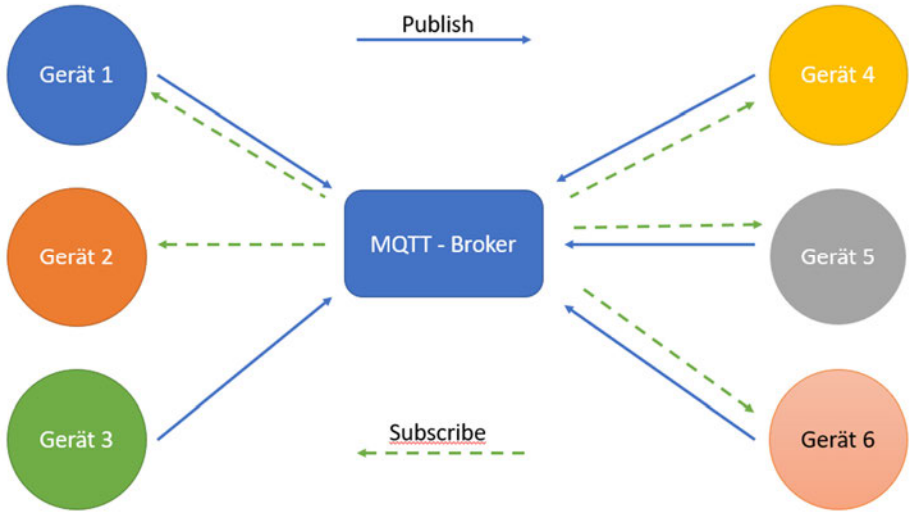


Abbildung 5: Schematischer Client-Server-Aufbau des MQTT

2.2.2 Software-Engineering

Software-Engineering ist eine technische Disziplin und behandelt die Entwicklung, die Pflege und den Einsatz von qualitativ hochwertiger Software. Um eine qualitativ hochwertige Software zu entwickeln, wird mit wissenschaftlichen Methoden gearbeitet. In diesen wissenschaftlichen Methoden werden wirtschaftliche Prinzipien verwendet, Vorgehensmodelle geplant und quantifizierbare Ziele entwickelt. Im Software-Engineering werden mit diesen Methoden Anforderungen aufgestellt und das Design der Software festgelegt. Es wird ein strukturierter Aufbau erzeugt, Tests geplant, sowie die Pflege, Wartung und Weiterentwicklung behandelt. So können mit der Software-Engineering beispielsweise Schnittstellen definiert oder Cloud-Strukturen dargestellt werden. [6]

Am Anfang der Softwareentwicklung werden die Anforderungen an das System ermittelt. Hierzu werden Interessengruppen ausgemacht und befragt. Die bei der Befragung entstehenden Anforderungen werden Benutzeranforderungen genannt und geben Randbedingungen, die das System leisten soll, wieder. Aus den Benutzeranforderungen werden die Systemanforderungen methodisch erarbeitet. [7] Nach der Ermittlung der Benutzeranforderungen wird mit Unified Modelling Language (UML), einer Modellierungssprache, das System modelliert. UML ist ein Zusammenschluss mehrerer unterschiedlicher Modellierungen von Systemen zur Darstellung der Struktur und des Verhaltens. Mit UML werden Spezifikationen, Visualisierungen und Dokumentationen der Software ausgearbeitet. [8] Nachdem die Modelle erstellt wurden, wird das Design festgelegt und anschließende Tests für das System geplant. Im

Software-Engineering wird die umfangreiche Planung der Software und des Systems behandelt, jedoch nicht die Erstellung der eigentlichen Software. Dies ist Teil der Informatik.

2.2.3 Speicherprogrammierbare Steuerung (SPS)

Die speicherprogrammierbare Steuerung (SPS) ist eine Ablaufsteuerung und wird in der Anlagenautomatisierung eingesetzt. Die SPS hat größtenteils die verbindungsprogrammierbare Steuerung (VPS) abgelöst, da hierbei die Logik über die Verdrahtung der Hardware hergestellt wird, was in einem hohen Platzbedarf resultiert und zu hohen Kosten führt. Die SPS ist kostengünstiger und flexibler, da die Logik mit einem Programm umgesetzt wird. Die Aktorik und Sensorik wird direkt an den Ein- und Ausgängen der SPS angeschlossen, um von dem Programm ausgelesen und angesprochen zu werden.

Der Ablauf der Steuerung der SPS erfolgt dabei nach dem EVA-Prinzip (Eingabe Verarbeitung Ausgabe), bei dem zuerst ein Prozessabbild der Eingänge (PAE) erstellt und danach mit diesen Eingangswerten die Logik durchlaufen wird. Nachdem die Logik durchlaufen und das Prozessabbild der Ausgänge (PAA) ermittelt wurden, werden die Ausgänge nach dem PAA geschaltet. Änderungen der Eingänge während des Durchlaufs der Logik haben keinen Einfluss mehr in diesem Zyklus. Das EVA-Prinzip wird jeden Zyklus der SPS durchgeführt. Die SPS ist ein Steuergerät, welches den Zustand des Prozesses mit Sensordaten ausliest und mit Hilfe der Aktorik den nächsten Schritt des Prozesses durchführt (siehe Abbildung 6). [9]

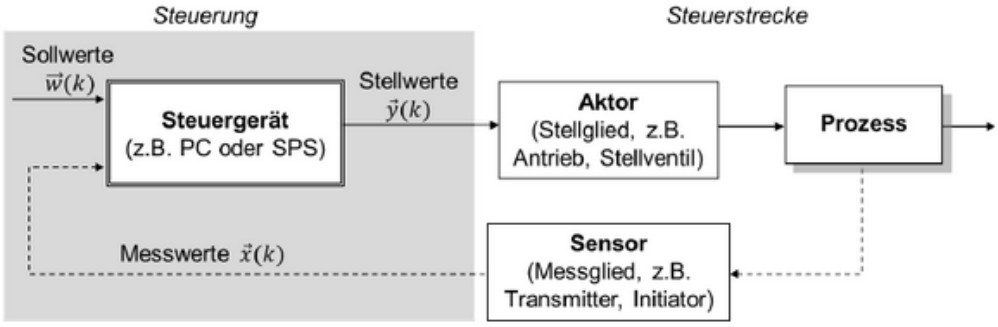


Abbildung 6: Aufbau eines Steuerkreises

Die SPS wird in drei Arten unterschieden: Hardware-SPS, Slot-SPS und Soft-SPS. Die Hardware-SPS ist die klassische Aufbauform im Schaltschrank bei der eine Zentralbaugruppe (CPU) mit Einsteckkarten erweitert wird und benötigt einen Computer als Programmiergerät. Bei der Slot-SPS hingegen handelt es sich um eine Einsteckkarte für den PC. Diese Einsteckkarte enthält alle Module einer SPS und benutzt den verwendeten Computer als Stromversorgung. Der Vorteil der Slot-SPS ist die starke Vernetzung beider Arbeitsspeicher miteinander, was in schnellen Arbeitsprozessen resultiert. Eine Soft-SPS ist eine reine durch Software visualisierte SPS und nutzt die CPU und den Arbeitsspeicher des Computers. [9]

Die Signale der Ein- und Ausgänge der Feldgeräte (Aktorik und Sensorik) sind binär oder analog. Hierfür gibt es bei der SPS spezielle analoge oder binäre Ein- und Ausgangskarten, an denen die Feldgeräte angeschlossen werden können. Binäre Signale sind Signale, die einer Null oder einer Eins entsprechen und haben somit nur die beiden Zustände eingeschaltet und ausgeschaltet. Sensoren mit binären Signalen sind beispielsweise von Lichtschranken oder Endlagenschalter. Binäre Signale für Aktoren werden zum Beispiel zum Öffnen von Ventilen oder zum Einschalten von Lampen benötigt. Analoge Signale spannen im Gegensatz zu binären Signalen einen Messbereich in Volt oder Ampere auf und können so unendliche viele Zustände annehmen. Solche analogen Sensorsignale können von Füllständen kommen oder als Stellsignal für Regelventile verwendet werden. Ebenfalls können Feldgeräte mit Hilfe von Bussystemen eine Verbindung zur SPS aufbauen (siehe Abbildung 7). Durch die Verwendung von Bussystemen wird Material eingespart und der Platzbedarf der Leitungen deutlich minimiert, da die

Kommunikation über eine Leitung und nicht über mehrere Leitungen verwaltet wird. Jedoch muss jeder Teilnehmer am Bus eine passende Schnittstelle besitzen und die Kommunikation aktiv von den Teilnehmern überwacht werden. Eine Lösungsmöglichkeit für das Fehlen einer Schnittstelle ist das Einsetzen einer dezentralen Peripherie. [9]

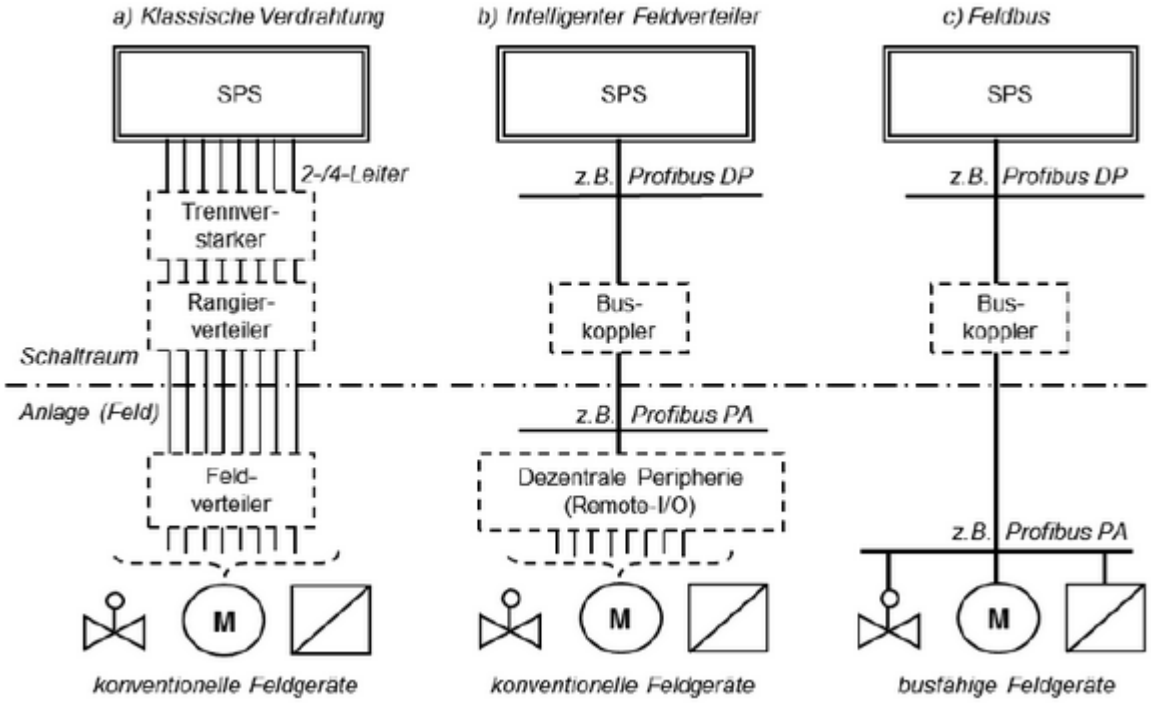


Abbildung 7: Aufbau der Verdrahtung der SPS

Zur Visualisierung der Prozessdaten und des Anlagenzustandes wird das HMI der SPS verwendet. Das HMI ist die Schnittstelle zum Menschen, bei dem der Bediener aktiv in den Prozess eingreifen, diesen beobachten oder Prozessdaten anzeigen lassen kann. Das HMI kann den Zustand der Anlage wie beispielsweise aktive Antriebe oder vorliegende Meldungen anzeigen und ermöglichen Pumpen, Ventile oder Regler im Handbetrieb anzusteuern. [9]

Um Prozessdaten auf der Prozessleitebene kontinuierlich erfassen, auswerten, filtern und dokumentieren zu können, wird das „Supervisory Control And Data Acquisition"-System, kurz SCADA-System, eingesetzt. Das SCADA-System ist ein System zur Beaufsichtigung der Steuerung und Sammlung von Prozessdaten und wird häufig zur Umsetzung von Instandhaltungsstrategien verwendet. Die HMI-Panels sind mit dem SCADA-System verbunden und werden vom System mit den benötigten Prozessdaten versorgt. Das SCADA-System befindet sich in der Prozessleitebene und ist somit der Steuerungsebene, in der die speicherprogrammierbaren Steuerungen sich befinden, übergestellt. So können in einem SCADA-System Prozessdaten mehrerer speicherprogrammierbarer Steuerungen hinterlegt werden, um eine Zusammenarbeit und eine Kommunikation herzustellen. [15]

2.3 Ausgangssituation

In diesem Abschnitt werden die vorausgesetzten und bestehenden Systemkomponenten aufgezeigt und erläutert. Die zu analysierende Fassreinigungsanlage wird dargestellt und verbaute Aktorik und Sensorik, sowie deren Signale aufgezeigt. Ebenfalls wird die Funktion der Fassreinigungsanlage und der Ablauf des Anlagenprozesses erläutert.

2.3.1 Fassreinigungsanlage

Die Funktion der Fassreinigungsanlage ist es Industriefässer, welche zur Produktaufbewahrung verwendet werden, von Produktresten zu reinigen. Für die Reinigung der Fässer wird zuerst die Anlage geschlossen und verriegelt. Nach erfolgreichem Schließen und Verriegeln der Fassreinigungsanlage wird die Anlage inertiesiert, also mit einem Inertgas, in diesem Fall Stickstoff, begast. Die Inertisierung wird im Explosionsschutz für ausgasende Stoffe angewendet, die in Verbindung mit Luftsauerstoff ein leicht entzündliches Gemisch bilden, um so jegliches Explosionsrisiko zu verhindern. Nach der Inertisierung der Anlage wird diese geöffnet und das verunreinigte Fass mit der Öffnung nach unten in die Anlage gestellt. Anschließend wird erneut mit dem Fass inertisiert, bevor es mit dem Lösemittel Aceton gespült wird. Das mit den Produktresten verunreinigte Aceton läuft an der Fasswand nach unten und wird nach dem Abtropfen am Boden gesammelt. Nachdem der Spülvorgang mit Aceton abgeschlossen ist, wird das Fass mit Ethylacetat, einem weiteren Lösemittel, gespült. Das verunreinigte Lösemittel tropft ebenfalls auf den Behälterboden. Nach Abschluss des Spülvorgangs mit Ethylacetat wird das Fass wird mit der Lösemittelmischung aus Aceton und Ethylacetat, die sich am Boden der Anlage gesammelt hat, gespült. Die Dauer des Spülvorgangs mit der Lösemittelmischung stellt der Bediener individuell ein, wonach die Mischung in einen Abfallbehälter gepumpt wird. Vor der Entnahme des Fasses muss dieses nochmal inertisiert werden, damit keine Reaktion der Gase entstehen kann. Nach der Entnahme des gereinigten Fasses kann der Reinigungsprozess mit einem neuen Gebinde gestartet werden. In der nachfolgenden Abbildung ist der Prozessablauf mit einem Petrinetz dargestellt (siehe Abbildung 8).

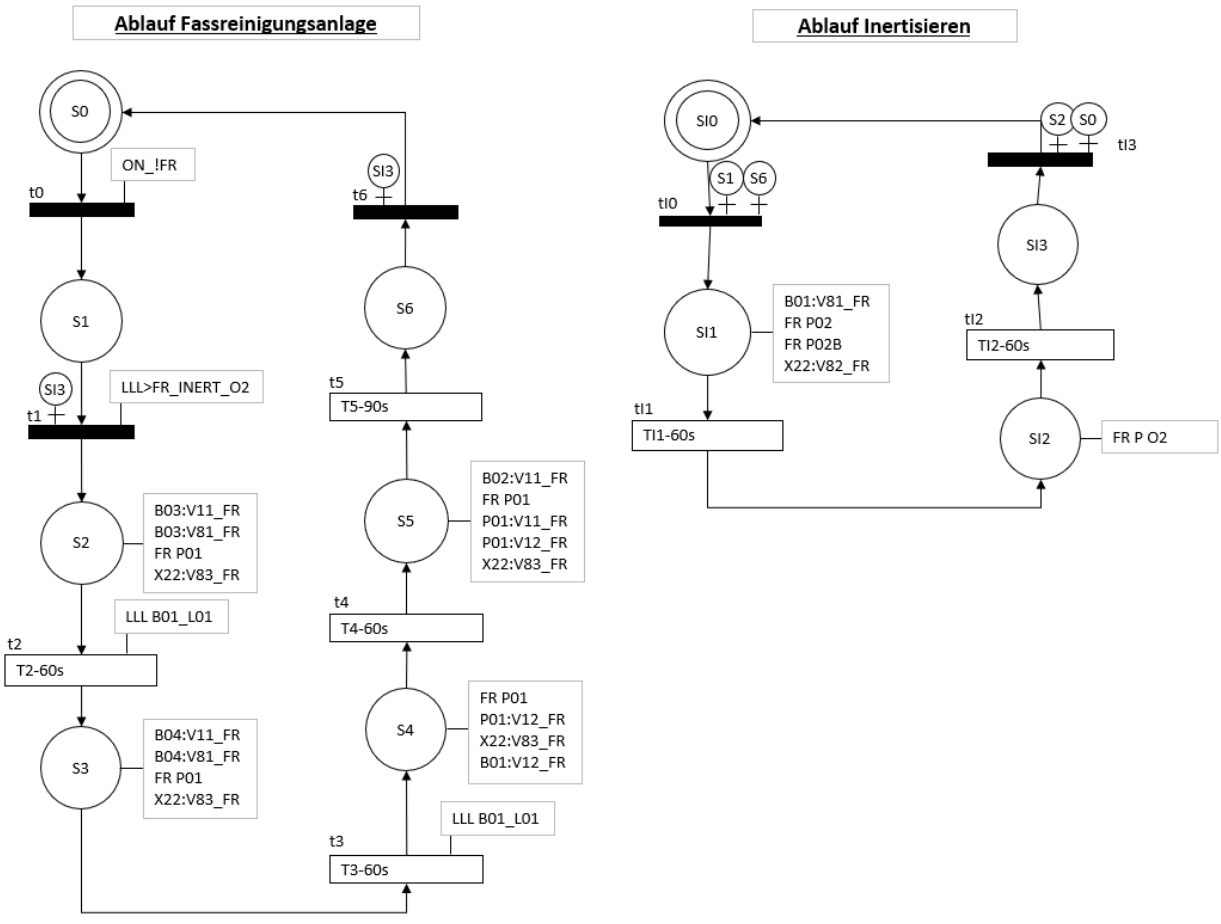


Abbildung 8: Schrittkette der Fassreinigungsanlage als Petrinetz

Die Endlagen der Ventile und Kugelhähne werden im Petrinetz aus Gründen der Übersichtlichkeit nicht an die Transitionen angetragen. Ebenfalls werden die Füllstände der Behälter nicht dargestellt, da diese nur als Start-Bedingung des Automatikbetriebs abgerufen und somit in der Variable „ON_!FR“ berücksichtigt werden. Die Automatisierung der Fassreinigungsanlage wird mit einer SPS der Firma Siemens realisiert. Die SPS ist eine S7-1500 und hat mehrere analoge und binäre Ein- und Ausgangskarten. An diese analogen und binären Ein- und Ausgangskarten sind die Sensoren und Aktoren der Fassreinigungsanlage angeschlossen, die durch das Ablaufprogramm abgefragt und angesprochen werden. Die Steuerung der Fassreinigungsanlage ist nur eine der Anlagensteuerungen, die diese SPS ausführt. In der nachfolgenden Tabelle sind die Signale der Fassreinigungsanlage mit den verwendeten Ein- und Ausgängen der SPS, den dazugehörigen Adressen, den im Ablaufprogramm verwendeten symbolischen Namen, der Art des Signals und eine Kurzbeschreibung aufgeführt (siehe Tabelle 1).

Adresse	symb. Name	Signalart	Beschreibung
E600.0	LLL B01_L01	binär	Minimaler Füllstand des Behälters B01 erreicht
E601.0	ON_!FR	binär	Starttaster gedrückt
E600.4	HHH-B01_FR	binär	Maximaler Füllstand des Abfallbehälters B01 erreicht
E603.0	LLL>FR_INERT_O2	binär	Minimaler Schwellwert für Sauerstoffkonzentration erreicht
E612.0	OPN B01_V12_FR	binär	Endlage Kugelhahn V12 vom Behälter B01 erreicht (Offen)
E612.1	CLS B01_V12_FR	binär	Endlage Kugelhahn V12 vom Behälter B01 erreicht (Geschlossen)
E612.2	OPN B01_V20_FR	binär	Endlage Kugelhahn V20 vom Behälter B01 erreicht (Offen)
E612.3	CLS B01_V20_FR	binär	Endlage Kugelhahn V20 vom Behälter B01 erreicht (Geschlossen)
E612.4	OPN B02_V11_FR	binär	Endlage Kugelhahn V11 vom Behälter B02 erreicht (Offen)
E612.5	CLS B02_V11_FR	binär	Endlage Kugelhahn V11 vom Behälter B02 erreicht (Geschlossen)
E612.6	OPN B02_V81_FR	binär	Endlage Kugelhahn V81 vom Behälter B02 erreicht (Offen)
E612.7	CLS B02_V81_FR	binär	Endlage Kugelhahn V81 vom Behälter B02 erreicht (Geschlossen)
E615.0	OPN B03_V11_FR	binär	Endlage Kugelhahn V11 vom Behälter B03 erreicht (Offen)
E615.1	CLS B03_V11_FR	binär	Endlage Kugelhahn V11 vom Behälter B03 erreicht (Geschlossen)
E615.2	OPN B04_V11_FR	binär	Endlage Kugelhahn V11 vom Behälter B04 erreicht (Offen)
E615.3	CLS B04_V11_FR	binär	Endlage Kugelhahn V11 vom Behälter B04 erreicht (Geschlossen)
E615.4	OPN P01_V11_FR	binär	Endlage Kugelhahn V11 der Pumpe P01 erreicht (Offen)
E615.5	CLS P01_V11_FR	binär	Endlage Kugelhahn V11 der Pumpe P01 erreicht (Geschlossen)
E615.6	OPN P01_V12_FR	binär	Endlage Kugelhahn V12 der Pumpe P01 erreicht (Offen)
E615.7	CLS P01_V12_FR	binär	Endlage Kugelhahn V12 der Pumpe P01 erreicht (Geschlossen)
E618.0	OPN X03_V85_FR	binär	Endlage Absperrkugelhahn V85 der Pumpen P02A/B zur Abluft erreicht (Offen)
E618.1	CLS X03_V85_FR	binär	Endlage Absperrkugelhahn V85 der Pumpen P02A/B zur Abluft erreicht (Geschlossen)
E618.2	OPN X22_V82_FR	binär	Endlage Absperrkugelhahn V82 vom Behälter B01 zur Abluft erreicht (Offen)
E618.3	CLS X22_V82_FR	binär	Endlage Absperrkugelhahn V82 vom Behälter B01 zur Abluft erreicht (Geschlossen)
E618.4	OPN X22_V83_FR	binär	Endlage Absperrklappe V83 der Sonderabluft des Behälters B01 erreicht (Offen)

E618.5	CLS X22_V83_FR	binär	Endlage Absperrklappe V83 der Sonderabluft des Behälters B01 erreicht (Geschlossen)
EW670	FR B02:L02	analog	Füllstand des Abfallbehälters B02
EW672	FR B03:L01	analog	Füllstand des Behälters B03 für Aceton
EW674	FR B04:L01	analog	Füllstand des Behälters B04 für Ethylacetat
A13.0	FR P02	binär	Anschalten der Pumpe P02A
A13.1	FR P02B	binär	Anschalten der Pumpe P02B
A13.2	FR P O2	binär	Anschalten der Pumpe zur Sauerstoffmessung
A610.0	B01:V12_FR	binär	Öffnen des Kugelhahns V12 vom Behälter B01
A610.1	B01:V20_FR	binär	Öffnen des Kugelhahns V20 vom Behälter B01
A610.2	B02:V11_FR	binär	Öffnen des Kugelhahns V11 vom Behälter B02
A610.3	B02:V81_FR	binär	Öffnen des Kugelhahns V81 vom Behälter B02
A610.4	B03:V11_FR	binär	Öffnen des Kugelhahns V11 vom Behälter B03
A610.5	B04:V11_FR	binär	Öffnen des Kugelhahns V11 vom Behälter B04
A610.6	FR P01	binär	Anschalten der Pumpe P01
A610.7	P01:V11_FR	binär	Öffnen des Kugelhahns V11 der Pumpe P01
A612.0	P01:V12_FR	binär	Öffnen des Kugelhahns V12 der Pumpe P01
A612.1	X03:V85_FR	binär	Öffnen des Absperrkugelhahns V85 der Pumpen P02A/B zur Abluft
A612.2	X22:V82_FR	binär	Öffnen des Absperrkugelhahns V82 vom Behälter B01 zur Abluft
A612.3	X22:V83_FR	binär	Öffnen der Absperrklappe V83 der Sonderabluft des Behälters B01
A612.4	B01:V81_FR	binär	Öffnen des Magnetventils V85 vom Behälter B01 zur Stickstoffzufuhr
A612.5	B03:V81_FR	binär	Öffnen des Magnetventils V85 vom Behälter B03 zur Acetonzufuhr
A612.6	B04:V81_FR	binär	Öffnen des Magnetventils V85 vom Behälter B04 zur Ethylacetatzufuhr
A612.7	P01:V75_FR	binär	Öffnen des Magnetventils V75 der Pumpe P01 für Druckluft

Tabelle 1: Signale der Fassreinigungsanlage

Die Fassreinigungsanlage ist eine, für die Produktion, sekundäre Anlage und wird nur sporadisch verwendet. Da die Fassreinigungsanlage bereits vollständig automatisiert ist, ist die genaue Anordnung der Bauteile nicht relevant. Die Bauweise der Aktorik und Sensorik ist essenziell für die Erstellung der Analysekonzepte zur vorbeugenden Instandhaltung. Nachfolgend ist sowohl eine Stückliste der eingebauten Aktorik und Sensorik (siehe Tabelle 3), sowie Bilder der Fassreinigungsanlage (siehe Abbildungen 9, 10, 11) eingefügt. Der schematische Aufbau der Fassreinigungsanlage befindet sich als Fließbild in Anhang B.

Stückliste der Aktorik und Sensorik		
Pos.	Bezeichnung	Anzahl
1	Sicherheitszuhaltung; EX AZM415 2Ö/2S R	1
2	Beweglicher Betätiger; AZM 415-B2	1
3	Kugelhahn; Kugelhahn-ZK, DN15, mit Antrieb-EE, EW55, Edelstahl 1.4408/PTFE-FKM, federrückstellend, EX-geschützt	1
4	Magnetventil; Typ 6027, Art. Nr. 298657, Zone 2 (ATEX)	5
5	Niveauschalter; Liquiphant FTL51-F29U1/0, FTL51_FTC2DB8N3A, Triclamp DN25-38	2
6	Pumpe; Druckluftmembranpumpe E15_UTT-F4	1
7	3 Wege Kükenhahn; 3 Wege Kükenhahn, L-Küken 90°, Typ 8325 mit Endschalter 8,2 V DC, PN64	2
8	Kugelhahn; ZA310062, PN64	2
9	Membranpumpe; N150.1.2St.9 E EX, 1,1 kW, 400 V	2
10	Niveaugrenzschalter; Liquiphant FTL51-F29T4/0, FTL51_FTC2BB8N3A, Triclamp DN25-38	2
11	Kugelhahn; ZA62-EE55, Kugelhahn ZA DN15-Anschweißende, mit Antrieb EE55, Edelstahl/PTFE-FKM, federrückstellend	2
12	Absperrarmatur; ZA64-EE55, Kugelhahn ZA DN25-Anschweißende, mit Antrieb EE63, Edelstahl/PTFE-FKM, federrückstellend, EX-geschützt	1
13	Absperrarmatur; WM534007	1

Tabelle 2: Stückliste der Aktorik und Sensorik der Fassreinigungsanlage

In der Abbildung 9 ist der Reinigungsbehälter der Fassreinigungsanlage abgebildet.

Abbildung 9: Reinigungsbehälter der Fassreinigungsanlage

Die Lösemittel- und Abfallbehälter der Fassreinigungsanlage für die Lösemittel Aceton und Ethylacetat, sowie für das Abfallgemisch aus beiden Lösemitteln, sind auf der Abbildung 10 zu sehen. Die Behälter

sind einfach von den Leitungen zu trennen und besitzen Rollen, damit diese befüllt und entleert werden können.

Abbildung 10: Lösemittelbehälter der Fassreinigungsanlage

Auf der nachfolgenden Abbildung 11 ist der Innenraum des Reinigungsbehälters der Fassreinigungsanlage zu sehen. Ebenfalls sind die sechs Reinigungsdüsen zu erkennen, welche in der Abbildung rot eingekreist und nummeriert sind. Die Reinigungsdüsen 1 - 4 sind zur seitlichen Reinigung, die Düse 5 reinigt das Innere des Gebindes und die Düse 6 reinigt das zu reinigende Fass von oben. Im Gegensatz zu allen anderen starren Düsen ist die Düse 5 rotierend, um eine flächendeckende Reinigung der inneren Wände des Behälters zu erzielen.

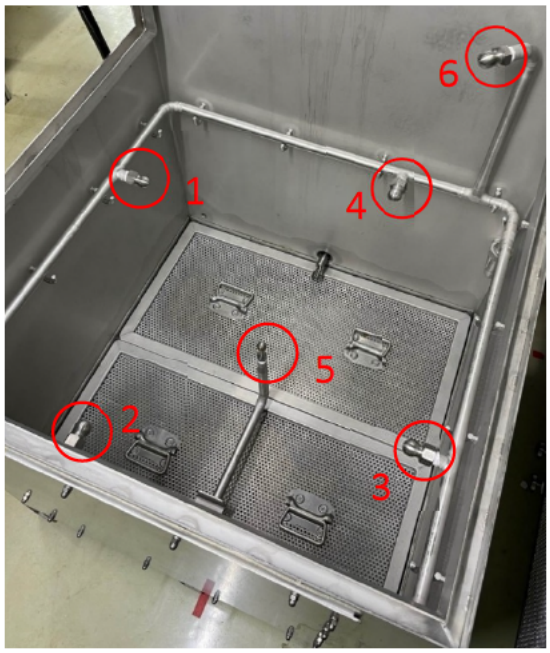


Abbildung 11: Reinigungsbehälter der Fassreinigungsanlage

2.3.2 EWON Flexy 205

Der EWON Flexy 205 ist ein Produkt der Firma HMS Industrial Networks GmbH. Die Produktreihe des EWON wird in Produktionen als Gateway und zur Fernwartung von industriellen Anlagen eingesetzt. Der EWON verbindet das lokale Maschinennetzwerk sicher mit dem Internet und wird bereits beim Kunden zur Fernwartung verwendet. [10]

Die HMS Industrial Networks GmbH bietet zwei Programme, um auf den EWON zugreifen und diesen konfigurieren zu können. Für den Zugriff vom lokalen Netzwerk wird das Programm „eBuddy“ verwendet, wofür eine einfache Authentifizierung notwendig ist. Der „eBuddy“ durchsucht das lokale Netzwerk nach Produkten der Firma HMS und listet diese mit Namen und der IP-Adresse auf. Aus dem „eBuddy“ heraus ist es möglich die Konfigurationsseite im Browser zu öffnen, wobei jeder EWON individuell mit Benutzername und Passwort abgesichert werden kann. Das zweite Programm, welches auch für die Fernwartung verwendet wird, ist der „eCatcher“ von HMS. Der „eCatcher“ hat eine mehrstufige Sicherheit, da aus dem globalen Netzwerk auf den EWON zugegriffen wird. Das Programm fragt beim Aufruf ein Kundenkonto der Firma HMS ab, um sich einzuloggen und mit der HMS eigenen Cloud zu verbinden. In der HMS-Cloud werden alle mit diesem Kundenkonto erreichbaren EWON abgebildet und der Betriebszustand dargestellt. Um über den EWON in das lokale Netzwerk des Kunden zu gelangen, ist ebenfalls eine Authentifikation nötig. Sobald die Identifikation für das lokale Netzwerk erfolgreich abgeschlossen wurde, ist es möglich auf die im Netzwerk befindlichen Anlagen zuzugreifen und eine Fernwartung zu beginnen. Für die Konfiguration des EWON hingegen, wird beim Aufrufen über den „eBuddy“ wieder eine für jeden EWON individuelle Authentifikation benötigt.

Mit dem EWON können IIoT-Lösungen, zum Beispiel zur vorbeugenden Instandhaltung, umgesetzt werden, da dieser als Gateway fungieren kann. Die Konfiguration des EWON unterstützt viele Übertragungsstandards wie OPC UA, MQTT oder ModBus und ist somit individuell einstellbar. Ebenfalls können im EWON Prozessdaten zur Weiterversendung dargestellt werden, welche mit einer einstellbaren Taktzeit aus der SPS entnommen werden. Die EWON-Konfiguration bietet einige Einstellmöglichkeiten zur Sicherheit der Daten, wie beispielsweise den Schreibschutz der Prozessdaten, Einstellungen der Sicherheitskategorie des Übertragungsstandards oder durch das Erstellen von Lizenzen und Zertifikaten. Im EWON ist es ebenfalls möglich Prozesse in einer BASIC IDE oder in der Programmiersprache JAVA zu programmieren. Die Programmiersprache BASIC ist der Programmiersprache C von der Syntax sehr ähnlich, von den Funktionen jedoch stark erweitert und auf den Anwendungsbereich des EWON angepasst. Mit den Prozessen aus der BASIC IDE lassen sich Verbindungen zu Clouds herstellen oder Prozessdaten weiterverarbeiten.

2.3.3 Qlik Sense

Qlik Sense ist eine Cloud-basierte Plattform zur Verarbeitung, Analyse und Darstellung von Prozessdaten und wird von der Firma QlikTech International AB vertrieben. Mit der Analysesoftware ist es möglich unterschiedlichste Konnektivitäten zu erzeugen, um die Prozessdaten zu importieren. Die Prozessdaten können entweder direkt aus Datenbanken erhalten, von Programmen geliefert, als Dateien importiert oder von anderen Cloud-Diensten bereitgestellt werden. Die Prozessdaten werden anschließend gefiltert und sortiert, bevor sie bearbeitet, verarbeitet und gruppiert werden. Nachdem die Prozessdaten für die Analyse vorbereitet sind, werden die anlagenspezifischen Analysen entwickelt und geschrieben. Qlik Sense bietet viele Darstellungsmöglichkeiten für die durchgeführten Analysen, womit diese Analysen aussagekräftig dargestellt werden können. Ebenfalls bietet die Software von QlikTech eine eigene künstliche Intelligenz, sowie die Möglichkeit such- und dialogorientierte Analysen durchzuführen. Das Analysewerkzeug besitzt vorgefertigte Verfahren zur Benachrichtigung und zur Warnung der Benutzer.

Die Software Qlik Sense ist mit der eigenen Datenhaltung und den bereitgestellten Werkzeugen leicht skalierbar und somit für Anlagen jeglicher Größe verwendbar. [1]

3. Analyse der Anforderungen

Um die Konzeption und Entwicklung des Systems durchführen zu können, müssen die Anforderungen an das Gesamtsystem definiert und in einer Anforderungsliste festgehalten werden. In der Tabelle 3 sind diese Anforderungen aufgelistet. Die Anforderungen werden in der Anforderungsliste in Wünsche und Forderungen des Kunden unterteilt. Forderungen des Kunden sollen zwingend umgesetzt werden im Gegensatz zu den Wünschen, welche nach Möglichkeit beachtet werden sollen, jedoch nicht notwendig sind. [11]

Forderung /Wunsch	ID	Anforderung	Wert
F	1	SPS aktueller Generation S7-1500 von Siemens	
F	2	SCADA-System WinCC 7.5 von Siemens	
W	3	EWON Flexy 205 als Gateway	
F	4	Verwendung einer Cloud-Datenhaltung	
F	5	Qlik Sense als Analyseplattform	
F	6	automatische Übertragung der Daten	
F	7	Filterung der Daten auf SPS-Ebene	
F	8	sequenzielle Übertragung von SPS zu SCADA	
F	9	Optimierung der Datenmenge	
W	10	differenzielle Datenübertragung von SPS zu Cloud	
W	11	leichte Skalierbarkeit der Datenstruktur	
W	12	Wiederverwendbarkeit der Software	
F	13	Schutz vor unberechtigtem Datenzugriff von außen	
W	14	Benachrichtigungen für erforderliche Wartungen	
F	15	Darstellung von Verlaufsanalysen	
F	16	Darstellung der Prozessdaten	
F	17	vorbeugende Instandhaltung für Sensorik und Aktorik	
F	18	Ende der Entwicklungszeit	02.12.2021
F	19	Zugang zum globalen Internet (WAN)	
F	20	Datenschutz der Prozessdaten	
W	21	fortlaufende Kosten	max. 250€ / Monat
F	22	Anschlussenergie	230 V
W	23	Kosten für Umsetzung	max. 1000 €

Tabelle 3: Anforderungsliste

Die Anforderungen werden aus den Forderungen und Wünschen des Kunden und den Rahmenbedingungen an das Projekt abgeleitet.

4. Konzeption

In der Konzeptionsphase wird das Projekt in zu konzipierende Teilkomponenten unterteilt und Konzepte für die Umsetzung der Teilfunktionen des Produkts entwickelt. Anschließend werden die entwickelten Konzepte mit der Anforderungsliste abgeglichen und die umzusetzenden Konzepte ausgewählt.

4.1 Überblick der zu konzipierenden Komponenten

Die Prozessdaten der Fassreinigungsanlage müssen zur prozessnahen Darstellung auf den HMI-Panels umfangsoptimiert an das SCADA-System übermittelt werden. Hierbei muss bei der Übertragung der Prozessdaten sichergestellt werden, dass keine Übertragungsfehler entstehen. Des Weiteren werden die Prozessdaten von der SPS dem Gateway bereitgestellt und von diesem aus weiter an die Cloud versendet. Bei der Auswahl des Gateways sind die Übertragungsstandards und die verwendbaren Sicherheitsoptionen besonders relevant, um aus dem globalen Netzwerk nicht auf die Anlagenprozesse zugreifen und in diese eingreifen zu können. Ebenfalls muss eine Cloud-Struktur erstellt werden, bei denen die übermittelten Prozessdaten erkannt, sortiert und gespeichert werden. Die, in der Cloud gespeicherten, Prozessdaten müssen der Analyseplattform Qlik Sense übergeben und Analysen zur vorbeugenden Instandhaltung umgesetzt werden. In der nachfolgenden Abbildung (siehe Abbildung 12) ist der schematische Aufbau des Gesamtkonzepts abgebildet.

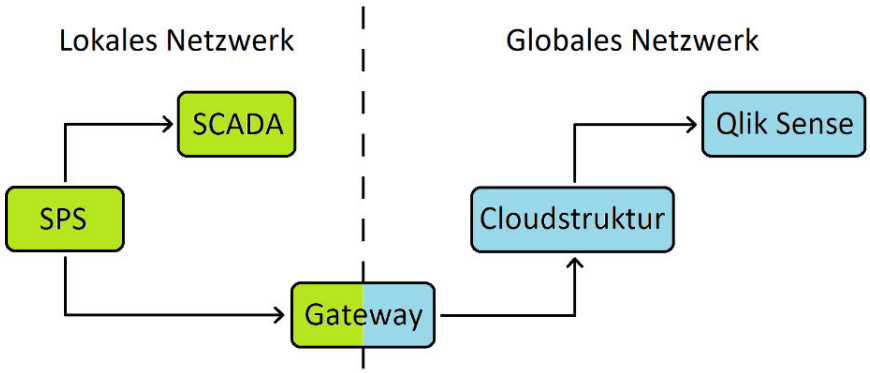


Abbildung 12: Schematischer Aufbau des Gesamtkonzepts

Die aus der Erläuterung des Gesamtkonzeptes abgeleiteten Teilkomponenten werden in der nachfolgenden Liste aufgezeigt:

- SPS-Baustein zur sicheren und datenumfangsoptimierten Übermittlung der Prozessdaten von SPS zu SCADA-System
- Auswahl des Gateways
- Übermittlung der Daten von SPS über Gateway an Cloud
- Erstellung der Cloud-Struktur
- Prozessdatenübergabe von der Cloud zu Qlik Sense
- Erstellung der Analysen zur vorbeugenden Instandhaltung in Qlik Sense

4.2 Erarbeitung verschiedener Grundkonzepte und Abgleich mit der Anforderungsliste

Bei der Erarbeitung der verschiedenen Teilkomponenten des Projekts, werden realisierbare Einzelkonzepte entwickelt, um nach Evaluation dieser, die für das Projekt beste Teilkonzept auswählen zu können. Ein wichtiges Augenmerk wird hierbei auf die Definition der Schnittstellen und der Übertragungsstandards gelegt, da eine große Abhängigkeit der Teilkomponenten untereinander vorliegt.

4.2.1 Planung des datenumfangsoptimierenden SPS-Bausteins

Der datenumfangsoptimierende SPS-Baustein soll die Prozessdaten der Anlage umfangsoptimiert dem SCADA-System zur Verfügung stellen, um die Datenlast im Netzwerk und die Kosten der SCADA-Lizenzen möglichst gering zu halten. Die zu erwerbende Lizenz für das SCADA-System wird anhand der, vom SCADA-System aus der speicherprogrammierbaren Steuerung zu entnehmenden, Variablen ausgelegt. Um die Vergrößerung dieser Variablenanzahl und damit das Erwerben einer teureren Lizenz zu verhindern, sollen die Prozessdaten nacheinander sequenziell dem SCADA-System bereitgestellt werden. Ebenfalls werden beim Anlegen einer direkten Entnahme der Prozessdaten aus der SPS, die abzufragenden Werte zyklisch aus dem SCADA-System entnommen, was zu einer hohen Datenlast und einem Anstieg der Zykluszeit führt. Die Zykluszeit bei Anlagenautomatisierungen ist die Zeit, die benötigt wird, um das Automatisierungsprogramm einmal vollständig zu durchlaufen und ist ausschlaggebend für die Reaktionsgeschwindigkeit der Anlagensteuerung. Daher sollte die Zykluszeit immer so gering wie möglich gehalten werden.

Die Prozessdaten, die an das SCADA-System zu übermitteln sind, werden in Datenbausteinen nach Datentypen sortiert eingetragen. Um diese Prozessdaten auslesen zu können, muss die Länge der Datenbausteine ermittelt werden. Da ein Funktionsbaustein für alle Datentypen erstellt werden soll, muss der zu übertragende Datentyp erkannt werden. Ebenfalls müssen die Daten zur Übertragung in Doppelwörter umgewandelt werden, um eine einheitliche Übertragung zu bewirken. Das SCADA-System greift die Daten von der SPS ab, jedoch können hierbei Übertragungsfehler auftreten. Diese Übertragungsfehler der übergebenen Prozessdaten müssen verhindert werden. Nach Überprüfung der Prozessdaten muss der nächste Datenpunkt übermittelt werden. Hierfür muss eine Iteration über den Datenbaustein umgesetzt werden.

Für die zuvor beschriebenen Probleme werden Lösungsvorschläge ermittelt und in dem nachfolgenden morphologischen Kasten eingetragen. Der morphologische Kasten ist eine Methodik zur Aufstellung verschiedener Konzepte. Hierbei werden die Teilfunktionen und Probleme in der linken Spalte und die einzelnen realisierbaren Lösungsvorschläge in den Zeilen rechts daneben eingetragen. Nachdem alle Problemlösungen eingetragen sind, werden farbliche Markierungen eingetragen, bei dem jeder Pfad einem Konzept entspricht. Die Lösungsvorschläge werden willkürlich erstellt und im Nachhinein ermittelt, welcher Lösungsvorschlag realisiert werden soll. Die Teillösungen müssen realisierbar sein, jedoch muss nicht jede Teillösung in einem Lösungskonzept geplant werden. Die Abbildung 13 zeigt den morphologischen Kasten mit den farblich eingetragenen Lösungskonzepten [18].

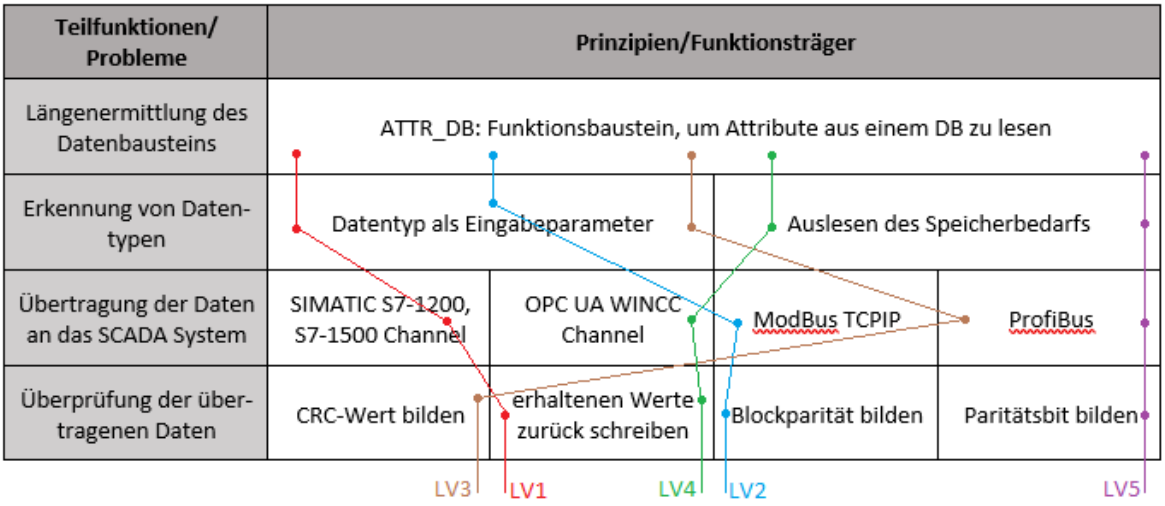


Abbildung 13: Morphologischer Kasten des SPS-Bausteins

Um eine Auswahl des Konzeptes treffen zu können, werden drei Konzepte von denen im morphologischen Kasten eingetragenen Lösungsvorschlägen (LV1-5) ausgewählt. Dies ist eine Vorauswahl, bei der keine Bewertung stattfindet. Nach der Vorauswahl werden die Konzepte nach technischen und wirtschaftlichen Kriterien mit einer Nutzwertanalyse bewertet. In der Nutzwertanalyse werden Bewertungskriterien aufgetragen und mit einem Gewichtungsfaktor beaufschlagt. Die Summe aller Gewichtungsfaktoren ergibt 1 und entspricht damit 100%. Die verschiedenen Lösungen werden für jedes Bewertungskriterium von 0 bis 5 bewertet, wobei 0 einer schlechten und 5 einer guten Eignung entspricht. Die individuelle Bewertung wird mit dem Gewichtungsfaktor multipliziert und als Teilnutzwert eingetragen. Die Teilnutzwerte werden summiert und ergeben den Gesamtnutzwert, aus dem hervorgeht welcher Lösungsansatz realisiert werden sollte. In den folgenden Abschnitten werden die ausgewählten Lösungskonzepte, die Bewertungskriterien und die Gewichtungsfaktoren erläutert.

Als Vorauswahl werden die Lösungsvorschläge 1, 3 und 4 ausgewählt. Diese werden anhand der im nächsten Absatz beschriebenen Bewertungskriterien bewertet, mit den darauffolgend erläuterten Gewichtungsfaktoren belegt und die Nutzwertanalyse zur Ermittlung des höchsten Gesamtnutzwertes durchgeführt (siehe Tabelle 4).

Die Bewertungskriterien, die in der Nutzwertanalyse zur Bewertung der Lösungsvorschläge verwendet werden, sind wirtschaftliche und technische Kriterien. Die wirtschaftlichen Kriterien sind die erzeugten Kosten und der geschätzte benötigte Umsetzungsaufwand. Zu den technischen Kriterien zählt die Übertragungsgeschwindigkeit und die Kontrolle der übertragenen Daten. Die Anforderung der automatischen Übertragung ist sowohl ein wirtschaftliches als auch ein technisches Kriterium.

Um die Gewichtungsfaktoren festzulegen, werden die Bewertungskriterien nach Wichtigkeit sortiert und die Faktoren vergeben. Das wichtigste Kriterium ist die Überprüfung der übertragenen Daten gefolgt von den, durch die Umsetzung, erzeugten Kosten. An dritter Stelle ist die Erstellung der automatischen Übertragung. Vor der Übertragungsgeschwindigkeit an fünfter Stelle ist der Umsetzungsaufwand. Der Umsetzungsaufwand spiegelt die Komplexität und den Zeitaufwand zur Umsetzung wider und ist damit nur vom Ersteller abhängig. Die Übertragungsgeschwindigkeit ist am wenigsten von Bedeutung, da die Prozessdaten, die übertragen werden, nicht in Echtzeit vorliegen müssen. Nachfolgend ist die Nutzwertanalyse eingefügt (siehe Tabelle 4). [19]

			LV1		LV3		LV4	
	Bewertung von 0 - 5 (0 schlecht, 5 gut)	Gewichtungs-faktor	Bewertung	Teilnutzwert	Bewertung	Teilnutzwert	Bewertung	Teilnutzwert
Bewertungskriterien	Kontrolle der übertragenen Daten	0,3	4	1,20	2	0,60	4	1,20
	automatische Übertragung	0,2	5	1,00	4	0,80	5	1,00
	Übertragungsgeschwindigkeit	0,1	1	0,10	2	0,20	2	0,20
	erzeugte Kosten	0,25	4	1,00	4	1,00	3	0,75
	Umsetzungsaufwand	0,15	4	0,60	3	0,45	3	0,45
	Gesamtnutzwert:			**3,90**		**3,25**		**3,60**

Tabelle 4: Nutzwertanalyse des SPS-Bausteins

Der Lösungsvorschlag 1 hat mit einem Gesamtnutzwert von 3,90 hat den größten Nutzen und wird dadurch ausgewählt und umgesetzt.

4.2.2 Auswahl des Gateways

Das Gateway verbindet das lokale Netzwerk mit dem Internet. Nachfolgend werden mehrere einsetzbare Gateways aufgezeigt und methodisch anhand der Bauteileeigenschaften und den Einstellungsmöglichkeiten erarbeitet, welches Gateway verwendet werden soll. Die, zur Auswahl stehenden, Gateways sind zum einen das IIoT-Gateway der Firma HMS Industrial Networks GmbH, der EWON Flexy 205 [10] und zum anderen das Gateway der Firma Weidmüller Interface GmbH & Co. KG, das IOT-GW30 [20]. Als dritte Option wird ein Computer mit dem Betriebssystem Linux betrachtet, der als Gateway aufgesetzt wird. Die verschiedenen Gateways werden nachfolgend genauer beschrieben, die Gewichtungsfaktoren erläutert und danach die Auswahl des Gateways mit einer Nutzwertanalyse methodisch ermittelt. Zuerst werden jedoch die Bewertungskriterien beschrieben.

Für die Auswahl eines Gateways sind Bewertungskriterien von Nöten anhand dessen ersichtlich wird, welches Gateway am geeignetsten ist. Ein essenzielles Kriterium ist die Sicherheit, die das Gateway bietet und damit der Schutz vor ungewolltem Fremdzugriff. Die wirtschaftlichen Kosten für die Anschaffung werden ebenfalls verglichen. Ebenso sind der benötigte Bauraum und der damit verbundene Arbeits- und Installationsaufwand relevant. Das Gateway sollte darüber hinaus Funktionen zur Fernwartung, dem Zugriff auf speicherprogrammierbare Steuerungen und den Aufbau einer MQTT-Verbindung vorweisen. Ebenfalls sollte die Möglichkeit bestehen das Gateway individuell durch Programmierung anzupassen. Nachfolgend werden die verschiedenen Gateways beschrieben und deren Eigenschaften und Funktionen aufgezeigt.

Der EWON Flexy 205 der Firma HMS Industrial Networks GmbH wird hauptsächlich von der Firma TIG Automation GmbH zur Fernwartung eingesetzt, bietet jedoch auch die Möglichkeit als Gateway verwendet zu werden. Der EWON benötigt durch seine Größe wenig Bauraum. Da der EWON bereits in der Anlage zur Fernwartung verbaut ist, erzeugt dieser keine neuen Kosten und keinen weiteren Arbeitsaufwand. Das Gateway besitzt Kommunikationsstandards für alle SPS-Modelle von Siemens und die Möglichkeit einen MQTT-Client in der eigenen BASIC-IDE zu programmieren. Der EWON besitzt durch die Verwendung als Fernwartungsrouter über mehrere Authentifizierungsebenen, die eine Sicherheit vor Fremdzugriff bietet. Ebenfalls kann ein Schreibschutz der, von der SPS gelesenen, Daten eingestellt werden.

Das Gateway IOT-GW30 der Firma Weidmüller Interface GmbH & Co. KG ist vom Bauraum kleiner als der EWON Flexy 205. Die Kosten für die Anschaffung befinden sich im mittleren Segment und die Sicherheit der Prozessdaten und des Zugriffes auf den Anlagenprozess ist durch den einstellbaren Schreibschutz und die Auslegung der MQTT-Verbindung genauso gut wie die Sicherheit des EWON. Durch die ähnliche Bauform des Gateways wie der EWON ist der Arbeitsaufwand vom Umfang ebenfalls ähnlich. Die Zugriffsmöglichkeiten des IOT-GW30 auf die SPS erfordert mehrere Schritte und ist sehr komplex, was durch die beschränkte Programmierbarkeit nicht erleichtert wird. Ebenfalls bietet das Gateway keine Möglichkeit zur Fernwartung der Anlage, sodass für die Fernwartung ein anderes Gerät eingesetzt werden müsste.

Der selbst aufgesetzte und installierte Computer mit dem Betriebssystem Linux bietet die meisten Möglichkeiten im Bereich der Sicherheit, da alle denkbaren Sicherheitsoptionen verwirklicht werden können, jedoch sorgt dies zu einem sehr hohen Arbeitsaufwand. Außerdem ist der Computer an sich ein Sicherheitsrisiko, da es mehr Möglichkeiten gibt auf diesen zuzugreifen. Auf den Computer wird das Betriebssystem Linux installiert, da dieses vom Benutzer am individuellsten angepasst werden kann. Der

benötigte Bauraum ist ebenfalls groß, da ein komplett funktionsfähiger Computer aufgebaut werden muss. Durch die hohe Programmierbarkeit kann eine Verbindung über MQTT und der Zugriff auf die SPS realisiert werden. Da der Computer in das Netzwerk integriert werden muss, kann über installierbare Software auch die Fernwartung ermöglicht werden. Die beiden größten Nachteile, die der Computer mit dem Betriebssystem Linux mit sich bringt, sind die hohen Anschaffungskosten und der entstehende Arbeitsaufwand beim Aufsetzen, wie auch bei der Wartung und Aktualisierung des Systems.

Um die Gewichtungsfaktoren der einzelnen Bewertungskriterien ermitteln zu können, müssen diese nach Wichtigkeit eingestuft werden. Hierfür werden die Bewertungskriterien in drei Ebenen eingeteilt. Das wichtigste Kriterium und damit auf der obersten Ebene ist die Sicherheit. Es darf nicht die Möglichkeit bestehen den Anlagenprozess zu beeinflussen oder auf das System zuzugreifen, um Prozessdaten unerlaubt zu entnehmen. Auf der zweiten Ebene sind die Kosten, die erzeugt werden und die SPS-Zugriffsmöglichkeit, um die Daten von der vorhandenen Siemens SPS abgreifen zu können. Die dritte Ebene enthält die Kriterien des Bauraums und des Arbeitsaufwandes genauso wie die Fähigkeit zur Anbindung mit MQTT, der Programmierbarkeit und des Fernwartungszugriffs. Da die Gewichtungsfaktoren in Summe immer 1 ergeben müssen, wird als Anhaltspunkt zur Auslegung der Gewichtungsfaktoren ermittelt, welche Gewichtung ohne Einstufung der Wichtigkeit entsteht. Hierzu wird die Summe der Gewichtungsfaktoren durch die Anzahl der Kriterien genommen. Hierbei entsteht ein Gewichtungsfaktor von 0,125. Die Bewertungskriterien werden in drei Ebenen eingeteilt, die die Wichtigkeit der Kriterien wiedergibt, um ordentlich gewichten zu können. Für die erste und wichtigste Ebene wird ein Faktor von 0,2 und für die zweite Ebene ein Faktor von 0,15 eingetragen. Somit bleibt für die restlichen fünf Kriterien auf der Ebene drei eine Restsumme von 0,5. Daraus folgt das die dritte Ebene einen Gewichtungsfaktor von jeweils 0,1 erhält. Nachfolgend sind die drei Lösungsansätze in der Tabelle der Nutzwertanalyse bewertet (siehe Tabelle 5). [19]

	Bewertung von 0 - 5 (0 schlecht, 5 gut)	Gewichtungs-faktor	EWON Flexy 205		IOT-GW30		PC als Gateway	
			Bewertung	Teilnutzwert	Bewertung	Teilnutzwert	Bewertung	Teilnutzwert
Bewertungskriterien	Bauraum	0,1	3	0,30	4	0,40	2	0,20
	Sicherheit	0,2	3	0,60	3	0,60	5	1,00
	Kosten	0,15	5	0,75	3	0,45	1	0,15
	MQTT-Fähigkeit	0,1	4	0,40	4	0,40	4	0,40
	Arbeitsaufwand	0,1	3	0,30	3	0,30	1	0,10
	SPS-Zugriffsmöglichkeiten	0,15	4	0,60	3	0,45	4	0,60
	Programmierbarkeit	0,1	3	0,30	2	0,20	5	0,50
	Fernwartungsmöglichkeit	0,1	5	0,50	0	0,00	4	0,40
	Gesamtnutzwert:			**3,75**		**2,80**		**3,35**

Tabelle 5: Nutzwertanalyse verschiedener Gateways

Da der Gesamtnutzwert des EWON Flexy 205 der Firma HMS Industrial Networks GmbH am höchsten ist, sollte dieser verwendet werden.

4.2.3 Übertragung der Prozessdaten von SPS zur Cloud

Für die Übertragung der Prozessdaten von der SPS zur Cloud müssen zwei Übertragungen erzeugt werden. Die erste Übertragung findet von der SPS zum Gateway statt. Die zweite Übertragung leitet die Prozessdaten vom Gateway an die Schnittstelle der Cloud-Struktur weiter. Die Auswahl der Teilkomponenten zur Übertragung der Prozessdaten von der SPS zur Cloud wird mit einem Morphologischen Kasten durchgeführt. Die Konzepte werden anschließend mit den Anforderungen aus der Anforderungsliste verglichen und das finale Teillösungskonzept ausgewählt. Im Morphologischen Kasten wird ebenfalls die Schnittstelle zur Cloud definiert. Nachfolgend ist der Morphologische Kasten mit den unterschiedlichen farblich markierten Lösungskonzepten zu erkennen (siehe Abbildung 14) [18].

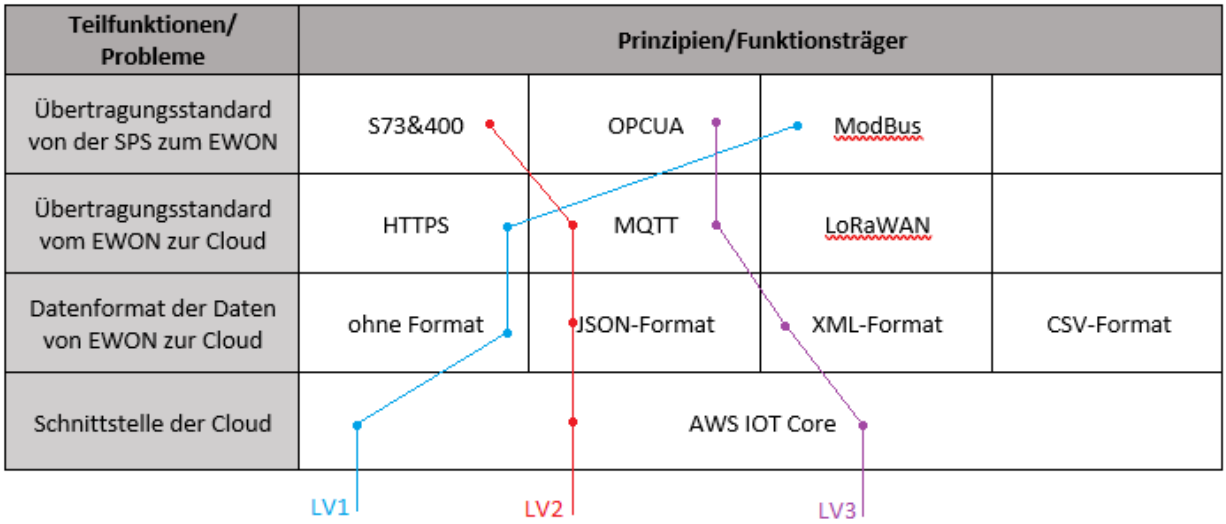


Abbildung 14: Morphologischer Kasten der Übertragung von SPS zu Cloud

Im morphologischen Kasten sind drei Lösungsvorschläge (LV1-3) eingetragen, die willkürlich erstellt werden. Die Übertragungsstandards für die Übertragung der Prozessdaten von der SPS zum EWON können alle über Ethernet angebunden werden, jedoch muss für OPCUA und ModBus auf der SPS ein Server eröffnet werden was zu einem erhöhten Arbeitsaufwand führt. Der Übertragungsstandard S73&400 ist der Übertragungsstandard von Siemens und erfordert keine weitere Arbeit. Durch die Schnittstelle der Cloud, den AWS IOT Core von Amazon, sind die drei Übertragungsstandards „Hyper Text Transfer Protocol Secure“ (HTTPS), MQTT und „Long Range Wide Area Network“ (LoRaWAN) möglich. Der AWS IOT Core wird verwendet, da bei der Planung der Cloud-Struktur als Einschränkung nur die Cloud-Angebote von Amazon AWS betrachtet werden. Dies wird im Punkt 4.2.4 genauer erläutert. Der Übertragungsstandard HTTPS wird für das Übertragen von Methoden an Server verwendet, um die übertragenen Datenpunkte einzulagern oder Daten abzugreifen. Die Übergabe einer Methode kann bei der Anbindung an den AWS IOT Core nicht verwendet werden, da der AWS IOT Core diese nicht ausführen kann. Der MQTT-Übertragungsstandard verbindet verschiedene Clients über einen Broker. MQTT wird häufig für IIoT-Lösungen eingesetzt und besitzt die Möglichkeiten, das Empfangen von Daten im Client zu deaktivieren, sodass dieser nicht angesprochen werden kann. Ebenfalls ist das Anlegen von Sicherheitszertifikaten möglich, damit eine gesicherte Verbindung aufgebaut wird. Der LoRaWAN-Standard nutzt die Daten auf der Bitübertragungsschicht, um diese dem Broker zu übergeben. Um den Übertragungsstandard LoRaWAN im EWON verwenden zu können, muss eine Datenübertragung in der IDE des EWON neu entwickelt werden. Die zu sendende Nachricht, die im Funktionsrumpf des Übertragungsstandards übermittelt werden soll, kann formatiert übergeben werden. Ein Datenformat ist eine Syntax wie die Daten in der Nachricht übermittelt werden sollen, um diese nach dem Übertragen wieder in die Bestandteile zerlegen zu können. Aufgrund der obigen Ausführungen wird der Lösungsvorschlag 2

ausgewählt, da bei der Umsetzung mit MQTT und S73&400 die meisten Sicherheiten bei geringstem Arbeitsaufwand zu erreichen sind. Ebenfalls ist durch den Aufbau der Nachricht im JSON-Format die Nachricht durch markante Punkte iterierbar.

4.2.4 Planung der Cloud-Struktur

Bei der Planung der Cloud-Struktur werden als Vereinfachung nur Cloud-Lösungen von Amazon betrachtet, um die Vielfalt der vorhandenen Angebote eingrenzen zu können. Im vorherigen Punkt 4.2.3 ist festgelegt worden, dass der AWS IOT Core als Schnittstelle zum EWON eingesetzt wird. Ebenfalls wird die Schnittstelle zu Qlik Sense betrachtet und eine zu verwendende Datenbank ausgewählt. Da Qlik Sense viele vorgefertigte Verbindungsoptionen besitzt, können relationale und nicht relationale Datenbanken verwendet werden. Der Begriff relational bedeutet „in Verbindung stehend“ und wird in Bezug auf Datenbanken für die Datenhaltung verwendet. Eine relationale Datenbank hat eine Tabellenstruktur, in der die Daten einsortiert werden müssen, was einen hohen Speicherbedarf erzeugt. Nicht relationale Datenbanken hingegen benötigen weniger Speicher, da diese die Nachrichten im Ganzen als Strings speichern. Um die, in der nicht relationalen Datenbank, eingelagerten Nachrichten für Analysen verwenden zu können, müssen diese bei der Entnahme aus der Datenbank in die Prozessdaten aufgespalten werden. Somit wird beim Entnehmen der Daten ein hoher Rechenaufwand benötigt. Eine relationale Datenbank von AWS ist zum Beispiel AWS Aurora, die sowohl als MySQL als auch als PostgreSQL Datenbank aufgesetzt werden kann. Nicht relationale Datenbanken von AWS sind beispielsweise die AWS DynamoDB, die die Nachrichten in einem DataLake ablegt oder AWS S3, bei der die Nachrichten in eine Ordnerstruktur einsortiert werden. Damit die vom AWS IOT Core über MQTT erhaltenen Prozessdaten in der Datenbank abgelegt werden können, muss die Nachricht von AWS Lambda an die Datenbank übergeben und gegebenenfalls verarbeitet werden. Eine Auswahl der Komponenten mit einer Methode wird nicht durchgeführt, da es nur für die Datenbank verschiedene Auswahlmöglichkeiten gibt.

Für die Umsetzung der Cloud-Struktur wird eine relationale Datenbank von AWS ausgewählt, da Qlik Sense die Prozessdaten in Tabellenform am einfachsten und effizientesten verarbeiten kann. Hierfür wird die AWS Datenbank Aurora im MySQL-Standard ausgewählt. Durch die Auswahl der relationalen Datenbank AWS Aurora, muss im AWS Lambda nicht nur die Übertragung der Prozessdaten umgesetzt werden, sondern auch die Verarbeitung der Nachricht und der Umgang mit SQL-Befehlen. Ebenfalls muss die Verbindung von dem MQTT-Broker, dem AWS IOT Core, zu AWS Lambda hergestellt werden. In der nachfolgenden Abbildung ist der Aufbau der Cloud-Struktur und die Funktionen grafisch dargestellt (siehe Abbildung 15).

Abbildung 15: Aufbau der Cloud-Struktur

4.2.5 Übertragung der Prozessdaten von der Cloud zu Qlik Sense

Für die Übertragung der Prozessdaten von der SQL-Datenbank AWS Aurora zu Qlik Sense wird ein vorhandener MySQL-Connector verwendet. Sobald Qlik Sense mit der Datenbank verbunden ist, können die Datenpunkte mit SQL-Befehle abgegriffen und zur Weiterverarbeitung bei Qlik Sense vorgehalten werden.

4.2.6 Planung der Analysen zur vorbeugenden Instandhaltung der Fassreinigungsanlage

Um die Analysen für die vorbeugende Instandhaltung planen zu können, wird das Instandhaltungsverfahren festgelegt. Es wird eine Kombination, aus der periodisch vorbeugenden und der zustandsabhängigen Instandhaltung verwirklicht. Bei der periodisch vorbeugenden Instandhaltung werden Intervalle eingesetzt, in denen das Bauteil gewartet oder ausgetauscht werden soll. Die Intervalle können zeit- oder ereignisgesteuert gesetzt werden. Bei diesem Konzept wird die maximale Betriebsdauer des zu instandhaltenden Bauteils aus dem Datenblatt entnommen und als zeitlicher Schwellwert für den Austausch des Bauteils verwendet. Bei der zustandsabhängigen Komponente der Instandhaltungsstrategie wird die Änderung des Zustands des Bauteils betrachtet, der Abnutzungsvorrat ermittelt und festgelegt ab wann das Bauteil auszutauschen ist.

Die in der Fassreinigungsanlage verbaute Aktorik, bei denen die Zustände ermittelbar sind, sind Kugelhähne, Magnetventile und Kükenhähne, da diese Endlagen besitzen. Durch die Endlagen kann die beim Auf- und Zufahren benötigte Schaltzeit der Bauteile ermittelt werden und über den Verlauf der Schaltzeiten eine Aussage über den Abnutzungsvorrat getroffen werden. Sobald der Abnutzungsvorrat größtenteils aufgebraucht oder der zeitliche Schwellwert erreicht ist, soll eine Meldung zum Warten oder Austauschen des Bauteils per E-Mail an den zuständigen Mitarbeiter versendet werden, damit dieser die Wartung für den nächsten Anlagenstillstand planen kann.

Nach der Verbindung der Datenbank mit Qlik Sense, werden die Prozessdaten aus der Datenbank entnommen und in Qlik Sense abgelegt. Die zwischengespeicherten Daten werden gefiltert und verarbeitet, damit anschließend die geplanten Analysen umgesetzt und durchgeführt werden.

4.3 Ergebnis der Konzeptentwicklung

In diesem Abschnitt werden die Teilkonzepte zusammengefasst und das Gesamtkonzept erläutert. Die benötigten Schnittstellen und Übertragungsstandards der Teilkonzepte und die Analysen zur vorbeugenden Instandhaltung für die Fassreinigungsanlage sind definiert.

Zur Darstellung der Prozessdaten auf den HMI-Panels, werden die Prozessdaten umfangsoptimiert mit dem Übertragungsstandard SIMATIC S7-1200, S7-1500 Channel an das SCADA-System übergeben. Dabei wird durch Zurückschreiben der Daten geprüft, ob die abgegriffenen Daten korrekt sind. Die Länge des auszulesenden DB wird mit einem ATTR_DB, einem in der SPS vorhandenen Funktionsbaustein, ermittelt und der zu behandelnde Datentyp als Eingabeparameter an den Funktionsbaustein übergeben. Der in der Anforderungsliste als Wunsch definierte EWON Flexy 205 wird als Gateway verwendet, um die Prozessdaten an die Cloud zu übergeben. Der EWON wird durch den Übertragungsstandard S73&400 mit der SPS verbunden und im JSON-Format mit MQTT an den AWS IOT Core gesendet. Der AWS IOT Core erhält die Nachricht und gibt diese an AWS Lambda zur Verarbeitung weiter. AWS Lambda trennt die Nachricht im JSON-Format auf und speichert die Daten in der MySQL-Datenbank AWS Aurora. Qlik Sense greift mit dem MySQL-Connector auf die Datenbank AWS Aurora zu und entnimmt die Prozessdaten von der Datenbank. In Qlik Sense werden die Prozessdaten für die Analysen vorbereitet und die geplanten Analysen zur vorbeugenden Instandhaltung umgesetzt.

5. Entwurf und Realisierung der Komponenten

In diesem Abschnitt werden die entwickelten Konzepte entworfen und realisiert. Der Aufbau der Cloud-Struktur wird erläutert und aufgezeigt welche Schritte zur Erstellung notwendig sind. Danach werden die benötigten Programme auf den verschiedenen Ebenen methodisch mit der Modellierungssprache UML entworfen und in den verschiedenen Programmiersprachen realisiert. Im Anschluss an die Erstellung der Programme werden die Analysen zur vorbeugenden Instandhaltung in Qlik Sense umgesetzt.

5.1 Aufbau der Cloud-Struktur

Für die Cloud-Struktur, bestehend aus dem AWS IOT Core, der Programmierumgebung AWS Lambda und der Datenbank AWS Aurora, müssen die einzelnen Komponenten erstellt werden. Die Prozessdaten werden mittels des EWON Flexy 205 per Übertragungsstandard MQTT an den AWS IOT Core gesendet, der als MQTT-Broker fungiert. Der AWS IOT Core sendet die erhaltenen Daten an AWS Lambda weiter, wo eine Verbindung zur AWS Aurora Datenbank aufgebaut wird und die Daten in der Datenbank abgelegt werden. Die Übergaben der Prozessdaten innerhalb der Cloud-Struktur werden im Unterpunkt 5.2.4 erläutert.

Um den MQTT-Broker im AWS IOT Core verwenden zu können, muss ein Objekt erzeugt werden. Dieses Objekt stellt den EWON dar, von dem die MQTT-Daten gesendet werden. Beim Erstellen des Objekts werden automatisch eine Amazon Ressource Number (ARN) sowie drei Zertifikate erstellt. Die ARN wird für die Verbindung der AWS-Cloud-Teilnehmer untereinander als Endpunkt verwendet und die Zertifikate werden benötigt, um die Daten an den IOT Core senden zu dürfen. Die Zertifikate bestehen aus drei Dateien: Einem Zugriffszertifikat, einem Authoritätszertifikat und einem Schlüsselzertifikat. Diese drei Zertifikate müssen auf dem EWON platziert und beim Verbindungsaufbau aufgerufen werden. Ebenfalls muss für den Fremdzugriff auf den IOT Core eine Richtlinie erstellt werden, in der festgelegt wird, welche Aktionen das Objekt, das heißt der EWON, ausführen darf. Die Richtlinie erlaubt dem Objekt nur das Veröffentlichen und nicht das Abonnieren. So ist es möglich Daten vom EWON an den IOT Core zu senden, jedoch nicht vom IOT Core zum EWON, um eine Anlagengefährdung auszuschließen. Die Verbindung und Weitergabe der Prozessdaten zu AWS Lambda, wird im Unterpunkt 5.2.4 erläutert.

In AWS Lambda, der Programmierumgebung zur Verarbeitung und Weitergabe der Prozessdaten, muss eine Funktion erstellt werden, mit der die, vom AWS IOT Core übergebenen Daten, verarbeitet und an die Datenbank AWS Aurora übergeben wird. Die Programmierung zur Übergabe der Prozessdaten wird im Unterpunkt 5.2.4 genauer erläutert.

Die Datenbank AWS Aurora wird als serverlose MySQL-Datenbank aufgesetzt. Die serverlose Datenbank von Aurora ist selbstskalierend, das heißt das bei der Erstellung der Datenbank eine minimale und maximale Anzahl an Kapazitätseinheiten eingestellt werden muss. AWS Aurora schaltet bei hohem Bearbeitungsaufwand automatisch die Kapazitätseinheiten hinzu und meldet diese auch wieder ab. Ebenfalls muss bei der Erstellung ein Benutzer mit Passwort angelegt werden, um eine Authentifizierung bei Zugriff durchzuführen. Die Erstellung der Datenbank als serverlose Option hat keine Zugangspunkte von außen und ist damit vor Fremdeingriff sicher. Um Zugang von außen auf die Datenbank zu erhalten, muss ein virtueller Computer, der sich in der gleichen Virtual Private Cloud (VPC) befindet, erstellt werden. Nachdem die Datenbank und der virtuelle Computer errichtet sind, wird in der Datenbank mit einem SQL-Editor eine Tabelle für die Datenhaltung aufgebaut. Die vier Spalten, die in der Tabelle vordefiniert werden müssen, sind für den Namen, den Wert und die Zeit die vom EWON übertragen werden. Die vierte Spalte ist für den Primärschlüssel, den eine SQL-Datenbank besitzen muss. Der Primärschlüssel muss pro Zeile der Tabelle eindeutig sein, um keine Probleme beim Auslesen der Prozessdaten zu verursachen. Der Primärschlüssel wird als solcher bei der Erstellung der Datenbank definiert und

festgelegt, dass dieser nicht Null sein darf. Ebenfalls wird definiert, dass der Primärschlüssel automatisch inkrementiert werden soll.

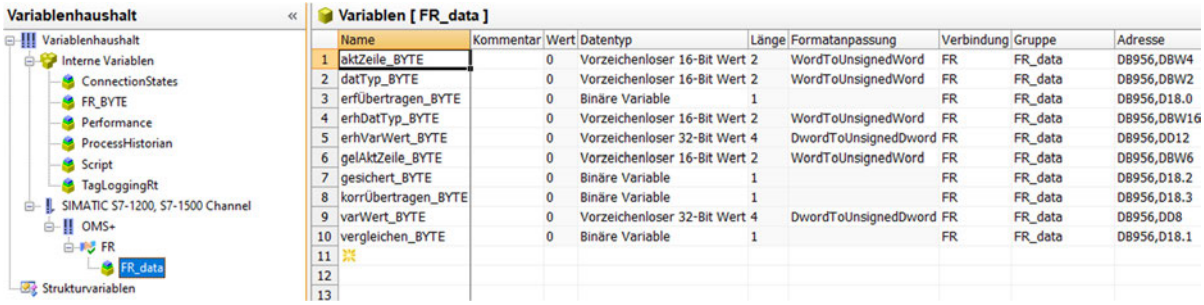
5.2 Programmieren der Übertragungen

In den nachfolgenden Unterpunkten werden die Übertragungen der Prozessdaten dargestellt und in der jeweiligen Programmiersprache umgesetzt. Hierzu gehört sowohl die Erstellung des SPS-Bausteins zur Übertragung an das SCADA-System als auch die Übergabe der Prozessdaten und die Anbindung des EWON Flexy 205. Ebenfalls werden die Übertragungen der Daten vom EWON zur Cloud, in der Cloud und von der Cloud zu Qlik Sense umgesetzt.

5.2.1 Übertragung der Daten von SPS zu SCADA

Die datenumfangsoptimierte Übertragung der Prozessdaten von der SPS zum SCADA-System zur Darstellung und Haltung der Daten auf dem lokalen Server wird mit einem SPS-Baustein umgesetzt. Da das SCADA-System die Daten von der SPS-Ebene abgreift, sind ebenfalls Programmbestandteile auf dem SCADA-System zu erzeugen. Der SPS-Baustein wartet auf das Zurückschreiben der Daten vom SCADA-System, um die Übertragung und deren Richtigkeit sicherzustellen.

Für die Übertragung muss im SCADA-System die Verbindung zur SPS erstellt und der Übertragungsstandard festgelegt werden. Ebenfalls muss eine Tabelle erzeugt werden, die die gleiche Anzahl an zu übertragenden Daten eines Datentyps enthält. Für jeden Datentyp, der übertragen werden soll, muss eine Tabelle angelegt werden. Sowohl die interne Tabelle als auch die Verbindung zur SPS werden im Variablenhaushalt des WINCC-Explorers angelegt. Um die Verbindung mit der SPS im SCADA-System zu erzeugen, wird ein „SIMATIC S7 1200, S7 1500 Channel“ erzeugt und mit der IP-Adresse der SPS versorgt. Die SPS muss sich im gleichen lokalen Netzwerk wie das SCADA-System befinden, um eine Verbindung herstellen zu können. In der Verbindung wird eine Tabelle erzeugt, in der alle Variablen erstellt werden müssen, die das SCADA-System von der SPS abgreifen soll. Jede erstellte Variable muss mit der SPS-Adresse verbunden werden. Die Abbildung 16 zeigt die aufgebaute Verbindung zur SPS und beispielhaft die abzugreifenden Daten für den Datentyp Byte.



	Name	Kommentar	Wert	Datentyp	Länge	Formatanpassung	Verbindung	Gruppe	Adresse
1	aktZeile_BYTE		0	Vorzeichenloser 16-Bit Wert	2	WordToUnsignedWord	FR	FR_data	DB956,DBW4
2	datTyp_BYTE		0	Vorzeichenloser 16-Bit Wert	2	WordToUnsignedWord	FR	FR_data	DB956,DBW2
3	erfÜbertragen_BYTE		0	Binäre Variable	1		FR	FR_data	DB956,D18.0
4	erhDatTyp_BYTE		0	Vorzeichenloser 16-Bit Wert	2	WordToUnsignedWord	FR	FR_data	DB956,DBW16
5	erhVarWert_BYTE		0	Vorzeichenloser 32-Bit Wert	4	DwordToUnsignedDword	FR	FR_data	DB956,DD12
6	gelAktZeile_BYTE		0	Vorzeichenloser 16-Bit Wert	2	WordToUnsignedWord	FR	FR_data	DB956,DBW6
7	gesichert_BYTE		0	Binäre Variable	1		FR	FR_data	DB956,D18.2
8	korrÜbertragen_BYTE		0	Binäre Variable	1		FR	FR_data	DB956,D18.3
9	varWert_BYTE		0	Vorzeichenloser 32-Bit Wert	4	DwordToUnsignedDword	FR	FR_data	DB956,DD8
10	vergleichen_BYTE		0	Binäre Variable	1		FR	FR_data	DB956,D18.1
11									
12									
13									

Abbildung 16: Variablenhaushalt von WINCC

Die Funktion der Programme wird anhand eines Flussdiagramms dargestellt und erläutert. Das Flussdiagramm ist eine Darstellungsmethode zur Veranschaulichung des Arbeitsablaufes, wobei ein Pfad von oben nach unten einem Durchlauf im Programm entspricht. Diese Programme werden auf der SPS- und auf der SCADA-Ebene zyklisch ausgeführt. Nachfolgend werden drei Flussdiagramme für die einzelnen

Programmbestandteile eingefügt und erläutert. Als erstes ist das Flussdiagramm des SPS-Bausteins dargestellt (siehe Abbildung 17) [21].

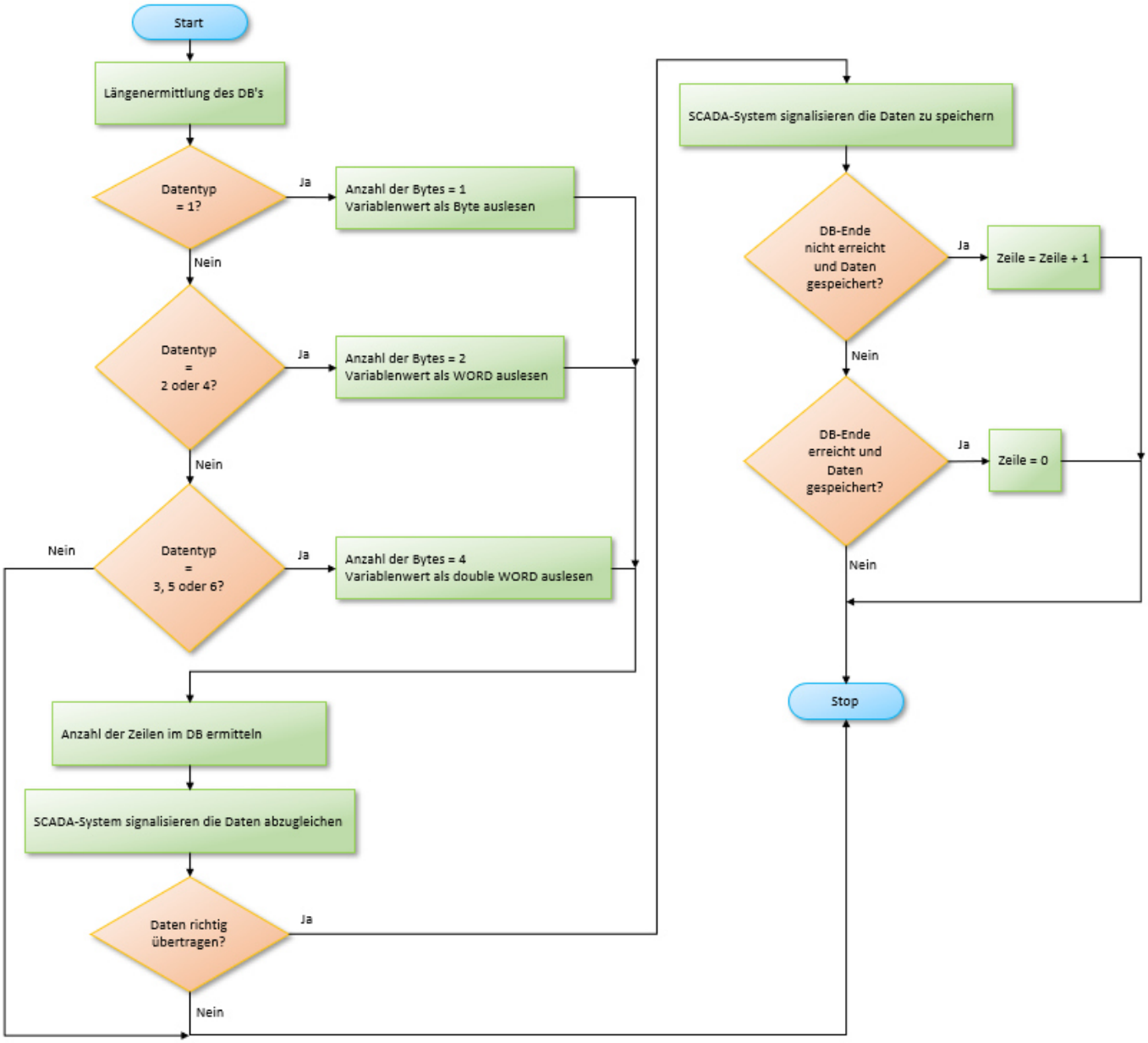


Abbildung 17: Flussdiagramm SPS-Baustein

Für die Programmierung des SPS-Bausteins wird ein Funktionsbaustein erstellt. Ein Funktionsbaustein hat gegenüber einer einfachen Funktion einen eigenen Datenbaustein zur Speicherung von Daten. Beim Aufruf des Funktionsbausteins im Hauptprogramm der SPS, dem OB1, wird eine Datenbausteininstanz erzeugt. Die Datenbausteininstanz hält nur die Daten für diesen einen Aufruf des Funktionsbausteins vor. Als Eingangswerte des Funktionsbausteins wird ein Platzhalter für den verwendeten Datentyp und die Nummer des auszulesende Datenbaustein übergeben. Für jeden Datentyp wird ein Integer-Wert

vergeben, der an den Funktionsbaustein als Eingangswert übergeben werden muss. Die Tabelle 6 zeigt die Zuordnung der Datentypen zum Integer-Wert.

Datentyp	Byte	Integer	double Integer	Word	double Word	Real
Integer-Wert	1	2	3	4	5	6

Tabelle 6: Zuordnung der Datentypen zum Integer-Wert

Zu Beginn des Programms wird die Länge des zu übertragenden Datenbausteins mit dem ATTR_DB, einem Funktionsbaustein zum Auslesen von Datenbausteinattributen, ausgelesen. Der erhaltene Rückgabewert gibt wieder, wie viele Byte der Datenbaustein an Speicherplatz benötigt. Um über den Datenbaustein iterieren zu können, wird die Bytegröße des übergebenen Datentyps ermittelt, der Iterationsparameter erstellt und der zu übertragende Variablenwert ausgelesen. Die Anzahl der Zeilen im Datenbaustein wird ermittelt und das Signal zum Speichern der Daten auf der SCADA-Ebene ausgelöst. Sobald die abgegriffenen Daten vom SCADA-System an das SPS-Programm zurückgeschrieben sind, werden die erhaltenen Daten auf Richtigkeit geprüft. Nach Feststellung der Richtigkeit wird das Speichern der Daten auf dem SCADA-System erlaubt. Das SCADA-System gibt das erfolgreiche Speichern an das SPS-Programm zurück und setzt die Zeile auf den nächsten Datenpunkt oder beim Erreichen des letzten Datenpunkts im Datenbaustein, wird die Zeile auf 0 zurückgesetzt. Das Programm wird in den SPS-Programmiersprachen Funktionsplan (FUP) und strukturierter Text (ST) umgesetzt. Die auskommentierte Programmierung des SPS-Bausteins ist im Anhang A hinterlegt.

Die zu erzeugenden Programmbestandteile auf der SCADA-Ebene werden in der Programmiersprache C umgesetzt. Es wird ein Programm zum Prüfen der abgegriffenen Daten und ein Programm zu Speicherung der Daten entworfen. Nachfolgend werden die Flussdiagramme zum Prüfen und zum Speichern der Daten eingefügt und erläutert (siehe Abbildungen 18 und 19) [21].

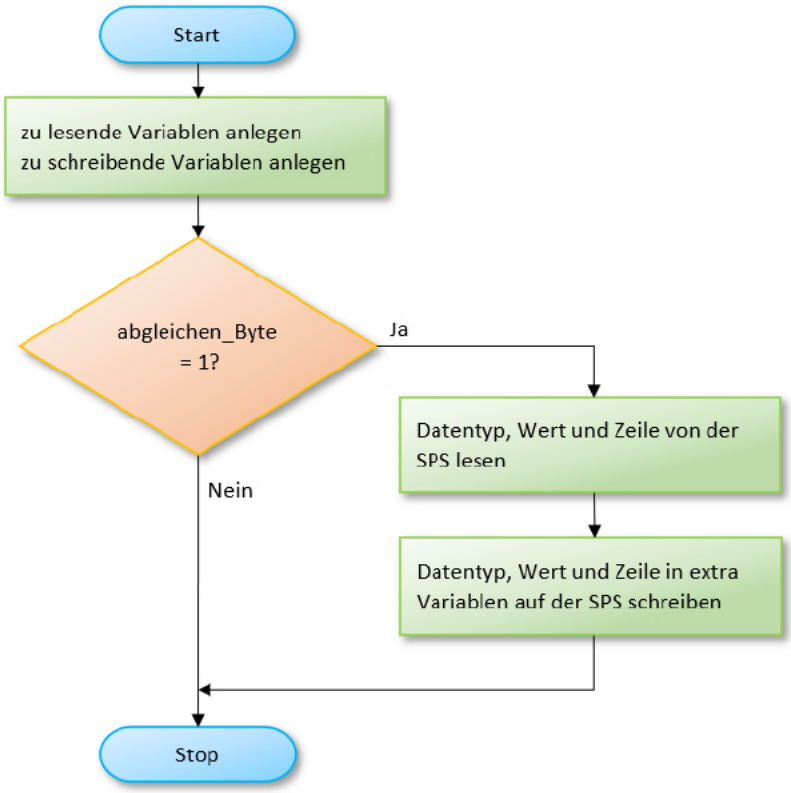


Abbildung 18: Flussdiagramm der Prüfung im SCADA-System

Das Programm zum Prüfen der korrekten Übertragung der Prozessdaten wird durch die ansteigende Flanke des Auslösers ausgeführt. Nachdem das Programm gestartet wurde, werden alle benötigten Variablen definiert und die Wenn-Abfrage ausgelöst. Für jeden Datentyp muss eine Wenn-Abfrage erzeugt werden, die durch die verschiedenen Auslöser freigeschaltet wird. In der Abfrage wird der Prozessdatenwert, die aktuelle Zeile und der behandelte Datentyp von der SPS abgegriffen und in eine andere variable auf der SPS zurückgeschrieben. Der kommentierte Quelltext für den Datentyp Byte ist im Anhang A eingefügt.

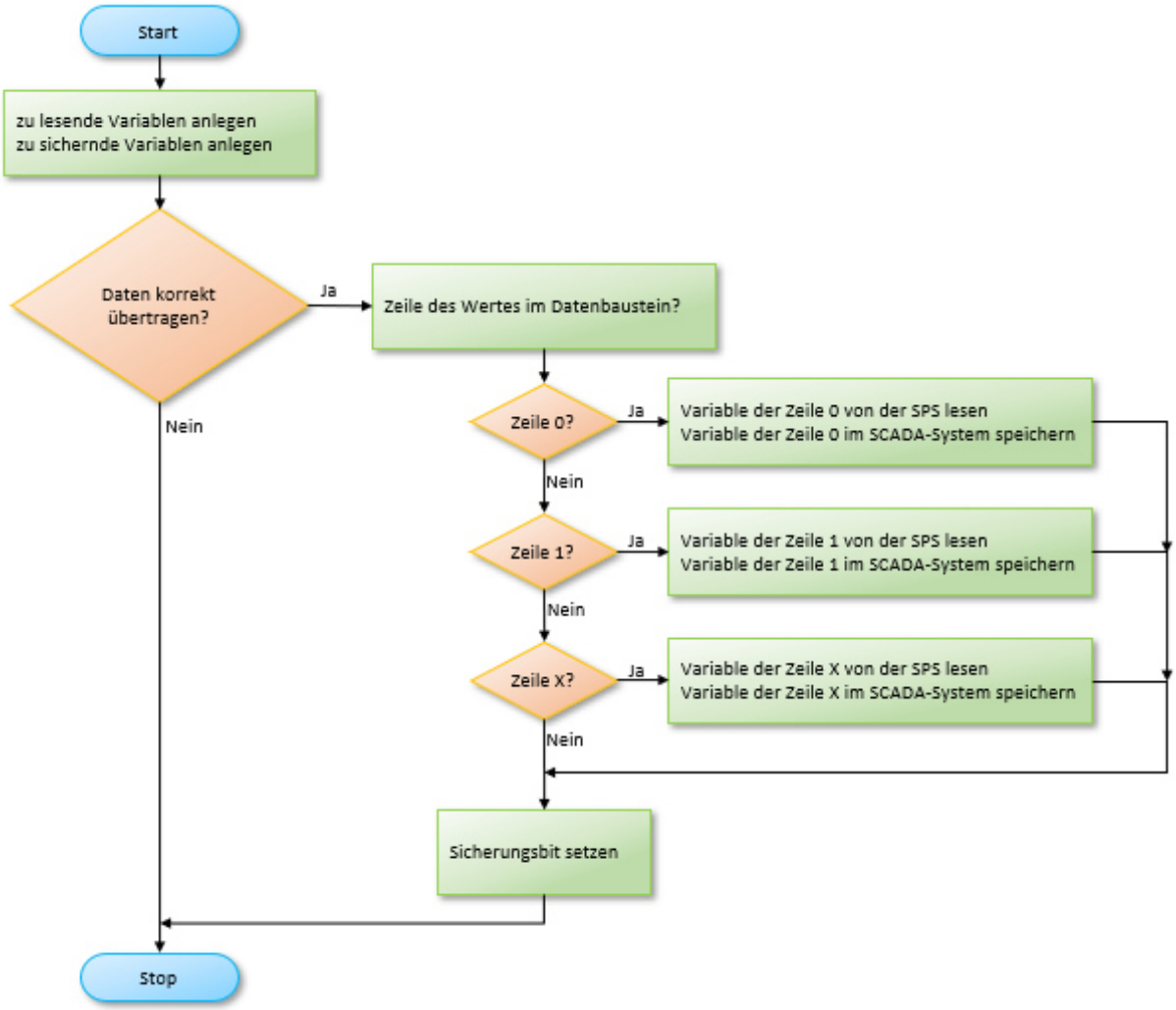


Abbildung 19: Flussdiagramm der Speicherung im SCADA-System

Die Speicherung der Daten wird mit der steigenden Flanke des Auslösers ausgeführt und die verwendeten Variablen definiert. Für jeden übertragenen Datentyp wird eine Switch-Case-Abfrage erzeugt, die anhand der Zeile im Datenbaustein die Prozessdaten zuordnet und diese in dem internen Variablenhaushalt des SCADA-Systems abspeichert. Die Switch-Case-Abfrage darf nur dann ausgelöst werden, wenn vorher die korrekte Übertragung der Daten erkannt wird. Zuletzt wird auf der SPS das Sicherungsbit auf 1 gesetzt, damit der nächste Datenpunkt behandelt werden kann und die aktuelle Zeile weiter oder zurückgesetzt wird. Der kommentierte Quelltext für den Datentyp Byte befindet sich im Anhang A.

5.2.2 Übertragung der Daten von SPS zu EWON

Die Übertragung der Prozessdaten von SPS zu EWON Flexy 205, wird mit einer S73&400-Verbindung erstellt, die der EWON bereitstellt. Hierzu muss ein I/O-Server, so benannt im EWON, aktiviert werden und die IP-Adresse der SPS eingegeben werden. Vor der IP-Adresse wird der Übertragungsstandard von ProfiBus durch die Eingabe von ISOTCP festgelegt. Da es sich um eine SPS der Klasse S7-1500 handelt, auf die zugegriffen werden soll, muss der klassenspezifische Standard im ProfiBus festgelegt werden. Die nachfolgende Abbildung 20 zeigt den aktivierten I/O-Server mit dem Aufruf der Geräteadresse, den zu benutzenden Standards und der Abfragerate der Prozessdaten.

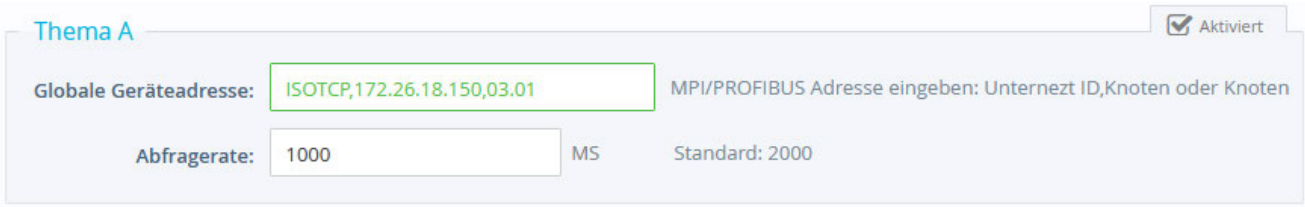


Abbildung 20: Aufbau der Verbindung von SPS zu EWON

Um die Prozessdaten zur Übermittlung mit MQTT an die Cloud vorhalten zu können, werden die zu übertragenden Daten mit einer Abfragerate von 1000ms von der SPS abgegriffen und im EWON als Wert dargestellt. Hierzu muss für jeden einzelnen zu übertragenden Wert eine Variable im EWON angelegt und die spezifische Adresse in der SPS eingetragen werden. Bei der Erstellung einer solchen Variable, wird im Identifikationsbereich ein Name für die Variable vergeben und der Schreibschutz aktiviert. Die Variable wird wie die entnommene Variable in der SPS benannt, um eine Rückführung und eine Eindeutigkeit der übertragenen Daten zu erhalten. Nachfolgend ist eine Abbildung zur Erstellung und der Konfiguration der Variable eingefügt (siehe Abbildung 21).

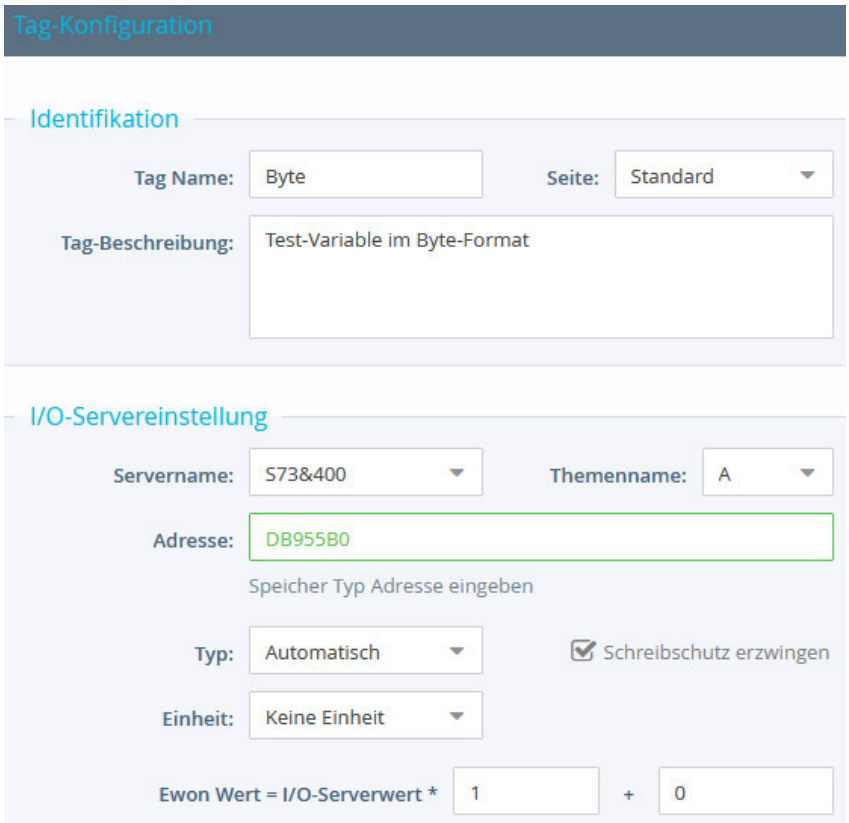


Abbildung 21: Variablenkonfiguration im EWON

Es wird ein Name für die Variable vergeben und eine Beschreibung angefügt. Hier wird eine Testvariable Byte erzeugt, die über den vorher aktivierten und konfigurierten I/O-Server die SPS-Variable ausliest. Der in Grün eingerahmte Bereich gibt die spezifische SPS-Adresse wieder, die über ProfiBus ausgelesen wird. Wichtig ist das Erzwingen des Schreibschutzes, damit die Prozessdaten vom EWON nicht ansteuerbar, sondern nur lesbar sind. Wenn der Datentyp oder die Einheit der SPS-Variable bekannt sind, können diese Informationen ebenfalls eingetragen werden. Der von der SPS erhaltene Wert kann durch Multiplikation oder mit dem Beaufschlagen eines Offsets verändert werden. Nachfolgend werden beispielhaft Variablen erstellt und mit der SPS verknüpft, um die Datenhaltung im EWON aufzuzeigen (siehe Abbildung 22).

Name	I/O-Server	I/O-Adresse	Wert	Tag-Beschreibung
Byte	S73&400	DB955B0	10	Testvariable für den Datentyp Byte
A_Status_SPS	S73&400	Status	1	Testvariable für den Status der SPS
Int	S73&400	DB955S2	-755	Testvariable für den Datentyp Integer
DInt	S73&400	DB955L4	32000	Testvariable für den Datentyp double Integer
Word	S73&400	DB955W8	65450	Testvariable für den Datentyp Word
DWord	S73&400	DB955D10	4026531855	Testvariable für den Datentyp double Word
Real	S73&400	DB955F14	-3.154325	Testvariable für den Datentyp Real

Abbildung 22: Tabelle der Daten im EWON

Im EWON werden die abgegriffenen Daten in Tabellenform dargestellt und alle notwendigen Informationen gezeigt. Ebenfalls werden der erhaltene Wert und die SPS-Adresse der Variablen dargestellt. Durch das, beim Wert befindliche, Schloss-Symbol wird der Schreibschutz der Daten visualisiert. Die im EWON angelegten Daten können mit einem Befehl der BASIC-IDE ausgelesen und danach versendet werden.

5.2.3 Übertragung der Daten von EWON zu Cloud

Die Übertragung der Prozessdaten vom EWON zum AWS IOT Core, wird in der BASIC IDE auf dem EWON Flexy 205 programmiert. Ebenfalls wird in dieser IDE der MQTT-Client aufgebaut, die Zertifikate aufgerufen und die Prozessdaten im JavaScript Object Notation Format, auch JSON-Format, übertragen. Das JSON-Format ist ein standardisiertes Datenformat für die Codierung von Daten und wird in einem späteren Abschnitt genauer erläutert. Die Zertifikate für den Zugriff des EWON auf die Cloud-Struktur müssen mit einem Programm mit File Transfer Protocol (FTP) auf dem EWON abgelegt werden. Hierfür wird auf dem EWON im Dateipfad „/usr“ ein neuer Ordner namens „AWSCertificates“ erzeugt und die in Punkt 5.1 beschriebenen Zertifikate darin abgelegt.

Die BASIC IDE vom EWON basiert auf der Syntax der Programmiersprache „Beginner’s All-purpose Symbolic Instruction Code“ (BASIC), wurde jedoch für die Anwendungsfälle des EWON erweitert. Die Programmiersprache BASIC ist eine symbolische Allzweck-Programmiersprache für Anfänger. Der in der BASIC IDE erstellte Programmcode wird von oben nach unten durchlaufen und die ermittelten auszuführenden Aufgaben in eine Warteschlange eingetragen. Die Warteschlange wird nacheinander nach dem First-In-First-Out-Prinzip abgearbeitet. Die Besonderheit der BASIC IDE ist es das Label erzeugt werden können, zu denen das Programm bei Aufruf springen kann.

Die Prozessdaten werden im JSON-Format übertragen. Das JSON-Format ist ein vom Menschen und Maschinen lesbares Format, bei dem die Prozessdaten durch Einklammerung und Trennung durch weitere Satzzeichen ein vordefiniertes und damit verarbeitbares Gerüst besitzen. Die komplette Nachricht wird in eckige Klammern gesetzt, um einen definierten Start und ein definiertes Ende zu haben. Die Prozessdaten, die einer Zeile entsprechen sind in geschweiften Klammern benannt und der Wert eingesetzt. Die Benennung wird von dem dazugehörigen Wert mit einem Doppelpunkt und die verschiedenen Namen-Wert-Paare mit Kommata voneinander getrennt. Nachfolgend ist ein Beispiel für eine Nachricht im JSON-Format gezeigt.

[{tag:Signal_1,value:25.21,time:13:03:55 18.11.21},{tag:Signal_2,value:13,time:08:53:12 18.11.21}]

Die Grundfunktionen des Programms auf dem EWON sind der Aufbau der MQTT-Verbindung, das Auslesen des internen Speichers, das Verpacken der Prozessdaten im JSON-Format und das Senden der Daten. Aufgrund der besonderen Sicherheitsanforderungen an das Projekt, wird beim Aufruf der MQTT-Verbindung die Abonnierungsfunktion nicht aktiviert und damit dem EWON die Möglichkeit genommen von anderen Teilnehmern Daten zu erhalten. Im Flussdiagramm ist die Funktion des Programms dargestellt und im Nachhinein textlich erläutert (siehe Abbildung 23) [21].

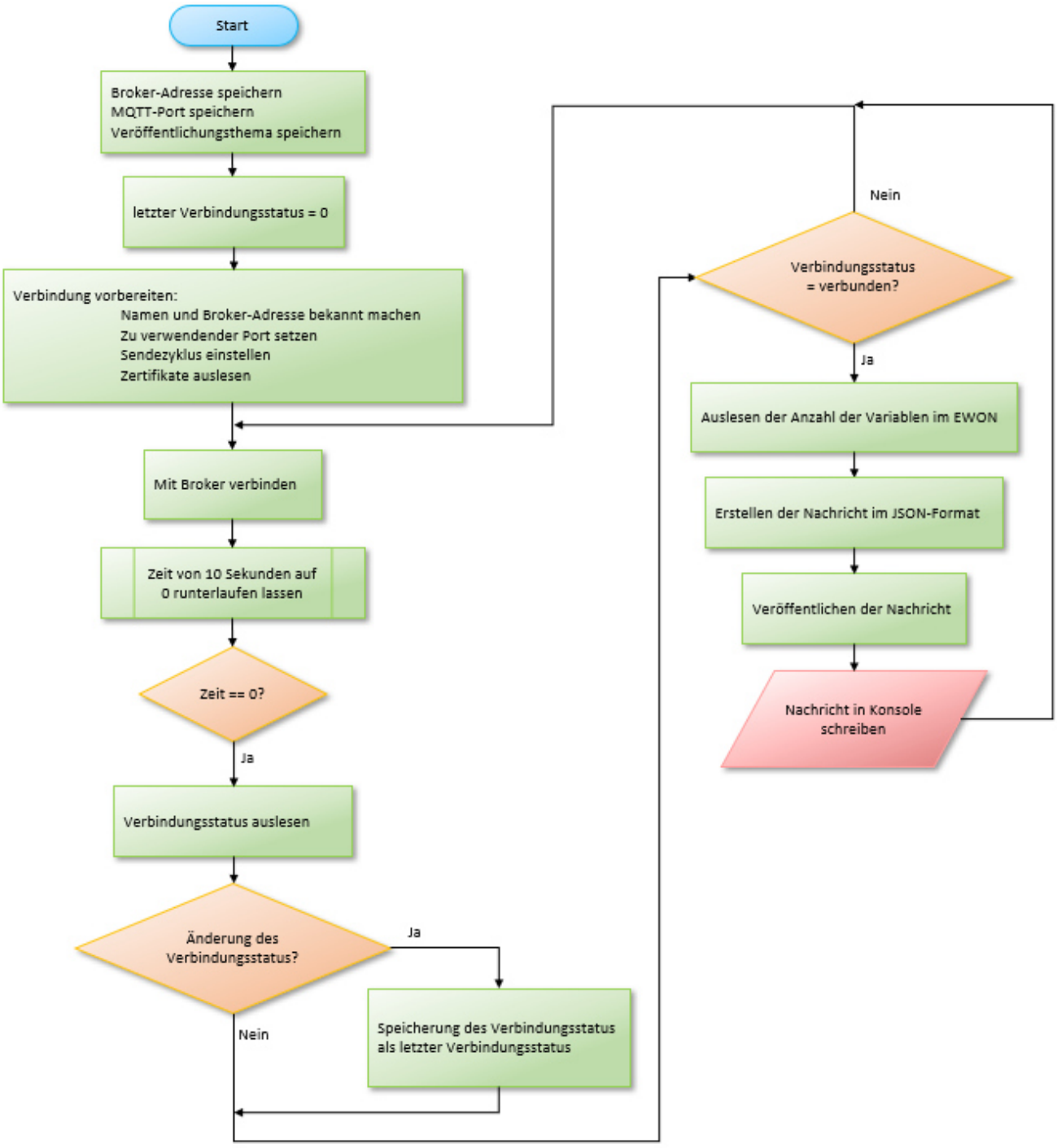


Abbildung 23: Flussdiagramm des EWON-Programms

Nach dem Start des Programms wird die Konsole gelöscht und drei String-Variablen erzeugt. Der erste String enthält die Adresse des MQTT-Brokers und der zweite String den zu verwendenden MQTT-Port. Als drittes wird das Thema gespeichert, zu dem mit MQTT veröffentlicht werden soll. Ebenfalls wird eine Integer-Variable erstellt, um den Verbindungsstatus mit dem MQTT-Broker wiederzugeben. Nachfolgend wird die MQTT-Verbindung eingestellt. Hierfür muss der Kanal definiert werden, indem der spezifische Name des EWON und die Adresse des MQTT-Brokers eingestellt werden. Es ist wichtig das der vergebene Name des EWON identisch mit dem Namen des im AWS IOT Core angelegten Objektes ist. Die für die erfolgreiche Verbindung benötigten Parameter wie der zu verwendende MQTT-Port und die

Sendefrequenz werden ebenfalls gesetzt. Die auf dem EWON abgelegten Zertifikatdateien werden aufgerufen und die Verbindung zum Broker aufgebaut. Bei erfolgreich aufgebauter Verbindung zum Broker werden die, auf dem EWON befindlichen, Prozessdaten zeilenweise in einer Schleife ausgelesen und im JSON-Format nacheinander in eine String-Variable geschrieben. Nach dem Auslesen der Prozessdaten und dem Aneinanderreihen im JSON-Format, wird die Nachricht auf dem offenen Kanal zu dem definierten Thema veröffentlicht. Dies geschieht in der eingestellten Sendefrequenz zyklisch. Der erzeugte und kommentierte Quelltext ist im Anhang A eingefügt.

5.2.4 Übertragung der Daten in der Cloud

Die Übertragung in der Cloud behandelt die Übergabe der vom AWS IOT Core erhaltenen Prozessdaten an AWS Lambda und das Einordnen dieser Daten von AWS Lambda in die Datenbank AWS Aurora. Für die Übergabe der Prozessdaten vom IOT Core zu Lambda, wird in der Oberfläche des IOT Cores eine Regel erstellt. Diese Regel besagt das, wenn zu dem Thema „/Fassreinigung“ eine Nachricht veröffentlicht wird, diese direkt an die Lambda-Funktion weitergegeben wird. AWS baut mit Angabe des Funktionsnamen die Verbindung zu AWS Lambda automatisch auf und löst das erstellte Programm aus. Es wird kein weiterer Auslöser zum Ausführen des Programms benötigt, da das Programm immer beim Erhalt einer Nachricht zum definierten Thema ausgelöst wird. In der Abbildung 24 ist die erstellte Regel abgebildet.

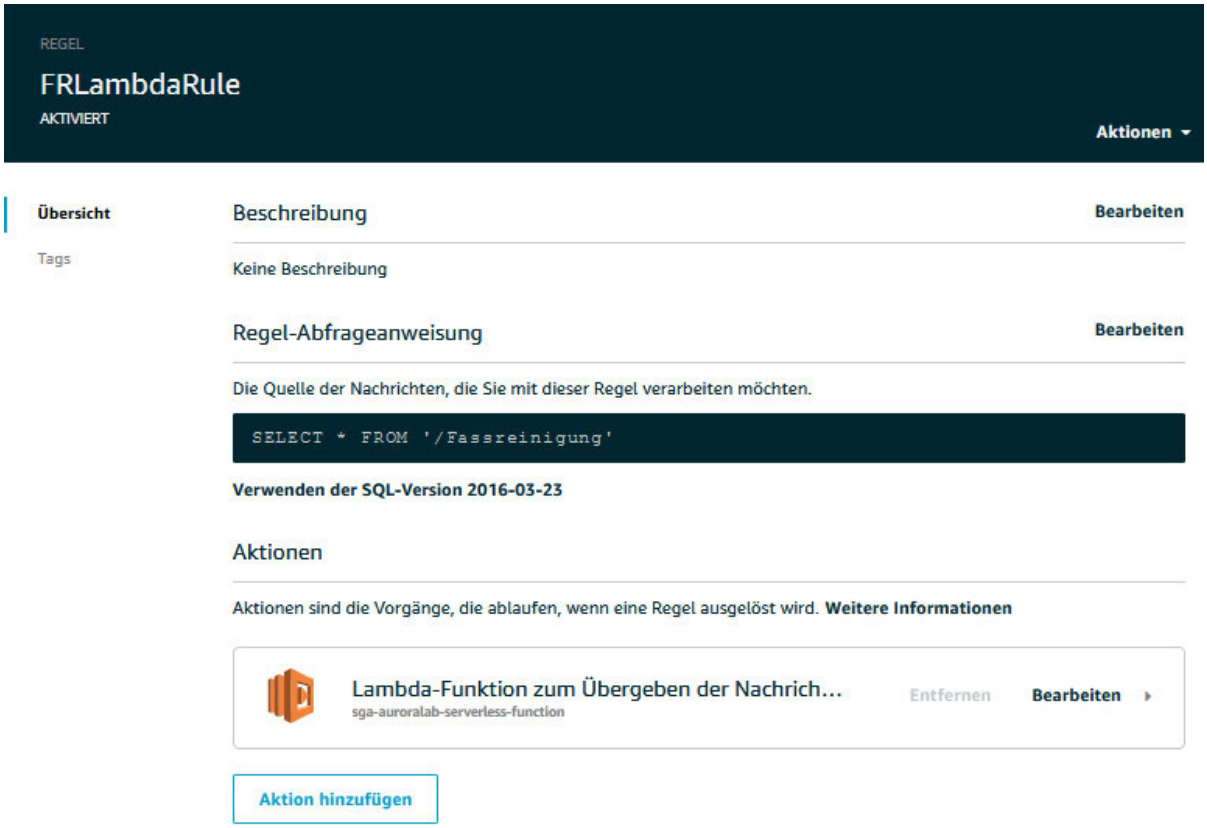


Abbildung 24: Regel zum Auslösen des AWS Lambda Scripts

Die Programmierung der Lambda-Funktion zum Einlagern der Prozessdaten in der Datenbank AWS Aurora wird in der Programmiersprache JavaScript geschrieben. Das Programm soll beim Ausführen die im JSON-Format erhaltene Nachricht zeilenweise in die Datenbank schreiben. Hierfür wird jede Zeile einzeln aus der Nachricht entnommen und abgespeichert bis das Ende der Nachricht erkannt wird. Für jede

Zeile wird ein SQL-Befehl neu erzeugt und ausgeführt. Nachfolgend wird das Programm in einem Flussdiagramm dargestellt und erläutert (siehe Abbildung 25) [21].

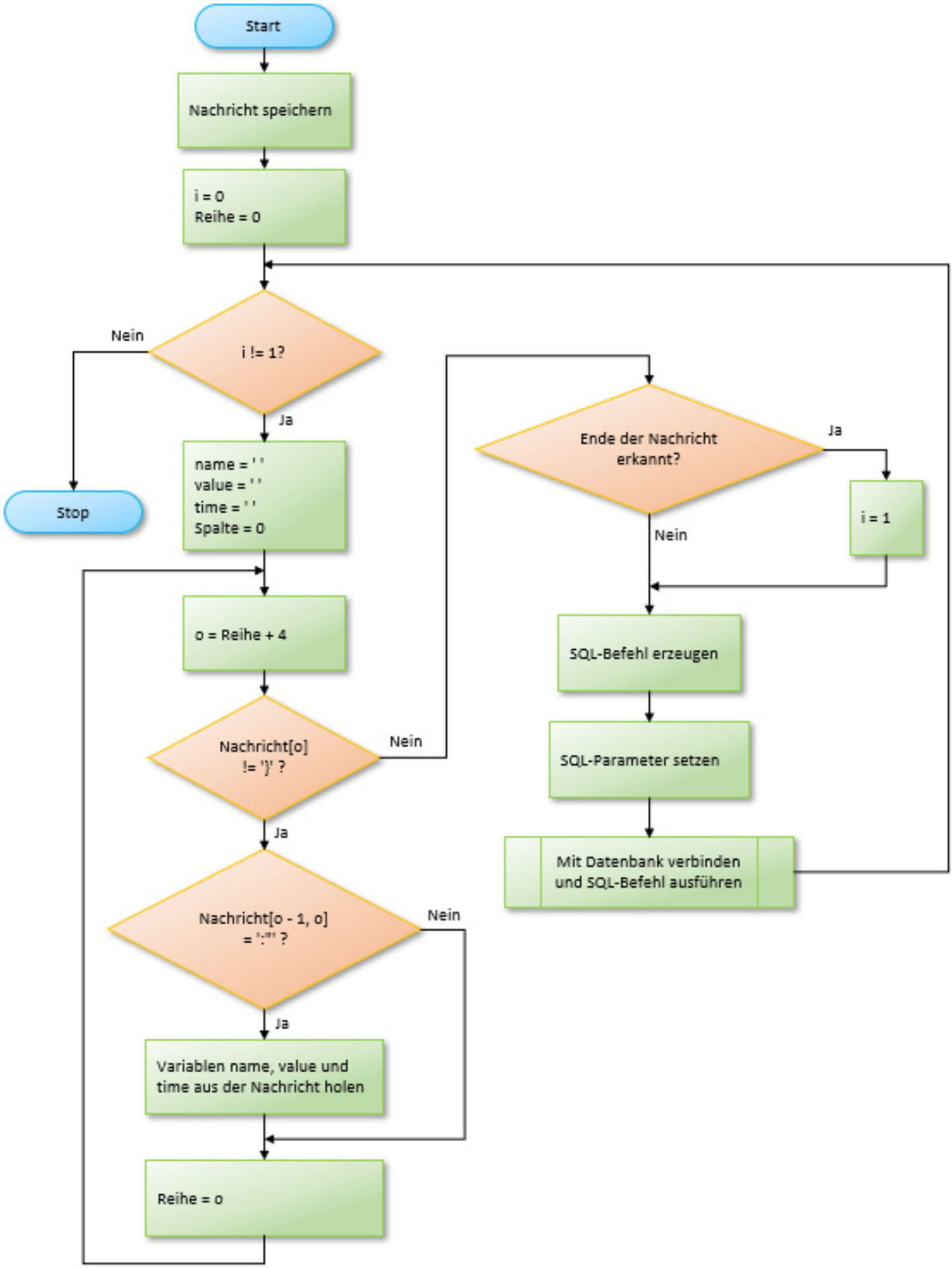


Abbildung 25: Flussdiagramm des Lambda-Programms

Das Programm wird nach dem Erhalt einer Nachricht automatisch ausgeführt und die erhaltene Nachricht in einer String-Variable gespeichert. In einer While-Schleife, die nach vollständigem Abarbeiten der Nachricht beendet wird, wird mit einer For-Schleife über die Nachricht iteriert und anhand von markanten Stellen in der Nachricht die übertragenen Werte ausgelesen. In der Nachricht sind pro Zeile der Datentyp, der Datenwert und die Zeit zu entnehmen und in der SQL-Datenbank abzulegen. Nachdem die Daten einer Zeile entnommen sind, wird ein SQL-Befehl erzeugt, der die Prozessdaten in die Datenbank eingeben soll. Bei der anschließenden Definierung des Zugangspunktes der Datenbank, wird der auszuführende SQL-Befehl gespeichert und im Nachhinein ausgeführt. Im Anhang A befindet sich der auskommentierte Quelltext.

5.2.5 Übertragung der Daten von der Cloud zu Qlik Sense

Die Prozessdaten der Fassreinigungsanlage, die in der AWS Datenbank Aurora liegen, sollen von der Analyseplattform Qlik Sense direkt aus der Datenbank entnommen werden. Für die Verbindung wird der MySQL-Connector von Qlik Sense verwendet. Um Qlik Sense über den MySQL-Connector mit der Datenbank zu verbinden, wird der Datenendpunkt, der verwendete Port und der Datenbankname benötigt. Ebenfalls werden Benutzerdaten und Passwort, sowie Zertifikate abgefragt. Die für die Verbindung benötigten Daten müssen bei AWS Aurora in der Benutzeroberfläche ermittelt und bei dem Qlik Sense Connector eingetragen werden. Der verwendete Datenbankname und der Port sind einfach ermittelbar, genauso wie der Benutzername und das Passwort. Bei der Erstellung der Datenbank AWS Aurora sind keine Zertifikate anlegbar, wodurch keine Zertifikate bei Qlik Sense zu hinterlegen sind. Ebenfalls ist die Ermittlung des Datenendpunktes sehr komplex. Es werden die Datenendpunkte der Datenbank, der virtual private Cloud (VPC), zu der die Datenbank gehört, und der VPC-Gruppe, zu der die VPC gehört, verwendet. Keiner dieser Endpunkte kann verwendet werden, um die Datenbank mit Qlik Sense zu verbinden. Damit ist die Datenbank AWS Aurora mit dem MySQL-Connector von Qlik Sense nicht verbindbar. Dies ist ein Abbruchskriterium für die Arbeit, da die für die Analysen benötigten Prozessdaten nicht übermittelbar sind.

5.3 Erstellen von Analysen zur vorbeugenden Instandhaltung auf Qlik Sense

Die geplanten Analysen zur vorbeugenden Instandhaltung können, durch die nicht realisierbare Verbindung zwischen Qlik Sense und der Datenbank, nicht umgesetzt werden.

6. Integrationstests

Die Integrationstest werden für die umgesetzten Teilkonzepte und für die Übermittlung der Prozessdaten von der SPS bis zu der Datenbank AWS Aurora durchgeführt. Es wird erläutert welche Tests durchgeführt werden und die Ergebnisse der Tests bewertet. Nachfolgend ist die Liste der Tests mit dem Ergebnis des Tests eingefügt (siehe Tabelle 7).

Tests	nicht OK	OK
Datenübertragung von SPS zu SCADA-System		X
Datenübertragung von SPS zu EWON		X
Datenübertragung vom EWON zum AWS IOT Core		X
Datenübertragung in der Cloud		X
Automatische Übertragung der Daten von der SPS in die Datenbank		X

Tabelle 7: Liste und Ergebnisse der Tests

6.1 Integrationstests der Teilkonzepte

Die Funktionen der in Punkt 5.2 erzeugten Übertragungen werden einzeln getestet und die Ergebnisse bewertet. Hierfür werden die Funktionen erläutert und ein passender Test entwickelt. Nach der Durchführung des Tests, wird das Ergebnis bewertet und festgestellt, ob die Funktion hinreichend erfüllt ist.

6.1.1 Datenübertragung von der SPS zum SCADA-System

Der Test der Datenübertragung von der SPS zum SCADA-System wird durchgeführt, um eine Aussage über die Richtigkeit der übertragenen Daten zu erhalten. Ebenfalls wird die benötigte Übertragungszeit ermittelt. Damit die Übertragungszeit gemessen werden kann, wird eine Zeitmessung für die Übertragung von zehn Datenpunkten erstellt und die durchschnittliche Übertragungszeit errechnet. Während der Zeitmessung werden die zu sendenden Datenpunkte mit den im SCADA-System gespeicherten Datenpunkten verglichen und die Übertragung bewertet. In der nachfolgenden Tabelle sind die gesendeten und empfangenen Datenpunkte eingetragen und miteinander verglichen (siehe Tabelle 8).

Anzahl der Übertragungen	Name	gesendeter Wert	erhaltener Wert
1	Signal_1	12	12
2	Signal_2	22	22
3	Signal_3	32	32
4	Signal_4	42	42
5	Signal_1	52	52
6	Signal_2	62	62
7	Signal_3	72	72
8	Signal_4	82	82
9	Signal_1	92	92
10	Signal_2	102	102

Tabelle 8: Vergleich der gesendeten und empfangenen Datenpunkte im SCADA-System

Die auf dem SCADA-System gespeicherten Daten sind identisch mit denen die in der SPS eingestellt sind. Die Funktion der sicheren Übertragung der Prozessdaten ist erfüllt. Die benötigte Zeit der Übertragung von zehn Datenpunkten beträgt 69,8 Sekunden. Die durchschnittliche Übertragungszeit eines Datenpunktes beträgt 6,98 Sekunden.

6.1.2 Datenübertragung von der SPS zum EWON Flexy 205

Die Datenübertragung von der SPS zum EWON Flexy 205 ist mit dem Übertragungsstandard S73&400 realisiert. Die Prozessdaten werden zyklisch im Sekundentakt durch den EWON von der SPS abgegriffen. Um Fehler bei der Übertragung auszuschließen, werden die Daten gegenübergestellt und verglichen. Hierfür werden auf der SPS Variablen mit unterschiedlichen Datentypen erstellt und im EWON angelegt. Es gibt sechs relevante Datentypen, für die eine Variable angelegt wird. Es werden fünf Datenpunkte für jede Variable an den EWON übertragen und anschließend die Daten tabellarisch dargestellt und verglichen (siehe Tabelle 9).

		Variablen					
		Byte	Int	Dint	Word	DWord	Real
Wertgruppe 1	SPS	12	32	16	26	0	3,14156
	EWON	12	32	16	26	0	3,14156
Wertgruppe 2	SPS	2	252	-22001	99562	23	-0,4
	EWON	2	252	-22001	99562	23	-0,4
Wertgruppe 3	SPS	15	-3000	99	888	56	6,2345
	EWON	15	-3000	99	888	56	6,2345
Wertgruppe 4	SPS	6	0	5	425	99999	1,23456
	EWON	6	0	5	425	99999	1,23456
Wertgruppe 5	SPS	10	-755	32000	65450	651612	-3,154325
	EWON	10	-755	32000	65450	651612	-3,154325

Tabelle 9: Vergleich der Daten von der SPS mit denen des EWON

Die mit dem Übertragungsstandard S73&400 übertragenen Daten auf dem EWON sind identisch mit den auf der SPS eingetragenen Daten. Die Funktion der Teilkomponente ist hinreichend erfüllt und die Korrektheit der Prozessdaten festgestellt.

6.1.3 Datenübertragung vom EWON Flexy 205 zum AWS IOT Core

Die, auf dem EWON Flexy 205, vorgehaltenen Daten werden im JSON-Format aneinandergereiht und über eine MQTT-Verbindung an den AWS IOT Core gesendet. Um die beiden Funktionen der Teilkomponente ausreichend testen zu können, werden die vorgehaltenen Daten mit der ausgehenden Nachricht im JSON-Format verglichen. Die, zu sendenden, sechs Testvariablen sind die in Punkt 6.1.2 verwendeten Variablen, um die Übertragung jedes Datentyps zu testen. Ebenfalls wird die ausgehende Nachricht des EWON mit der im AWS IOT Core erhaltenen Nachricht abgeglichen, um gegebenenfalls Änderungen festzustellen. Die, von dem Programm des EWON erstellte, Nachricht wird in der Konsole ausgegeben. Die im EWON vorgehaltenen Daten, werden in der Tabelle 10 aufgezeigt.

Variablenname	Byte	Int	Dint	Word	DWord	Real
Wert	10	-755	32000	65450	651612	-3,154325

Tabelle 10: Daten im EWON

Die in der Tabelle dargestellten Daten, werden durch das Programm im EWON im JSON-Format aneinandergereiht. Die dabei entstehende und mit MQTT versendete Nachricht ist nachfolgend gezeigt.

[{tag:Byte,value:10,time:17:26:59 23.11.21},{tag:Int,value:-755,time:17:26:59 23.11.21},
{tag:DInt,value:32000,time:17:26:59 23.11.21},{tag:Word,value:65450,time:17:26:59 23.11.21},
{tag:DWord,value:651612,time:17:26:59 23.11.21},{tag:Real,value:-3.154325,time:17:26:59 23.11.21}]

Die durch den AWS IOT Core erhaltene Nachricht, wird mit der im AWS IOT Core vorhandenen Testoberfläche dargestellt. Hierfür wird die Testoberfläche als MQTT-Client verbunden und das Veröffentlichungsthema des EWON abonniert. Die in der Testoberfläche erhaltene Nachricht ist nachfolgend dargestellt.

[{tag:Byte,value:10,time:17:26:59 23.11.21},{tag:Int,value:-755,time:17:26:59 23.11.21},
{tag:DInt,value:32000,time:17:26:59 23.11.21},{tag:Word,value:65450,time:17:26:59 23.11.21},
{tag:DWord,value:651612,time:17:26:59 23.11.21},{tag:Real,value:-3.154325,time:17:26:59 23.11.21}]

Um die Ursprungsdaten mit den Nachrichten vergleichen zu können, werden die Daten aus den beiden Nachrichten entnommen und in der nachfolgenden Tabelle 11 eingetragen. Für die Übersichtlichkeit werden dabei die Zeiten vernachlässigt.

Daten im EWON	Variablenname	Byte	Int	Dint	Word	DWord	Real
	Wert	10	-755	32000	65450	651612	-3,154325
versendete Nachricht	Variablenname	Byte	Int	Dint	Word	DWord	Real
	Wert	10	-755	32000	65450	651612	-3,154325
erhaltene Nachricht	Variablenname	Byte	Int	Dint	Word	DWord	Real
	Wert	10	-755	32000	65450	651612	-3,154325

Tabelle 11: Vergleich der per MQTT versendeten Daten

Der Vergleich der Daten zeigt, dass die in den Nachrichten enthaltenen Daten den Ursprungsdaten entsprechen. Es sind keine Fehler beim Erstellen der Nachricht im JSON-Format entstanden und die vom AWS IOT Core erhaltene Nachricht ist ebenfalls fehlerfrei. Die Funktionen der Teilkomponente sind somit hinreichend erfüllt.

6.1.4 Datenübertragung in der Cloud

Nach dem Erhalt der Prozessdaten im AWS IOT Core, wird die Nachricht automatisch an AWS Lambda weitergegeben. In AWS Lambda wird die erhaltene Nachricht in die einzelnen Variablen zerlegt und in AWS Aurora tabellarisch eingelagert. Die zu testenden Funktionen der Datenübertragung in der Cloud sind die automatische Weitergabe der Daten vom AWS IOT Core an AWS Lambda, das Entnehmen der Daten aus der Nachricht und das korrekte Einlagern dieser in der Datenbank. Das automatische Auslösen des Lambda-Scripts kann mit den gegebenen Darstellungs- und Testwerkzeugen nicht festgestellt werden. Die Zerlegung der Nachricht kann durch das automatische Auslösen des Programms in AWS Lambda ebenfalls nicht festgestellt werden, da beim automatischen Ausführen keine Konsolenausgaben erstellt werden. Eine Aussage über das automatische Auslösen des Programms und die Zerlegung der Nachricht kann nur über die Betrachtung der in AWS Aurora eingelagerten Prozessdaten und deren Vollständigkeit getroffen werden. Um einen aussagekräftigen Test zu erhalten, werden fünf Nachrichten vom EWON an den AWS IOT Core gesendet und in AWS Aurora eingelagert. Die in der SQL-Datenbank

eingelagerten Daten werden nachfolgend tabellarisch mit den Nachrichten verglichen (siehe Tabellen 12 und 13).

Wertgruppen	Werte	Nachricht 1	Nachricht 2	Nachricht 3	Nachricht 4	Nachricht 5
1	Name	Byte	Byte	Byte	Byte	Byte
	Wert	10	3	7	12	0
	Zeit	16:03:13 22.11.21	16:04:13 22.11.21	16:05:13 22.11.21	16:06:13 22.11.21	16:07:13 22.11.21
2	Name	Int	Int	Int	Int	Int
	Wert	38	22	-1	255	3
	Zeit	16:03:13 22.11.21	16:04:13 22.11.21	16:05:13 22.11.21	16:06:13 22.11.21	16:07:13 22.11.21
3	Name	DInt	DInt	DInt	DInt	DInt
	Wert	0	-22	22222	-365	3
	Zeit	16:03:13 22.11.21	16:04:13 22.11.21	16:05:13 22.11.21	16:06:13 22.11.21	16:07:13 22.11.21
4	Name	Word	Word	Word	Word	Word
	Wert	1	2	4	8	16
	Zeit	16:03:13 22.11.21	16:04:13 22.11.21	16:05:13 22.11.21	16:06:13 22.11.21	16:07:13 22.11.21
5	Name	DWord	DWord	DWord	DWord	DWord
	Wert	32	64	128	256	512
	Zeit	16:03:13 22.11.21	16:04:13 22.11.21	16:05:13 22.11.21	16:06:13 22.11.21	16:07:13 22.11.21
6	Name	Real	Real	Real	Real	Real
	Wert	3,21	2,65	-4,6	8	16,55
	Zeit	16:03:13 22.11.21	16:04:13 22.11.21	16:05:13 22.11.21	16:06:13 22.11.21	16:07:13 22.11.21

Tabelle 12: Gesendete Nachrichten zur Einlagerung in der Datenbank

Primary Key	Name	Wert	Zeit
1	Byte	10	16:03:13 22.11.21
2	Int	38	16:03:13 22.11.21
3	DInt	0	16:03:13 22.11.21
4	Word	1	16:03:13 22.11.21
5	DWord	32	16:03:13 22.11.21
6	Real	3,21	16:03:13 22.11.21
7	Byte	3	16:04:13 22.11.21
8	Int	22	16:04:13 22.11.21
9	DInt	-22	16:04:13 22.11.21
10	Word	2	16:04:13 22.11.21
11	DWord	64	16:04:13 22.11.21
12	Real	2,65	16:04:13 22.11.21
13	Byte	7	16:05:13 22.11.21
14	Int	-1	16:05:13 22.11.21
15	DInt	22222	16:05:13 22.11.21
16	Word	4	16:05:13 22.11.21
17	DWord	128	16:05:13 22.11.21
18	Real	-4,6	16:05:13 22.11.21
19	Byte	12	16:06:13 22.11.21
20	Int	255	16:06:13 22.11.21
21	DInt	-365	16:06:13 22.11.21
22	Word	8	16:06:13 22.11.21
23	DWord	256	16:06:13 22.11.21
24	Real	8	16:06:13 22.11.21
25	Byte	0	16:07:13 22.11.21
26	Int	3	16:07:13 22.11.21
27	DInt	3	16:07:13 22.11.21
28	Word	16	16:07:13 22.11.21
29	DWord	512	16:07:13 22.11.21
30	Real	16,55	16:07:13 22.11.21

Tabelle 13: Eingelagerte Daten in der Datenbank

Die, in der Datenbank eingelagerten, Daten sind identisch mit den Daten in den Nachrichten. Da alle Daten aus den Nachrichten in der Datenbank eingelagert sind, kann das automatische Auslösen des Lambda-Programms und das Zerlegen der Nachrichten durch das Lambda-Programm als erfolgreich angesehen werden.

6.2 Integrationstest der automatischen Übertragung von der SPS zur Datenbank

Bei dem Integrationstest der automatischen Übertragung der Prozessdaten von der SPS bis in die Datenbank AWS Aurora, wird überprüft, ob die Übertragung und das Einlagern der Daten automatisch stattfinden. Hierfür wird die erstellte Datenbank geleert und die Übertragung der Prozessdaten vom EWON Flexy 205 zum AWS IOT Core aktiviert. Damit beginnt der EWON die Daten von der SPS abzugreifen und an den AWS IOT Core zu senden. Der AWS IOT Core löst automatisch die Übergabe an AWS Lambda aus, wo die Daten in die Datenbank einsortiert werden. Auf der SPS-Ebene werden nun die zu

sendenden Prozessdaten verändert, sodass unterschiedliche Daten in der Datenbank eingelagert werden müssen. Die Anzahl der übersendeten Nachrichten vom EWON wird auf dem AWS IOT Core beobachtet und gezählt. Nach fünffacher Sendung der Prozessdaten wird die Datenbank über einen SQL-Befehl ausgelesen und die Sendung der Daten auf dem EWON gestoppt. Die, durch den SQL-Befehl in die Datenbank eingelagerten, Daten werden den in der SPS eingetragenen Daten gegenübergestellt und verglichen. Die MQTT-Sendezeit wird im EWON Flexy 205 auf 60 Sekunden eingestellt, um genügend Zeit zum Ändern der Daten auf der SPS-Ebene zu haben. Die nachfolgende Tabelle zeigt die auf der SPS-Ebene einzustellenden Datenpunkte (siehe Tabelle 14).

Variablenname	Wert 1	Wert 2	Wert 3	Wert 4	Wert 5
Byte	1	2	4	8	16
Int	-56	465	-231	56	23
DInt	369	-25	-1	255	10
Word	1	2	4	8	16
DWord	32	64	128	256	512
Real	0,23	-2,36	6,258	-99,3	10,0

Tabelle 14: Von der SPS zu sendende Daten

Die Daten in der Tabelle 14 sind, um das Vergleichen zu vereinfachen, in Datenpaketen dargestellt. Diese Datenpakete befinden sich jeweils immer in einer vom EWON gesendeten MQTT-Nachricht und werden somit auch immer gleichzeitig in AWS Aurora eingelagert.

Die durch AWS Lambda in der Datenbank AWS Aurora eingelagerten Daten werden in der Tabelle 15 dargestellt.

Primary Key	Name	Wert	Zeit
1	Byte	1	18:33:25 22.11.21
2	Int	-56	18:33:25 22.11.21
3	DInt	369	18:33:25 22.11.21
4	Word	1	18:33:25 22.11.21
5	DWord	32	18:33:25 22.11.21
6	Real	0,23	18:33:25 22.11.21
7	Byte	2	18:34:25 22.11.21
8	Int	465	18:34:25 22.11.21
9	DInt	-25	18:34:25 22.11.21
10	Word	2	18:34:25 22.11.21
11	DWord	64	18:34:25 22.11.21
12	Real	-2,36	18:34:25 22.11.21
13	Byte	4	18:35:25 22.11.21
14	Int	-231	18:35:25 22.11.21
15	DInt	-1	18:35:25 22.11.21
16	Word	4	18:35:25 22.11.21
17	DWord	128	18:35:25 22.11.21
18	Real	6,258	18:35:25 22.11.21
19	Byte	8	18:36:25 22.11.21
20	Int	56	18:36:25 22.11.21
21	DInt	255	18:36:25 22.11.21
22	Word	8	18:36:25 22.11.21
23	DWord	256	18:36:25 22.11.21
24	Real	-99,3	18:36:25 22.11.21
25	Byte	16	18:37:25 22.11.21
26	Int	23	18:37:25 22.11.21
27	DInt	10	18:37:25 22.11.21
28	Word	16	18:37:25 22.11.21
29	DWord	512	18:37:25 22.11.21
30	Real	10,0	18:37:25 22.11.21

Tabelle 15: In der Datenbank automatisch eingelagerte Daten

Nach Vergleich der beiden Tabellen 14 und 15 ist zu erkennen das die von der SPS stammenden Daten identisch mit den Daten in der Datenbank sind und die Übertragung der Daten automatisch stattfindet. Die Funktion der automatischen Übertragung von der SPS bis in die Datenbank ist somit erfüllt.

7. Zusammenfassung und Ausblick

Ziel dieser Arbeit war die Konzeption und Entwicklung eines automatisierten, datenumfangsoptimierten, Cloud-basierten Systems zur vorbeugenden Instandhaltung einer Fassreinigungsanlage. Um dieses Ziel umzusetzen, wurde das Projekt in Arbeitsphasen unterteilt. Zuerst wurde der aktuelle Forschungsstand untersucht, um wichtige Erkenntnisse über die Themengebiete der Arbeit zu erlangen. Ebenfalls wurde eine verkürzte Marktanalyse durchgeführt, um eine Aussage über die Wirtschaftlichkeit des Produktes und das bestehende Marktpotential des Zielmarktes zu treffen. Es wurden benötigte technische Grundlagen und die bestehende Ausgangssituation, sowie die Funktion der Fassreinigungsanlage erläutert. Nach der Beschreibung der Ausgangssituation wurden die Anforderungen an das Projekt in einer Anforderungsliste festgehalten und diese Anforderungen der Konzeption bereitgestellt. In der Konzeptionsphase wurden die umzusetzenden Teilkomponenten aus dem Gesamtsystem extrahiert. Für die Teilkomponenten wurden Konzepte zur Umsetzung methodisch ermittelt und mit den Anforderungen verglichen, um das optimale Teilkonzept auszuwählen. Zur Findung der Teilkonzepte wurde die Methode des morphologischen Kastens verwendet. Im Anschluss wurden Nutzwertanalysen durchgeführt, mit denen die Teilkonzepte mit den Anforderungen abgeglichen wurden. Nach der Auswahl der einzelnen Teilkonzepte wurden die Ergebnisse der Konzeptentwicklung erläutert und in einem Gesamtkonzept zusammengefasst. In der Realisierungsphase wurden die geplanten Teilkonzepte und die Programme zur Übertragung der Prozessdaten methodisch durch Flussdiagramme umgesetzt. Die in den Flussdiagrammen beschriebene Logik, wurde nachfolgend in den unterschiedlichen Programmiersprachen realisiert. Zuerst wurde der SPS-Baustein zur datenumfangsoptimierten Übertragung der Prozessdaten auf das SCADA-System umgesetzt, damit die Prozessdaten anlagennah ohne Kostenzunahme dargestellt werden können. Nachdem die datenumfangsoptimierte Übertragung implementiert wurde, wurden die Prozessdaten an das ausgewählte Gateway, den EWON Flexy 205, übergeben. Bevor die Prozessdaten mit dem Übertragungsstandard MQTT an die Cloud-Struktur übergeben werden konnten, musste die Cloud-Struktur aufgebaut werden. Die Cloud-Struktur besteht aus dem AWS IOT Core, der die Prozessdaten vom Gateway erhält, dem Verarbeitungswerkzeug AWS Lambda, der die Prozessdaten aus der Nachricht entnimmt und die Datenbankbefehle ausführt, und der Datenbank AWS Aurora, zur Haltung der Daten. Die Übertragung der Prozessdaten vom EWON zum AWS IOT Core wurde mit dem Übertragungsprotokoll MQTT umgesetzt und die Daten werden im JSON-Format versendet. Nach dem Erhalt der Nachricht, wird diese an AWS Lambda übergeben und automatisch in der Datenbank eingelagert. Die Datenbank sollte mit der Analyseplattform Qlik Sense verbunden werden, jedoch ist dies mit dem geplanten Verbinder nicht möglich. Die Produktkomponenten, die realisiert wurden, wurden vollständig getestet und sichergestellt das die Funktionen hinreichend erfüllt sind. Das Projekt kann durch die fehlende Verbindungsmöglichkeit von der Datenbank AWS Aurora zu der Analyseplattform Qlik Sense nicht abgeschlossen werden. Ebenfalls kann ohne die Verbindung der Datenbank mit Qlik Sense und den dadurch fehlenden Daten keine Analysen in Qlik Sense umgesetzt werden.

Für ein Gelingen des Projektes muss eine zweite Entwicklungsschleife durchgeführt werden, in der die nicht realisierbaren Bestandteile ausgetauscht werden. Der Verbinder von Qlik Sense muss gewechselt werden, um eine Verbindung zu ermöglichen. Der Wechsel des Verbinders garantiert jedoch nicht das Gelingen der Verbindung. Sobald alle möglichen Verbinder getestet sind und die Verbindung immer noch nicht zustande kommen kann, muss die Datenbank angepasst und verändert werden. Gegebenenfalls müssen andere Datenbanken implementiert und getestet werden. Nach der erfolgreichen Verbindung der Datenbank mit Qlik Sense können die Analysen zur vorbeugenden Instandhaltung durchgeführt werden.

Literaturverzeichnis

[1] Qlik (2020). *Qlik Sense* [Datasheet]. https://www.qlik.com/us/-/media/files/resource-library/global-us/direct/datasheets/ds-qlik-sense-datasheet-en.pdf abgerufen am 27.11.2021

[2] Schenk, M. (2010). *Instandhaltung technischer Systeme: Methoden und Werkzeuge zur Gewährleistung eines sicheren und wirtschaftlichen Anlagenbetriebs* (1. Aufl.). Springer-Verlag Berlin Heidelberg.

[3] Theuerkauf, J. (2012). *Schreiben im Ingenieurstudium: Effektiv und effizient zu Bachelor-, Master- und Doktorarbeit* (1. Aufl.). Brill | Schöningh.

[4] ENIP AG (2012). *DIN 30051: Grundlagen der Instandhaltung* [DIN]. https://www.enip.ch/images/enip/pdfs/ih-grundlage-din-31051.pdf abgerufen am 27.11.2021

[5] Bök, P.-B., Noack, A., Müller, M. & Behnke, D. (2020). *Computernetze und Internet of Things: Technische Grundlagen und Spezialwissen* (1. Aufl.). Springer Vieweg.

[6] Sommerville, I. (2018). *Software Engineering* (10. Aufl.). Pearson Studium.

[7] Dai., Z. R. (2018). *Software Engineering 2. Skript (SE – Einführung)*.

[8] Dai., Z. R. (2018). *Software Engineering 4. Skript (SE – UML Use Cases)*.

[9] Seitz, M. (2015). *Speicherprogrammierbare Steuerungen für die Fabrik- und Prozessautomation: Strukturierte und objektorientierte SPS-Programmierung, Motion Control, Sicherheit, vertikale Integration* (4. Aufl.). Carl-Hanser-Verlag.

[10] HMS Industrial Networks GmbH (2020). *EWON Flexy 205* [Datasheet]. https://www.ewon.biz/de/produkte/flexy/ewon-flexy-205?ordercode=Flexy20500_00MA abgerufen am 27.11.2021

[11] Naefe, P. (2012). *Einführung in das Methodische Konstruieren: für Studium und Praxis* (2. Aufl.). Springer Vieweg.

[12] Statista (2021). *Wichtigste Unternehmen für industrielle Instandhaltung* in Deutschland nach Umsatz 2019* [Statistik]. https://de.statista.com/statistik/daten/studie/449578/umfrage/wichtigste-deutsche-industrieservice-unternehmen-nach-umsatz/ abgerufen am 27.11.2021

[13] Statista (2021). *Welche der folgenden Predictive-Maintenance-Anwendungen nutzen Sie bereits?* [Statistik]. https://de.statista.com/statistik/daten/studie/1078451/umfrage/nutzung-von-predictive-maintenance-anwendungen-in-deutschland/ abgerufen am 27.11.2021

[14] Schmertosch, T. & Krabbes, M. (2018). *Automatisierung 4.0: objektorientierte Entwicklung modularer Maschinen für die digitale Produktion* (1. Aufl.). Carl-Hanser-Verlag.

[15] Hausladen, I. (2016). *IT-gestützte Logistik: Systeme - Prozesse – Anwendungen* (3. Aufl.). Springer Gabler.

[16] Berekoven, L., Eckert, W. & Ellenrieder P. (2009). *Marktforschung: methodische Grundlagen und praktische Anwendung* (12. Aufl.). Gabler Verlag.

[17] Backerra, H., Malorny, C. & Schwarz, W. (2007). *Mensch und Computer - Workshopband* (1. Aufl.). De Gruyter Oldenbourg.

[18] Burmester, M., Schmidt, A. & Weisbecker, A. (2015). *Kreativitätstechniken: Kreative Prozesse anstoßen, Innovationen fördern* (3. Aufl.). Carl Hanser Verlag.

[19] Schlattmann, J. & Seibel, A. (2017). *Aufbau und Organisation von Entwicklungsprojekten* (1. Aufl.). Springer.

[20] Weidmüller Interface GmbH & Co. KG (2021). *IOT GW30* [Datasheet]. https://catalog.weidmueller.com/catalog/Start.do?ObjectID=2682620000 abgerufen am 27.11.2021

[21] Meyer, U.B., Creux, S.E. & Weber Marin, A.K. (2005). *Grafische Methoden der Prozessanalyse: Für Design und Optimierung von Produktionssystemen* (1. Aufl.). Hanser Verlag.

Anhang A - Programmcode

SPS-Baustein:

SEQ TRANS

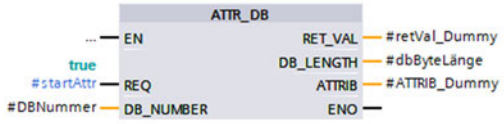
IF... | CASE... OF... | FOR... TO DO... | WHILE... DO... | (*...*) | REGION

Bausteintitel: Sequentielle Übertragung von Prozessdaten

Stellt sequentiell Prozessdaten eines Datentyps zur Abfrage vom SCADA-System bereit und prüft die Richtigkeit dieser Daten.

Netzwerk 1: Länge des DB in Byte ermitteln

Ermittelt die Länge des bei "DB_NUMBER" angegebenen DB´s in Byte und schreibt den Wert in eine Variable

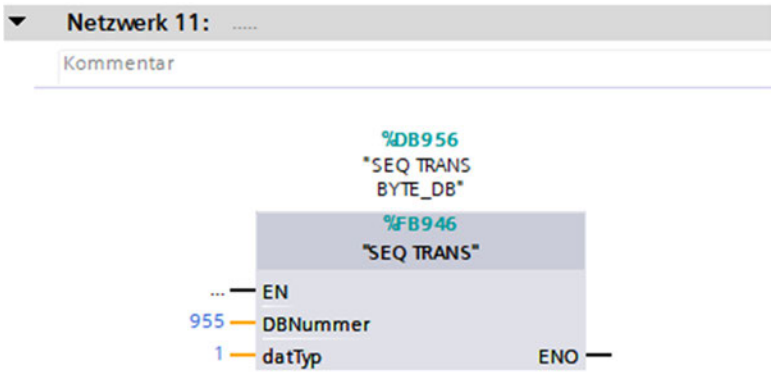


Netzwerk 2: Bereitstellen der Variablen und Überprüfung der sicheren Übertragung

Kommentar

```
// Ermittelt für den angegebenen Integer, welcher einem Datentyp entspricht, die Länge in Byte.
// Datentypen mit entsprechenden Zahlen:
// (Byte = 1; INT = 2; DINT = 3; WORD = 4; DWORD = 5; REAL = 6)
IF #datTyp = 1 THEN
    #anzByte := 1;
    // Liest mit dem Peek-Befehl den Variablenwert als BYTE aus der aktuellen Zeile
    #varWert := PEEK_BYTE(area := 16#84, dbNumber := #DBNummer, byteOffset := #aktZeile * #anzByte);
ELSIF #datTyp = 2 OR #datTyp = 4 THEN
    #anzByte := 2;
    #varWert := PEEK_WORD(area := 16#84, dbNumber := #DBNummer, byteOffset := #aktZeile * #anzByte);
ELSIF #datTyp = 3 OR #datTyp = 5 OR #datTyp = 6 THEN
    #anzByte := 4;
    #varWert := PEEK_DWORD(area := 16#84, dbNumber := #DBNummer, byteOffset := #aktZeile * #anzByte);
END_IF;

// Wandelt die ermittelte DB-Länge in Byte in die Anzahl der zu übertragenden Variablen und somit in die Anzahl der Zeilen
#anzZeilen := UDINT_TO_UINT(IN := #dbByteLänge) / #anzByte;

// Wiederholendes Ausführen des Triggers zum Abgleich der vom SCADA-Sytsem gesendeten Daten
IF #vergleichen = TRUE AND #korrÜbertragen = FALSE THEN
    #vergleichen := FALSE;
ELSIF #vergleichen = FALSE AND #korrÜbertragen = FALSE THEN
    #vergleichen := TRUE;
END_IF;

// Prüft ob die vom SCADA-System abgefragten Daten den reellen Daten entsprechen, setzt den Indikator für die erfolgreiche
                                                                    Übertragung auf TRUE und die Variablen zur Kontrolle auf 0
IF #varWert = #erhVarWert AND #aktZeile = #gelAktZeile AND #datTyp = #erhDatTyp THEN
    #korrÜbertragen := TRUE;
    #vergleichen := FALSE;
    #erhVarWert := 0;
    #gelAktZeile := 0;
    #erhDatTyp := 0;
END_IF;

// Wiederholendes Ausführen des Triggers zur Speicherung der Daten im SCADA-System
IF #erfÜbertragen = TRUE AND #korrÜbertragen = TRUE THEN
    #erfÜbertragen := FALSE;
ELSIF #erfÜbertragen = FALSE AND #korrÜbertragen = TRUE THEN
    #erfÜbertragen := TRUE;
END_IF;

// Setzt die aktuelle Zeile auf die nächste Zeile nach erfolgreicher Speicherung
IF (#aktZeile + 1) < #anzZeilen AND #gesichert = TRUE THEN
    #aktZeile := #aktZeile + 1;
    #erfÜbertragen := FALSE;
    #gesichert := FALSE;
    #korrÜbertragen := FALSE;
ELSIF (#aktZeile + 1) = #anzZeilen AND #gesichert = TRUE THEN
    #aktZeile := 0;
    #erfÜbertragen := FALSE;
    #gesichert := FALSE;
    #korrÜbertragen := FALSE;
END_IF;
```

Aufruf des SPS-Bausteines im OB1:



Überprüfen der SPS-Daten im SCADA-System:

```
#include "apdefap.h"

int gscAction( void )
{
	// WINCC:TAGNAME_SECTION_START
	// syntax: #define TagNameInAction "DMTagName"
	#define abgleichen_BYTE "vergleichen_BYTE"
	#define antwort_BYTE "erhVarWert_BYTE"
	#define nachricht_BYTE "varWert_BYTE"
	#define gelZeile_BYTE "gelAktZeile_BYTE"
	#define aktZeile_BYTE "aktZeile_BYTE"
	#define erhTyp_BYTE "erhDatTyp_BYTE"
	#define datTyp_BYTE "datTyp_BYTE"
	// next TagID : 1
	// WINCC:TAGNAME_SECTION_END

	// WINCC:PICNAME_SECTION_START
	// syntax: #define PicNameInAction "PictureName"
	// next PicID : 1
	// WINCC:PICNAME_SECTION_END

	// Schreibt die gelesenen Werte an die SPS zurück, wenn der Trigger wahr ist
	if(GetTagBit(abgleichen_BYTE) == TRUE){
		SetTagDWord(antwort_BYTE, GetTagDWord(nachricht_BYTE));
		SetTagWord(gelZeile_BYTE, GetTagWord(aktZeile_BYTE));
		SetTagWord(erhTyp_BYTE, GetTagWord(datTyp_BYTE));
	}

	return 0;
}
```

Speichern der SPS-Daten im SCADA-System:

```
#include "apdefap.h"

int gscAction( void )
{
	// WINCC:TAGNAME_SECTION_START
	// syntax: #define TagNameInAction "DMTagName"
	#define TRUE 1
	#define FALSE 0

	#define nachricht_BYTE "varWert_BYTE"
	#define aktZeile_BYTE "aktZeile_BYTE"
	#define gesichert_BYTE "gesichert_BYTE"
	#define korrÜbertragen_BYTE "korrÜbertragen_BYTE"

	#define Kugelhahn_BYTE_1 "Signal_BYTE_1"
	#define Kugelhahn_BYTE_2 "Signal_BYTE_2"
	#define Kugelhahn_BYTE_3 "Signal_BYTE_3"
	#define Kugelhahn_BYTE_4 "Signal_BYTE_4"
	#define Kugelhahn_BYTE_5 "Signal_BYTE_5"
	#define Kugelhahn_BYTE_6 "Signal_BYTE_6"
	// next TagID : 1
	// WINCC:TAGNAME_SECTION_END

	// WINCC:PICNAME_SECTION_START
	// syntax: #define PicNameInAction "PictureName"
	// next PicID : 1
	// WINCC:PICNAME_SECTION_END

	// Wenn der Trigger = TRUE ist wird mit der aktuellen Zeile das Signal
	//		ermittelt und der Wert beim richtigen Signal gespeichert
	if(GetTagBit(korrÜbertragen_BYTE) == TRUE){
		switch(GetTagWord(aktZeile_BYTE)){
			case 0:
				SetTagByte(Kugelhahn_BYTE_1, GetTagByte(nachricht_BYTE));
				break;
			case 1:
				SetTagByte(Kugelhahn_BYTE_2, GetTagByte(nachricht_BYTE));
				break;
			case 2:
				SetTagByte(Kugelhahn_BYTE_3, GetTagByte(nachricht_BYTE));
				break;
			case 3:
				SetTagByte(Kugelhahn_BYTE_4, GetTagByte(nachricht_BYTE));
				break;
			case 4:
				SetTagByte(Kugelhahn_BYTE_5, GetTagByte(nachricht_BYTE));
				break;
			case 5:
				SetTagByte(Kugelhahn_BYTE_6, GetTagByte(nachricht_BYTE));
				break;
		}
		// Setzt nach erfolgreichem Speichern das Bit, damit die SPS den
		//		nächsten Wert übergeben kann
		SetTagBit(gesichert_BYTE, TRUE);
	}
	return 0;
}
```

Programm der BASIC-IDE des EWON:

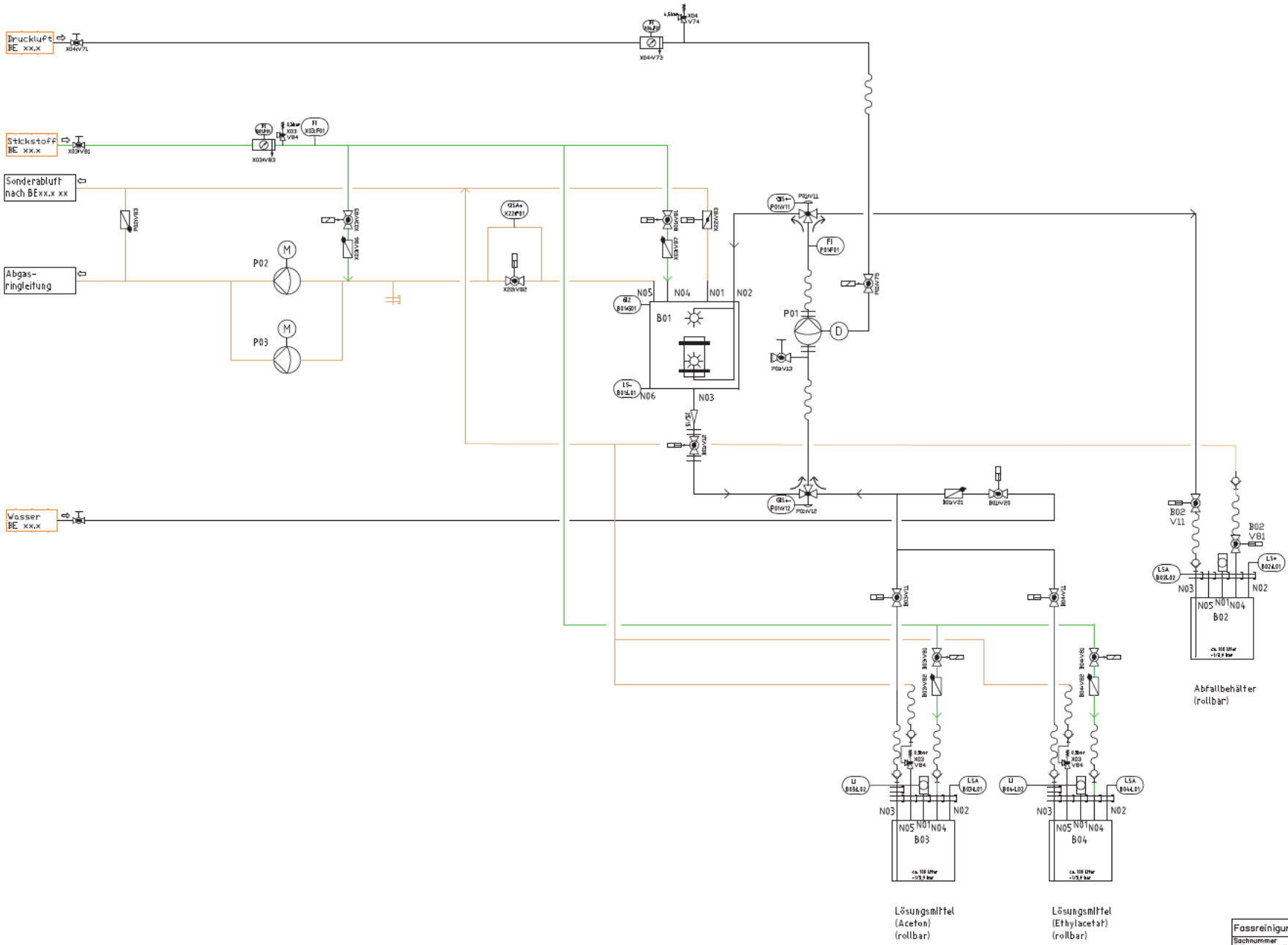
```
Init Section
CLS                             // Zurücksetzen der Konsole
SETSYS INF, "LOAD"              // Lädt die Informationen des EWON
SerNum$ = GETSYS INF, "SERNUM"  // Speichert die IP-Adresse des EWON in einer String-Variable

//########   Konfigurierung der MQTT-Daten   ###############
MQTTBrokerURL$ = "a2iz8wzea6h79h-ats.iot.eu-central-1.amazonaws.com"  // String-Variable mit Broker-Adresse
MQTTPort$ = "8883"                                                   // String-Variable mit Definition des MQTT-Ports
TopicToPublishOn$ = "/Fassreinigung"                                 // String-Variable mit Beschreibung zu welchem Thema veröffentlicht wird
//######## Ende der Konfigurierung der MQTT-Daten ###############

// Start des Scripts
Last_ConnStatus% = 0            // Deklaration und Initialisierung einer Int-Variable der den Status der letzten Verbindung angibt

// Konfigurieren des Verbindungsaufbaus mit dem MQTT-Broker
CONNECTMQTT:
  MQTT "OPEN", "SGA-EWON", MQTTBrokerURL$                        // Öffnet die Verbindung mit dem MQTT-Broker
  MQTT "SETPARAM", "PORT", MQTTPort$                             // Setzt die Parameter für den Port
  MQTT "SETPARAM", "KEEPALIVE", "20"                             // Setzt die Parameter wann gesendet werden soll
  MQTT "SETPARAM", "cafile", "/usr/AwsCertificates/rootCA.crt"
  MQTT "SETPARAM", "certfile", "/usr/AwsCertificates/device.cert.pem"
  MQTT "SETPARAM", "keyfile", "/usr/AwsCertificates/device.private.key"

  SETSYS PRG,"RESUMENEXT", 1                  // Setzt die Programmparameter, sodass bei Auftreten eines Fehlers der nachfolgende Quelltext trotzdem ausgeführt wird
  MQTT "CONNECT"                              // Verbindet die Teilnehmer über MQTT
  ErrorReturned% = GETSYS PRG,"LSTERR"        // Prüft ob ein Fehler aufgetreten ist und schreibt den entsprechenden Intergerwert in eine Variable
  IF ErrorReturned% = 28 THEN                 // Wenn der Fehlerintegerwert 28 entspricht wird die Fehlermeldung gespeichert
    @Log("[MQTT SCRIPT] WAN-Schnittstelle noch nicht bereit")
  ENDIF
  SETSYS PRG,"RESUMENEXT", 0       // Setzt die Programmparameter zurück, sodass bei Auftreten eines Fehlers der nachfolgende Quelltext nicht ausgeführt wird
  ONTIMER 1, "GOTO SENDDATA"       // Sendet nachdem der Timer 1 abgelaufen ist die Nachricht
  TSET 1, 10                       // Setzt den Timer 1 auf 10 Sekunden und führt ihn immer wieder aus
END

// Sendet die im EWON befindlichen Tags im JSON-Format als MQTT-Nachricht
SENDDATA:
  ConnStatus% = MQTT "STATUS"                                    // Liest den MQTT-Verbindungsstatus (5 = verbunden, andere Werte = nicht verbunden)
  IF Last_ConnStatus% <> ConnStatus% THEN                        // Prüft die Änderung des Verbindungsstatus
    IF ConnStatus% = 5 THEN                                      // Prüft ob die Verbindung mit dem Broker intakt ist
      @Log("[MQTT SCRIPT] Flexy verbunden mit Broker")           // Speichert mit der Funktion "Log" die erfolgreiche Verbindung zum Broker
    ELSE
      @Log("[MQTT SCRIPT] Flexy getrennt von Broker")            // Speichert mit der Funktion "Log" die gescheiterte Verbindung zum Broker
    ENDIF
    Last_ConnStatus% = ConnStatus%                               // Speichert den aktuellen Verbindungsstatus als letzten Verbindungsstatus
  ENDIF
  IF ConnStatus% = 5 THEN                                        // Prüft ob erfolgreich verbunden wurde und sendet die Daten
    NB% = GETSYS PRG,"NBTAGS"                                    // Gibt die Anzahl der Im EWON definierten Tags wieder

    // Zusammenbau der Nachricht im JSON-Format
    MsgToPublish$ = "["
    FOR i% = 0 TO NB%-1
      SETSYS Tag, "load",-i%
      TagName$ = GETSYS Tag, "Name"
      TagValue$ = GETSYS Tag, "TagValue"
      IF i% = 0 THEN
        MsgToPublish$ = MsgToPublish$ + '{"tag":"' + TagName$ + '","value":"' + TagValue$ + '","time":"' + @GetTime$() + '"}'
      ELSE
        MsgToPublish$ = MsgToPublish$ + ',{"tag":"' + TagName$ + '","value":"' + TagValue$ + '","time":"' + @GetTime$() + '"}'
      ENDIF
    NEXT i%
    MsgToPublish$ = MsgToPublish$ + "]"

    // Publishen der Nachricht
    MQTT "PUBLISH",  TopicToPublishOn$ , MsgToPublish$, 0,0
    PRINT "[MQTT SCRIPT] Nachricht '" + MsgToPublish$ + "' gesendet zum Thema : " +  TopicToPublishOn$  // Veröffentlicht die Nachricht in der Konsole
  ELSE
    @Log("[MQTT SCRIPT] Flexy nicht verbunden")                  // Wenn nicht verbunden zu Broker logge das der Flexy nicht verbunden ist
  ENDIF
END

// Funktion zum lokalen Loggen der Meldungen
FUNCTION Log($Msg$)
  LOGEVENT $Msg$ ,100
  PRINT $Msg$
ENDFN

// Funktion zur Ermittlung des Datums und der Zeit
FUNCTION GetTime$()
  $a$ = Time$
  $GetTime$ = $a$(7 To 10) + "-" + $a$(4 To 5) + "-" + $a$(1 To 2) + " " + $a$(12 To 13)+":"+$a$(15 To 16)+":"+$a$(18 To 19)
ENDFN
```

Programm in AWS Lambda:

```
// require the AWS SDK
const AWS = require('aws-sdk');
const rdsDataService = new AWS.RDSDataService();

exports.handler = (event, context, callback) => {

  var FRstring = JSON.stringify(event, null, 0);

  var i = 0;                // Bedingung für While-Schleife
  var row = 0;              // Iterationsparameter für die Nachricht, gibt immer die erste String-Stelle der zu speichernden Zeile

  while (i !== 1) {
    var name = '';          // Name der aktuellen Teilnachricht
    var value = '';         // Wert der aktuellen Teilnachricht
    var time = '';          // Zeit der aktuellen Teilnachricht
    var col = 0;            // Spalte des aktuellen Wertes

    for (var o = row + 4; FRstring[o] !== '}'; o++) {           // Iteriert durch die Nachricht bis an der nächsten Stelle eine "}" kommt
      if (FRstring[o - 1] === ':' & FRstring[o] === '"') {        // Sucht den Beginn der zu entnehmenden Werte
        col = col + 1;                                              // Setzt die Spalte hoch
        o++;                                                        // Zählt die Laufvariable der Schleife hoch
        while (FRstring[o] !== '"' & col === 1) {                   // Schleife läuft solange hoch bis das Ende des Wertes erkannt wird
          name = name + FRstring[o];                                  // Speichert den Namen
          o++;                                                        // Zählt die Laufvariable der Schleife hoch
        }
        while (FRstring[o] !== '"' & col === 2) {                   // Schleife läuft solange hoch bis das Ende des Wertes erkannt wird
          value = value + FRstring[o];                                // Speichert den Wert
          o++;                                                        // Zählt die Laufvariable der Schleife hoch
        }
        while (FRstring[o] !== '"' & col === 3) {                   // Schleife läuft solange hoch bis das Ende des Wertes erkannt wird
          time = time + FRstring[o];                                  // Speichert die Zeit
          o++;                                                        // Zählt die Laufvariable der Schleife hoch
        }
      }
      row = o;                                                    // Setzt den Wert der Iteration als Zeilenwert
    }

    if (FRstring[row + 2] === ']'){                             // Prüft ob das Ende der nachricht erreicht wurde
      i = 1;                                                      // Setzt die While-Bedingung auf 0
    }

    // Erstellt den SQL-Befehl als string
    var sqlcommand = "INSERT INTO sga.Fassreinigung (tag, value, xtime) VALUES ('" + name + "', '" + value + "', '" + time + "');";

    // Definiert den SQL-Befehl und den Zugangspunkt der Datenbank
    let sqlParams = {
      secretArn: 'arn:aws:secretsmanager:eu-central-1:570826932166:secret:auroralab-mysql-serverless-secret-sga-w7KbdD',
      resourceArn: 'arn:aws:rds:eu-central-1:570826932166:cluster:sga-auroralab-mysql-serverless',
      sql: sqlcommand,
      database: 'sga',
      includeResultMetadata: true
    };

    // Führt den definierten SQL-Befehl aus
    rdsDataService.executeStatement(sqlParams, function (err, data) {
      if (err) {
        // error
        console.log(err);
        callback('Query Failed');
      }
    });
  }
};
```

Anhang B - Fließbilder

Fließbild der Fassreinigungsanlage:



Fassreinigungsanlage		
Sachnummer BExx.xx_A0x_RI_20xxxxxx		
Gesetzliche Eigentümer Sigma-Aldrich Biochemie Beispielstrasse 14 21147 Hamburg www.sial.com	Dokumentenart RI—Fliessbild	Ersteller Kuhlmann
	Abschnitts-/Blattnummer 1/1	Genehmigende Person
	Ausgabedatum	Letzte Änderung durch/am Timmann 13.04.21

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: Banse

Vorname: Aike

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

Entwicklung eines automatisierten, datenumfangsoptimierten, Cloud-basierten Systems für die vorbeugende Instandhaltung einer Fassreinigungsanlage

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	29.11.2021	[redacted]
Ort	Datum	Unterschrift im Original