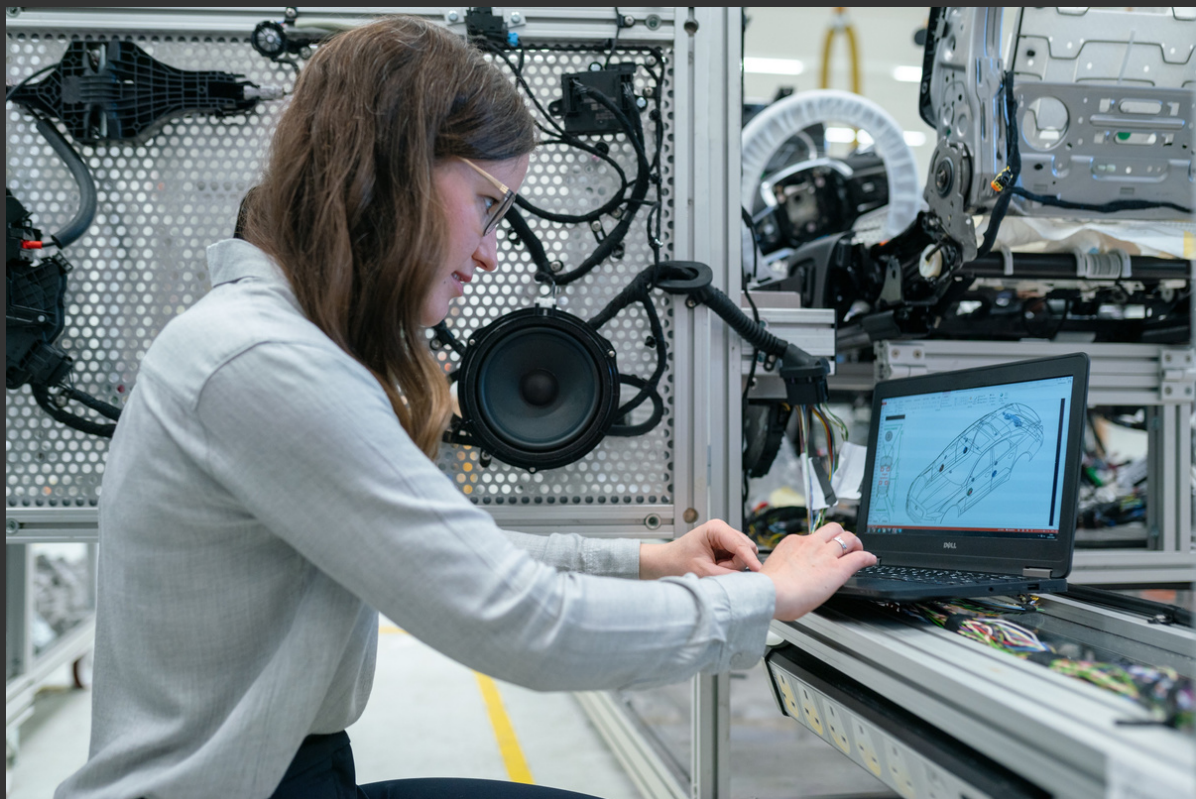


So gelingt der Weg zum digitalen Produkt

Connectivity - Data Platforms - KI Services



Inhalte

Abstract	01
-----------------	-----------

1. CONNECTIVITY

1.1	Warum überhaupt die lokale Maschinenwelt in die Cloud bringen?	02
1.2	Strategische Aspekte von Konnektivität: Welche Art der Datenintegration ist die „Richtige“?	05
1.3	Hardware-Komponenten und Übertragungsprotokolle zur Ermöglichung von Konnektivität	07
1.4	Die wichtigsten Übertragungsprotokolle im Industriekontext	11
	4.1 MQTT	12
	4.2 OPC UA	14
	4.3 Protokollvergleich	17
1.5	Zusammenfassung und Ausblick	19

2. DATA PLATFORMS

2.1	Der Plattform-Ansatz für die Welt der digitalen Daten	20
2.2	Anforderungen und Ziele einer Datenplattform	23

2.3	Komponenten und Funktionen einer Datenplattform	25
2.3.1	Source & Ingestion Layer	25
2.3.2	Storage & Compute Layer	28
2.3.3	Consumption Layer	29
2.3.4	Governance & Orchestration Layer	29
2.4	Zusammenfassung und Ausblick	31

3. KI SERVICES

3.1	Industrial Analytics und datengetriebene Wertschöpfung	32
3.2	Typische Use Cases (Problem - Ansatz - Lösung)	33
3.2.1	Predictive Maintenance	33
3.2.2	Automated Quality Control	34
3.2.3	Machine Efficiency Optimization	35
3.3	Vorgehensweise und Rollen - Von der Idee zum digitalen Produkt	37
3.3.1	Allgemeines Vorgehen	37
3.3.2	Involvierte Rollen	43
3.4	Zusammenfassung und Ausblick	45

Kontakt	46
----------------	-----------

Abstract

Dieses Whitepaper zeigt, wie sich Industrial Analytics in der Industrie einsetzen lässt, um mit KI-basierten Services Mehrwert – sowohl für interne Abläufe als auch für externe Kunden – zu generieren.

Derzeit erlebt die globale Fertigungsindustrie einen fundamentalen Wandel: von der Produktion ausschließlich physischer Güter hin zur zusätzlichen Bereitstellung von datenbasierten Produkten und Services.

Denn die digitale Transformation hat dafür gesorgt, dass eine zunehmende Menge an digitalen Daten entlang der industriellen Wertschöpfungskette zur Verfügung steht: von Simulationsdaten in der Entwicklung, über Sensordaten in der Fertigung bis hin zu Wartungsberichten und Lokalisierungsdaten beweglicher Maschinen.

Die Gewinnung von Erkenntnissen und Wissen aus diesen Daten kann insbesondere im produzierenden Gewerbe ein strategischer Erfolgsfaktor sein, der Herstellern neben der Bereitstellung neuer Produkte und Services nicht zuletzt auch dabei helfen kann, Prozesse zu optimieren, somit ihre Effizienz zu steigern und gleichzeitig die Kosten zu senken. Das macht sie auf einem globalen, sich schnell verändernden Markt wettbewerbsfähiger.

Zusammenfassend erhalten Sie einen ganzheitlichen Einblick in den Prozess der Umwandlung von Sensorinformationen in digitale Produkte durch den Einsatz von Data Science und KI-Services. Sie sind hier richtig, wenn Sie Ihr Potenzial zur Datennutzung und Produktentwicklung maximieren möchten, indem Sie moderne Technologien und Analysemethoden einsetzen.

Die Autoren



Daniel Schweimer
Consultant
Industrial Analytics & IoT
Autor Whitepaper
KI Services



Filip Stepniak
Senior Consultant
Industrial Analytics & IoT
Autor Whitepaper
Data Platforms



Simon Kneller
Head of
Industrial Analytics & IoT
Autor Whitepaper
Connectivity

1 CONNECTIVITY

1.1 Warum überhaupt die lokale Maschinenwelt in die Cloud bringen?

Warum sollte ein Komponenten-, Maschinen- oder Anlagenhersteller überhaupt versuchen Produktionsdaten in die Cloud zu bekommen? Er könnte doch auch intelligente Software ausliefern, die ausschließlich bei den Kunden lokal läuft? Solche grundlegenden Fragen kommen beim Thema Konnektivität zunächst einmal völlig zurecht auf, weshalb in diesem Kapitel genau diese Fragen beantwortet werden.

Angenommen der Hersteller entscheidet sich, sein physisches Produkt mitsamt einer wohldefinierten Datenschnittstelle sowie lokaler Software auszuliefern, die gewisse Analysen für den Kunden ausführt. Falls sowohl Datenschnittstelle als auch Analytics-Software eine hohe Qualität haben, kann dies zu einer positiven Customer Experience führen.

Allerdings hat der Hersteller mit dieser Herangehensweise gleich mehrere Nachteile:

- ✗ Er hat wenig bis keine Informationen darüber, wie der Kunde das Produkt selbst und auch die zugehörige Software nutzt und kann dadurch auch kaum proaktive Unterstützung anbieten. Stattdessen kann er eigentlich nur reaktiv auf Kundenprobleme reagieren.
- ✗ Er kann in die Analytics-Software nur bereits bekanntes Ingenieurwissen einarbeiten. Von datengetriebener

Herangehensweise oder gar datengetriebenen Geschäftsmodellen kann nicht die Rede sein, da keine neuen Features basierend auf erfassten Daten ausgerollt werden können. Denn die im Betrieb anfallenden Daten liegen nur bei jedem Anwender lokal ab.

- ✗ Ein Update der Analytics-Software ist nur sehr eingeschränkt möglich. Updates können überhaupt nur dann gefahren werden, wenn die Software auf Rechnern mit Internetzugang läuft und zusätzlich der Kunde aktiv dem Software-Update zustimmt. Für die klassische App auf dem Smartphone sind diese Voraussetzungen zwar gegeben, in einer Produktionshalle jedoch alles andere als selbstverständlich. Das führt mit der Zeit oftmals zu einem hohen Wartungsaufwand verschiedener Versionen der Software. Denn auch das im Smartphone Bereich längst akzeptierte schnelle Support-Ende älterer Softwareversionen ist im Industrieumfeld alles andere als gewöhnlich oder akzeptiert.
- ✗ Der Hersteller hat die Qual der Wahl: Entweder er entwickelt seine Software für nur ein Betriebssystem und verkleinert damit seine Grundgesamtheit potenzieller Kunden, oder er muss seine Software für mehrere Betriebssysteme entwickeln und warten.

Landen die Daten - und es ist an dieser Stelle nicht die Rede von sensiblen personenbezogenen Daten - hingegen in der Cloud, ergeben sich folgende Vorteile:

- ✓ Der Hersteller erhält Informationen darüber, wie sein Produkt beim Kunden angewendet wird. Er kann bei Fehlanwendung Hilfestellungen anbieten, was die Kundenbindung erhöht und zu zahlreichen Touchpoints mit dem Kunden führt. Außerdem kann er das Anwendungswissen für die Weiterentwicklung seines Produktes nutzen.
- ✓ Der Hersteller kann basierend auf Echtdaten bei seinen Kunden intelligente Services entwickeln, wie Anomaly Detection, Predictive Maintenance, Visual Inspection, usw. (siehe Kapitel 3 *KI-basierte Services*). Dadurch ergeben sich völlig neue Geschäftsmodelle für den Hersteller. Und auch der Anwender profitiert, da das Produkt kontinuierlich besser wird und aus Ausfällen bei anderen Kunden gelernt werden kann, was wiederum den Service für jeden einzelnen Anwender verbessert.
- ✓ Da der Hersteller selbst die Software in der Cloud betreibt, und dem Endkunden "nur" ein Frontend oder auch zusätzlich Datenschnittstellen zur Verfügung stellt, hat dieser die volle Kontrolle über Softwareupdates, ist unabhängig von den bevorzugten Betriebssystemen seiner Kunden und kann auch nachträglich noch grundlegende Datenstrukturen verändern. Er muss lediglich dafür sorgen, dass sich

für den Kunden keine Logikbrüche im Vergleich zum bereits bekannten Frontend ergeben und sollte bei Vorhandensein von Datenschnittstellen diese versionieren.

- ✓ In der Cloud sind leistungsfähigere Rechenressourcen verfügbar als lokal beim Kunden. So können beispielsweise Streaming Services im großen Stil aufgebaut werden, die sich für den einzelnen Kunden gar nicht lohnen würden oder mit den eingeschränkten Rechenressourcen nicht ausführbar wären, für die Gesamtheit der Daten aber durchaus Sinn ergeben.



Wie der vorangegangene Text zeigt, sprechen aus Sicht des Komponenten-, Maschinen- oder Anlagenherstellers zahlreiche Argumente dafür, die lokale Maschinenwelt der Kunden in die Cloud zu bringen.

Doch wie steht es um die Argumente aus Sicht des Kunden und damit des Maschinenbetreibers?

Für diesen ergibt sich ein Trade-off zwischen Datenhoheit/-sicherheit und Serviceorientierung. Steht für ihn der Servicegedanke im Vordergrund, d.h. legt der Kunde hohen Wert auf einen proaktiven Support und zuverlässige Services wie Anomalieerkennung oder Predictive Maintenance, so ist er vermutlich Unterstützer der Übertragung der lokalen Maschinendaten in die Cloud.

Steht für ihn hingegen die alleinige Datenhoheit im Vordergrund, ist es durchaus möglich, dass er eine Übertragung der Daten in die Cloud kategorisch ablehnt.



Unserer Erfahrung nach findet hier aktuell ein Umdenken statt: Während noch vor einigen Jahren Cloudanbindungen im Industrieumfeld kategorisch abgelehnt wurden, nehmen solche Anbindungen gerade im industriellen Umfeld kontinuierlich zu. Die Gründe dafür sind vielschichtig und im Folgenden sollen lediglich die beiden wesentlichsten genannt werden:

- Der zunehmende „Kontakt“ mit der Cloud durch andere IT-Systeme wie z.B. CRM-Systeme oder ERP-Systeme, die in die Cloud verlagert werden, reduziert die Einstiegsängste und -hürden.
- Auf Bedenken bzgl. Datensicherheit wurden inzwischen zahlreiche Antworten in Form von Anonymisierung, Verschlüsselung, etc. gefunden.

Dennoch sollte sich jeder Komponenten-, Maschinen- oder Anlagenhersteller, der vor der in diesem Kapitel beschriebenen „Cloud-Entscheidung“ steht, ausgiebig mit den Bedürfnissen seiner Kunden auseinandersetzen und herausfinden, wie seine Kunden mit dem oben genannten Trade-off umgehen.



Im weiteren Verlauf des Whitepapers wird davon ausgegangen, dass die Entscheidung getroffen ist, lokale Maschinendaten in die Cloud zu bringen.

1.2 Strategische Aspekte von Konnektivität:

Welche Art der Datenintegration ist die „Richtige“?

Wenn erst einmal die Entscheidung getroffen ist, die lokalen Maschinendaten in die Cloud zu bringen, stellt sich die Frage nach der richtigen Integrationsstrategie. Dabei ist grundsätzlich zwischen einer *horizontalen* und einer *vertikalen Datenintegration* zu unterscheiden.

Eine *vertikale Datenintegration* bedeutet, dass der Komponenten-, Maschinen- oder Anlagenhersteller als zukünftiger Anbieter eines digitalen Produktes die notwendige Hardware, um die Daten vom Sensor in die Cloud zu übertragen, vollständig mitliefert. Dies umfasst neben der Sensorik und einem Steuergerät (PLC/SPS) auch ein Gateway, das eine Internetverbindung hat und die anfallenden Daten so in die Cloud senden kann.

Bei einer *horizontalen Datenintegration* hingegen wird davon ausgegangen, dass der Kunde des Komponenten-, Maschinen- oder Anlagenhersteller das Gateway selbst (oft in Form eines Industrie-PCs) bereitstellt und sich

Hardwareleistung des Komponenten-, Maschinen- oder Anlagenherstellers auf Sensorik und Steuergerät (PLC/SPS) beschränkt.

Bei dieser Datenintegrationsstrategie ist es wichtig, dass das Steuergerät die anfallenden Daten in einem Standardformat published, sodass es für den Endkunden relativ einfach ist, diese Daten auf dem Industrie-PC entgegenzunehmen und weiter in die Cloud zu schicken.

Nun stellt sich die Frage, welche Art der Datenintegration die Richtige ist und welche Vor- und Nachteile die beiden Strategien mit sich bringen.

Um diese Frage zu beantworten, sollte der Komponenten-, Maschinen- oder Anlagenhersteller sein Unternehmen zunächst auf Herstellerseite einordnen. Wie Abbildung 1 verdeutlicht, ist es hilfreich, sich als Unternehmen darüber klar zu werden, ob man eher als Teile-, Komponenten-, Maschinen- oder Anlagenhersteller fungiert.

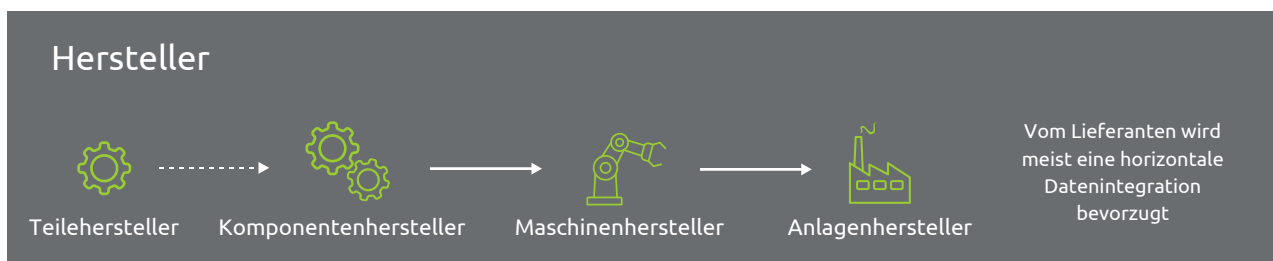


Abb. 1 Herstellerseite

In der Abbildung bedeutet das, sich für eines der grünen Symbole im Hersteller-Bereich zu entscheiden. Je nach Hersteller und Produktportfolio wird diese Einordnung sehr leicht oder auch ziemlich schwerfallen.

Als Nächstes gilt es zu untersuchen, an welche Art von Kunden der Komponenten-, Maschinen- oder Anlagenhersteller sein physisches Produkt hauptsächlich verkauft. In Abbildung 2 bedeutet das, den Pfeilen zu folgen, die vom zuvor ausgewählten Herstellersymbol weg zeigen.

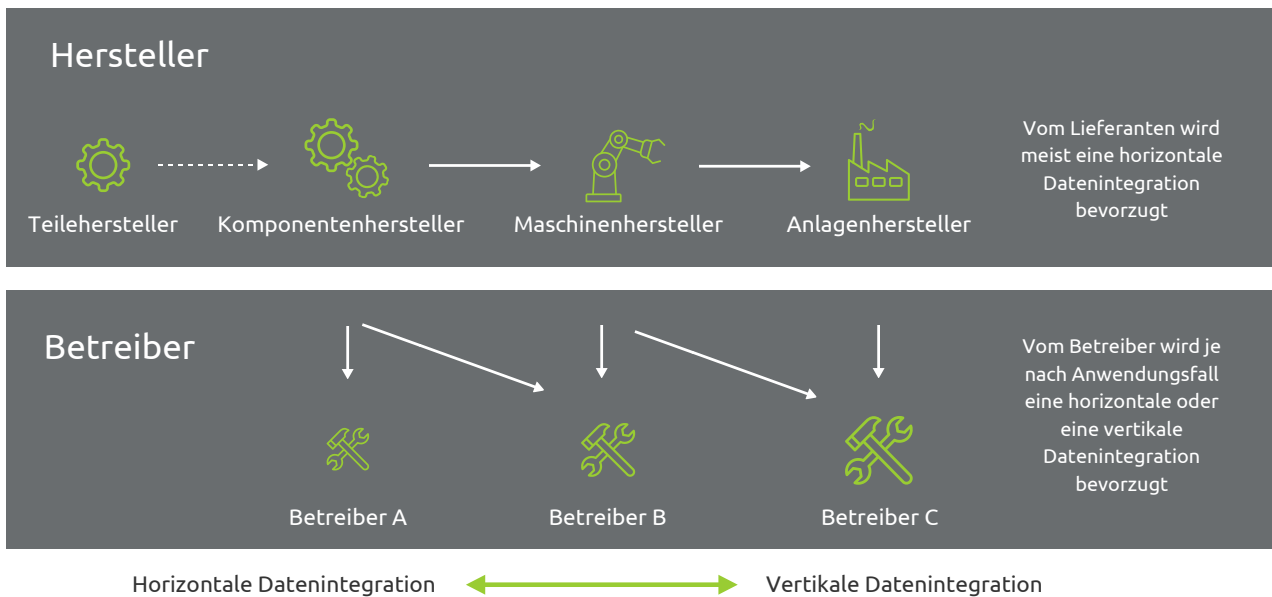


Abb. 2 Vertikale vs horizontale Datenintegration



Grundsätzlich gilt: Wer sein physisches Produkt eher an einen weiteren Hersteller verkauft, sollten eher auf eine *horizontale Datenintegration* setzen. Denn ein Anlagenhersteller, der zehn unterschiedliche Maschinen in seine Anlage integriert, möchte keine zehn Gateways verwalten, die eigenständig Daten in die Cloud schicken. Der Anlagenhersteller möchte vielmehr auf einfache und standardisierte Art und Weise die Daten der unterschiedlichen Maschinen über einen einzigen Industrie-PC in die Cloud schicken.

Nehmen wir als Beispiel den Maschinenhersteller:

Verkauft dieser eher an einen Betreiber, der mithilfe seiner Maschine ein Produkt in

größeren Stückzahlen produziert und dieses wiederum an seine Kunden verkauft? Oder ist der direkte Kunde eher ein Anlagenhersteller, der die Maschine des Maschinenherstellers - neben vielen weiteren Maschinen - in seine Anlage integriert?

Verkauft der Komponenten-, Maschinen- oder Anlagenhersteller sein physisches Produkt hingegen an einen Betreiber, kommt es stark darauf an, welchen Stellenwert das Produkt in der Gesamtproduktion seines Kunden einnimmt.

Ist das Produkt Kern der Wertschöpfung beim Kunden und hat damit einen hohen Anteil an der Gesamtproduktion des Kunden, wird dieser eher den vertikalen Integrationsansatz bevorzugen, da er überhaupt keinen eigenen Industrie-PC nutzt. Deckt das Produkt jedoch nur einen geringen Teil der Wertschöpfung des Kunden

ab, gilt dasselbe Argument wie beim Anlagenbauer: Der Kunde möchte keine zehn Gateways verwalten, die eigenständig Daten in die Cloud schicken.

Die passende Art der Datenintegration ist also stark von der Art der Kunden des Komponenten-, Maschinen- oder Anlagenherstellers abhängig. Außerdem gilt es, die folgenden Vorteile der beiden Integrationsarten zu bedenken. Dabei sind die Vorteile der *vertikalen Integration* auch als Nachteile der *horizontalen Integration* zu verstehen und vice versa.

Vorteile der *vertikalen Integration*:

- Vollständige Kontrolle über den Datenfluss vom Sensor bis in die Cloud. Möchten der Komponenten-, Maschinen- oder Anlagenhersteller beispielsweise die Abtastrate seines Vibrationssensors von zwei Kilohertz auf zehn Kilohertz erhöhen

(natürlich muss der Sensor dazu in der Lage sein) kann er dies über ein Softwareupdate ermöglichen.

- Sicherheit, dass das digitale Produkt beim spezifischen Kunden nicht am Internetzugang des Gateways scheitert, da alles aus einer Hand bereitgestellt wird.
- Keine Kompatibilitätsprobleme bei unterschiedlichen Industrie-PCs.

Vorteile der *horizontalen Integration*:

- Keine Kosten für ein Gateway, das den Preis eines digitalen Produktes ggfs. deutlich erhöht.
- Da der Kunde selbst dafür verantwortlich ist, dass die Daten in die Cloud geroutet werden, (man kann durchaus Anreize dafür schaffen, dass der Kunde dies auch macht) wird der Hersteller nicht zum Verantwortlichen bei Verbindungsproblemen gemacht.

1.3 Hardware-Komponenten und Übertragungsprotokolle zur Ermöglichung von Konnektivität

Nachdem im vorangegangenen Kapitel vor allem auf strategische Aspekte der Datenintegration eingegangen wurde, steht in diesem Kapitel und dem Folgekapitel die Operationalisierung im Mittelpunkt.

Dazu werden in diesem Kapitel sowohl die benötigten Hardware-Komponenten, als auch deren Kommunikationsprotokolle betrachtet.

Auf die Frage, welche Hardware es eigentlich braucht, um Cloud-Konnektivität zu ermöglichen, hat jeder Integrator seine eigene Antwort. Insbesondere die sehr großen Player versuchen i.d.R. die gesamte Strecke mit Geräten der Eigenmarke abzudecken. Bei Siemens erstreckt sich diese Strecke der Eigengeräte vom Sensor komplett bis in die Cloud mit Mindsphere, der Siemens IoT Cloud.

Wir bei esentri sind in der Softwareentwicklung der Meinung, dass offene Schnittstellen dafür sorgen, dass sich einzelne Anbieter auf Kernthemen fokussieren, diese dadurch meistern und außerdem auf sich ändernde Bedingungen wie z.B. neu aufkommende Standards schnell reagieren können. Auch bei der Hardware folgen wir diesem Ansatz, weshalb in der folgenden Abbildung Geräte verschiedener

Hersteller abgebildet sind, die über offene Standards miteinander kommunizieren und so eine moderne und flexible Cloud-Anbindung mit geringem Vendor Lock-in ermöglichen.

Die nachfolgende Abbildung gibt einen Überblick über die Gerätekategorien und deren Interaktionen und illustriert einzelne Beispielgeräte der jeweiligen Kategorie.

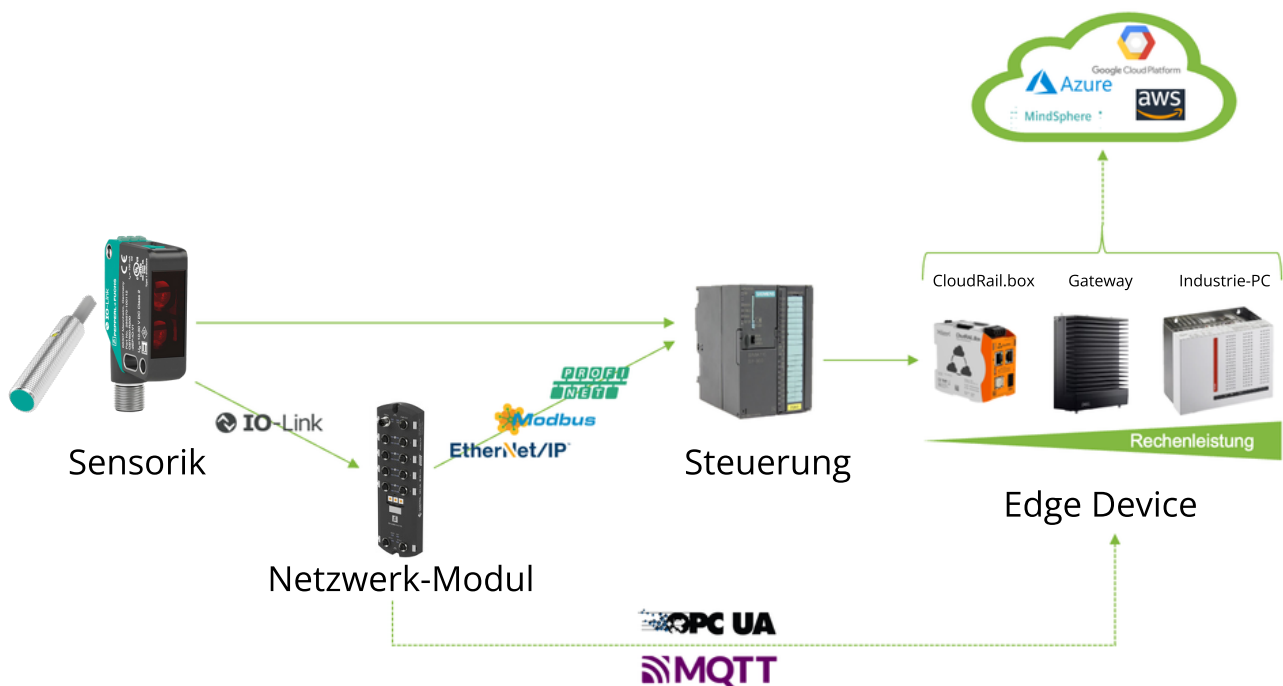


Abb. 3 Hardwarekomponenten und Übertragungsprotokolle (Sensorik und Netzwerk-Modul sind originalgetreue Produkte von Pepperl+Fuchs)

Ganz links in der Abbildung befindet sich die **Sensorik**, welche die Messwerte direkt an den Maschinen aufzeichnet. In dieser Abbildung sind beispielhaft zwei Sensoren vom Sensorhersteller und esentri-Partner Pepperl+Fuchs dargestellt, nämlich ein induktiver Näherungsschalter und ein Reflektionslichttaster - beides Sensoren, die zur Detektion von Objekten dienen. Die Bandbreite an verfügbaren Sensoren ist dabei riesig; detaillierte Informationen zu einzelnen Sensoren sind auf der Website unseres Partners Pepperl+Fuchs zu finden.

Die nächste Gerätekategorie in der obenstehenden Abbildung sind die **Netzwerk-Module**, die eine einfache Anbindung von Sensoren und Geräten an übergeordnete Ethernet- oder Industrial-Ethernet-Netzwerke ermöglichen. In der oberen Abbildung ist beispielhaft ein IO-Link-Master der Generation ICE3 von Pepperl+Fuchs dargestellt. An diesen IO-Link-Master können bis zu 8 Sensoren per Kabel angeschlossen werden. Zur Übertragung der Sensorsignale ist IO-Link das dominierende Kommunikations-

system, was in der oben stehenden Abbildung ebenfalls angedeutet ist. Um die Daten weiter an eine Steuerung zu übertragen, konkurrieren mehrere Ethernet-Protokolle wie Profinet, Modbus TCP, EtherNet/IP oder EtherCAT.

Der dargestellte IO-Link-Master der Generation ICE3 von Pepperl+Fuchs beherrscht beispielsweise die Protokolle Profinet und Modbus TCP. Es finden sich am Markt jedoch auch zahlreiche Netzwerk-Module, die speziell für ein bestimmtes Ethernet-Protokoll ausgelegt sind.

Die nächste Gerätekategorie in Abbildung 3 ist die **Steuerung**. Während im Englischen meist von einer PLC (Programmable Logical Controller) gesprochen wird, hat sich im Deutschen der Begriff SPS (Speicherprogrammierbare Steuerung) durchgesetzt. In der Abbildung ist eine Siemens Simatic S7 Steuerung abgebildet. Eine solche Steuerung befindet sich typischerweise in den Schaltschränken von Produktionshallen.

Nun stellt sich die Frage, warum überhaupt das Netzwerk-Modul notwendig ist, wenn die Sensorik auch direkt mit der Steuerung interagieren kann. Dies ist zwar grundsätzlich möglich, jedoch ist eine Analogbaugruppe (Klemme) nötig, um das analoge Sensorsignal in ein digitales Signal zu übertragen. Steht der Schaltschrank nicht in unmittelbarer Nähe zur Sensorik, ist dies allein aufgrund der Verkabelung nicht praktikabel. Es wird also ein dezentraler Teilnehmer direkt im Anlagenfeld benötigt. Vom Anbieter Siemens könnte das z.B. eine ET 200SP sein oder eben ein IO-Link-Master.

Letztere Variante benötigt keinen Schaltschrank, lässt sich einfacher verkabeln, ermöglicht die Parametrierung von Sensoren und vereinfacht den Sensortausch, weshalb der Zukunftstrend zu dieser Variante tendiert. Da die Netzwerkmodule wie der dargestellte IO-Link-Master der Generation ICE3 in der Lage sind, alle Konfigurationen zu speichern, ist auch eine eigenständige Verwendung ohne übergeordnete SPS möglich. Detaillierte Anwendungsszenarien können bei unserem Sensorpartner eingesehen werden.

Als Nächstes stellt sich die Frage, wie die Daten neben der Interaktion mit der Steuerung zusätzlich in die Cloud übertragen werden können. Wie Abbildung 3 zeigt, erfolgt dies über die Gerätekategorie **Edge Devices**. Edge Device ist ein Sammelbegriff für lokale Rechenkapazitäten. Teilweise wird bereits bei einem einfachen Sensor von Edge Device gesprochen.



Im Kontext dieses Whitepapers sind jedoch solche Recheneinheiten gemeint, die ausreichend Prozessor- und Speicherleistung, etc. besitzen, um eine Linux-Distribution zu betreiben.

Wie die Abbildung zeigt, kann es sich dabei um kleine Recheneinheiten handeln wie die Cloudrail Box - oder dem in der Experimentierumgebung beliebten Raspberry Pi - oder aber um sehr leistungsstarke Gateways oder sogar Industrie-PCs.

Doch wie gelangen die Daten zu den Edge Devices? Wie Abbildung 3 zeigt, gibt es dafür insbesondere zwei Wege:

1. Nutzung der Multilink Funktion vom IO-Link Netzwerk-Modul

Durch die Multilink Funktion des Netzwerk-Moduls können neben der Kommunikation mit einer übergeordneten SPS z.B. über das Profinet Protokoll gleichzeitig Daten per OPC UA oder MQTT an ein *Edge Device* weitergereicht werden. Wie der gestrichelte Pfeil andeutet, erfolgt diese Übertragung kabellos. Details zu den Protokollen OPC UA und MQTT werden im Folgekapitel erläutert.

2. Übertragung der Daten aus der Steuerung an ein Edge Device

Je nach Steuerung muss an dieser Stelle viel Arbeit für Übersetzungen zwischen Protokollen geleistet werden. Dieser Weg ist also grundsätzlich möglich, wenn jedoch sowieso schon Netzwerk-Module im Einsatz sind (aus den oben genannten Gründen) kann der erstgenannte Weg der einfachere sein. An dieser Stelle ist jedoch zu erwähnen, dass neue Generationen von Steuerungen bereits ebenfalls MQTT- oder OPC UA-fähig sind und damit auch dieser Weg durchaus sinnvoll sein kann.

In der Kategorie der Edge Devices ist abschließend der Anbieter CloudRail zu erwähnen, der sich hier in den letzten Jahren etabliert hat. Der Anbieter ermöglicht Plug&Play Anbindung von Sensoren (über Netzwerkmodule oder SPS) als SaaS (Software as a Service). Durch eine sehr große Auswahl an bereits vorkonfigurierten Sensoren wird die Anbindung beschleunigt. Außerdem unterstützt Cloudrail auch beim Weg von der Cloud in Richtung Edge Device im Bezug auf das Device Management.

Um diese Vorteile nicht nur auf dem Cloudrail eigenen Gerät anzubieten und stattdessen auch leistungsstärkere Gateways oder Industrie-PCs nutzen zu können, bietet CloudRail außerdem Cloudrail.OS, welches die Cloudrail Software in einem Container betreibt.

Möchte ein produzierendes Unternehmen seine eigene Produktionslinie digitalisieren und über Cloud Services Monitoring betreiben und Daten analysieren, überwiegt der Vorteil einer schnellen Cloudanbindung von Industriesensoren oft den Nachteil der Abhängigkeit von einem weiteren Drittanbieter. Möchten das produzierende Unternehmen jedoch sein eigenes Produkt digitalisieren, sollten die Konsequenzen einer solchen Abhängigkeit gut untersucht sein, um die Entscheidung für oder gegen einen solchen Anbieter bewusst treffen zu können.

Die letzte Kategorie in der Abbildung beschreibt die **Cloud-Anbieter**. Alle Anbieter haben gemeinsam, dass die Daten über ein Edge Device übertragen werden. Um die Sicherheit der Verbindung zu gewährleisten und Missbrauch zu vermeiden, werden in der Cloud Zertifikate erstellt und an die Edge Devices weitergegeben. Nur wenn sich Cloud Account und Edge Device gegenseitig erfolgreich authentifizieren, dürfen Daten ausgetauscht werden. Der Datenaustausch zwischen Edge Device und Cloud erfolgt im Industriekontext fast ausschließlich über MQTT. Welche Möglichkeiten sich durch die Weiterverarbeitung der Daten in der Cloud eröffnen, wird in den beiden Whitepapern "Data Platform" und "KI-Services" im Detail erklärt.

1.4 Die wichtigsten Übertragungsprotokolle im Industriekontext

Im vorangegangenen Kapitel wurden insbesondere die Protokolle MQTT sowie OPC UA bereits mehrfach erwähnt und deren Bedeutung im Industrial Internet of Things herausgestellt.

In diesem Kapitel sollen diese Protokolle im technischen Detail analysiert werden. Dazu werden die Protokolle zunächst in den Kontext des TCP/IP Modells eingeordnet. Das TCP/IP-Modell ist ein Netzwerk-Referenzmodell, das aus vier Schichten besteht: der *Netzwerkschicht*, der *Internetschicht*, der *Transportschicht* und der *Anwendungsschicht*.

- Die **Netzwerkschicht** stellt die physische Verbindung zwischen den Geräten her und regelt den Datenaustausch über

diese Verbindung. Die bekanntesten Standards dieser Schicht sind Ethernet und Wireless.

- Die **Internetschicht** sorgt für die Übertragung von Datenpaketen zwischen Netzwerken. Das Internet Protocol (IP) ist das wichtigste Protokoll in dieser Schicht.
- Die **Transportschicht** ist für die Verbindung von Endpunkten und den Datenfluss zwischen ihnen zuständig. Hier werden die Protokolle TCP und UDP verwendet.
- Die **Anwendungsschicht** enthält Protokolle für den Datenaustausch zwischen Anwendungen, wie z.B. HTTP für Web-Browsing oder SMTP für E-Mail. Auch MQTT und OPC UA sind dieser Schicht zuzuordnen.

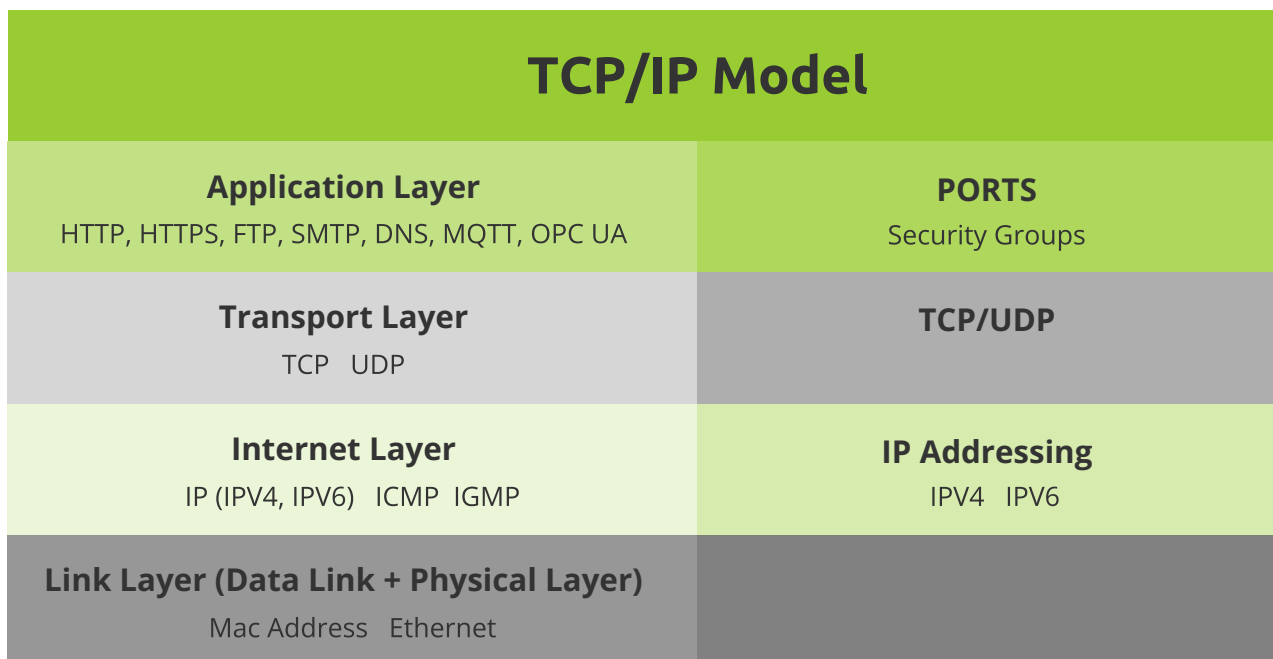


Abb. 4 TCP/IP Model

4.1 MQTT (Message Queuing Telemetry Transport)

MQTT ist ein leichtgewichtiges Protokoll, zertifizierter OASIS- und ISO-Standard (ISO/IEC 20922:2016) und wird oft in IoT-Anwendungen eingesetzt, bei denen viele Geräte miteinander kommunizieren müssen. Es eignet sich insbesondere in Fällen, bei denen Clients einen schlanken Code-Footprint brauchen und mit unzuverlässigen Netzwerken verbunden sind.

MQTT Clients

Sensors

MQTT Clients

Cloud Server

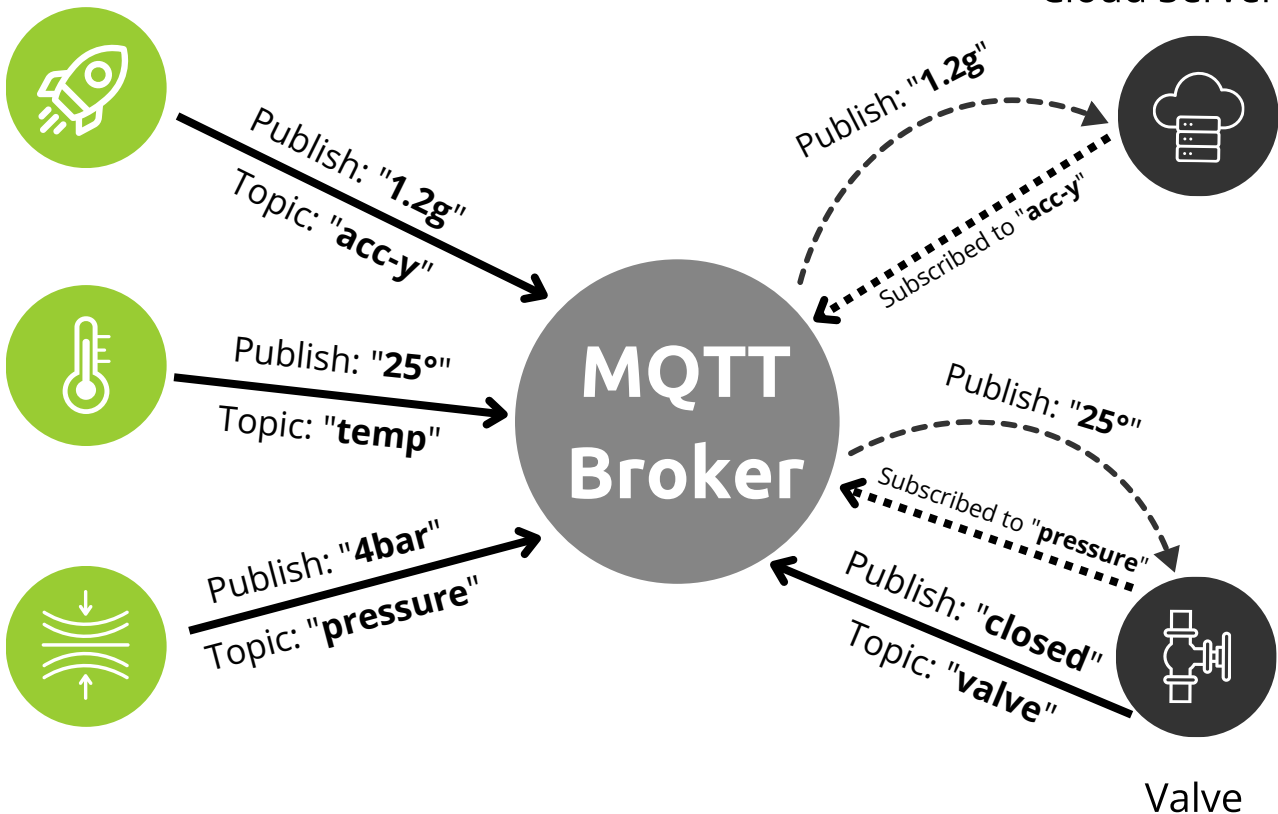


Abb. 5 MQTT Publish-Subscriber-Modell

Die obenstehende Abbildung visualisiert das Publish-Subscriber-Modell, das MQTT zugrunde liegt: Ein Gerät (**Client**) sendet Nachrichten (sogenannte "**Publishes**") an einen **Broker**, der diese an alle Geräte (**Clients**) weiterleitet, die sich für diese Nachrichten (sogenannte "**Topics**") interessieren. Jeder Client kann entweder ein Publisher, ein Abonnent oder beides sein. In der obigen Abbildung sind die Sensoren Publisher, das Valve ist sowohl Publisher als

auch Abonnent und der Cloud Server ist reiner Abonnent. Dadurch können viele Geräte gleichzeitig und effizient miteinander kommunizieren, ohne dass jedes Gerät direkt mit jedem anderen kommunizieren muss. Der große Vorteil liegt in der Entkoppelung von Publishern und Subscribern: Es müssen keine unzähligen Einzelverbindungen hergestellt werden, sondern nur die zum Broker, wodurch die Kommunikation asynchron stattfinden kann.

Im Folgenden sind die wichtigsten Merkmale der MQTT Spezifikation zusammengefasst:

➤ **Topic:** Jede Nachricht besteht aus einem Topic und einer Payload. Das Topic ist das zentrale Element, womit der Broker Nachrichten filtern und an die passenden Abonnenten verschicken kann. Topics können dabei mehrere Ebenen umfassen, wie z.B. `werk1/maschine4/data/druck`. Jede Topic-Ebene wird dabei von einem Schrägstrich getrennt. Durch Platzhalter können Informationen gebündelt werden: z.B. `werk1/+ /data/druck`, dass alle Druckwerte von allen Maschinen im Werk 1 abonniert werden.

➤ **Quality of Service:** MQTT bietet verschiedene QoS-Level, die es ermöglichen, die Qualität und Zuverlässigkeit der Nachrichtenübertragung entsprechend den Anforderungen der Anwendung einzustellen. Es gibt insgesamt drei QoS-Level in MQTT:

- **QoS 0:** Auch bekannt als "At most once" (Höchstens einmal). Bei diesem Level wird die Nachricht nur ein Mal gesendet, ohne dass eine Bestätigung des Empfangs erfolgt. Es wird keine Garantie dafür gegeben, dass die Nachricht den Empfänger erreicht oder in der richtigen Reihenfolge ankommt. Dieses Level hat die geringste Zuverlässigkeit, bietet jedoch auch die geringste Netzwerkbelastung.
- **QoS 1:** Auch bekannt als "At least once" (Mindestens einmal). Hierbei wird die Nachricht mindestens ein Mal gesendet und der Empfänger sendet eine Bestätigung (ACK) an den Sender, um den erfolgreichen Empfang zu

bestätigen. Wenn der Sender keine ACK erhält, wird die Nachricht erneut gesendet. Dadurch wird sichergestellt, dass die Nachricht zumindest ein Mal ankommt, es besteht jedoch die Möglichkeit, dass sie mehrmals empfangen wird.

- **QoS 2:** Auch bekannt als "Exactly once" (Genau einmal). Dieses Level bietet die höchste Zuverlässigkeit. Die Nachricht wird genau ein Mal gesendet und der Empfänger sendet eine Bestätigung an den Sender. Darüber hinaus wird ein zweistufiger Handshake verwendet, um sicherzustellen, dass die Nachricht nur einmalig empfangen wird, selbst wenn es zu Netzwerkproblemen oder Ausfällen kommt. QoS 2 hat die höchste Netzwerkbelastung, da zusätzlicher Overhead für die Bestätigungen und den Handshake erforderlich ist.

Die Wahl des QoS-Levels hängt von den Anforderungen der Anwendung ab. Wenn beispielsweise eine geringe Netzwerkbelastung wichtiger ist als eine 100%ige Zuverlässigkeit, kann QoS 0 verwendet werden. Wenn jedoch eine zuverlässige Zustellung von Nachrichten erforderlich ist, sollten die *QoS-Levels 1 oder 2* verwendet werden.



Es ist wichtig zu beachten, dass der QoS-Level für jede Nachricht separat festgelegt werden kann. Dies ermöglicht eine granulare Kontrolle über die Zuverlässigkeit und Netzwerkbelastung je nach den spezifischen Anforderungen und Prioritäten der Nachrichten in der Anwendung.

➤ **Session Awareness:** Hierbei handelt es sich um die Fähigkeit des MQTT-Brokers, den Zustand einer Verbindung zwischen einem Client und dem Broker beizubehalten. Es ermöglicht die Wiederherstellung einer Verbindung und den nahtlosen Austausch von Nachrichten, wenn die Verbindung zwischen dem Client und dem Broker vorübergehend unterbrochen wurde. Session Awareness wird durch das „Clean Session“-Flag in der Verbindungsanforderung des MQTT-Clients gesteuert.

Es gibt zwei Möglichkeiten:

1. *Clean Session = 0:* Hierbei handelt es sich um eine persistente Sitzung. Der MQTT-Broker speichert den Zustand der Verbindung, einschließlich Abonnements und Nachrichtenwarteschlangen, wenn der Client offline geht oder die Verbindung unterbrochen wird. Wenn der Client die Verbindung wiederherstellt, nimmt der Broker den vorherigen Zustand wieder auf und der Client kann seine Abonnements sowie Warteschlangen fortsetzen, als ob die Verbindung nie unterbrochen wurde.
2. *Clean Session = 1:* Hierbei handelt es sich um eine nicht persistente Sitzung. Der MQTT-Broker verwirft den Zustand der Verbindung, wenn der Client offline geht oder die Verbindung unterbrochen wird. Wenn der Client die Verbindung wiederherstellt, beginnt er mit einem sauberen Zustand und der Broker hat keine Kenntnis von den vorherigen Abonnements oder Warteschlangen des Clients.



Die Session Awareness ist besonders nützlich, wenn es um die Kommunikation in intermittierenden Netzwerken geht, in denen die Verbindung zwischen Client und Broker nicht dauerhaft aufrechterhalten werden kann. Indem der Broker den Zustand der Verbindung beibehält, können Clients nahtlos Nachrichten austauschen, sobald die Verbindung wiederhergestellt ist, ohne dass Informationen verloren gehen.

Weitere spannende Spezifikationsmöglichkeiten, die mit der Veröffentlichung von MQTT 5 eingeführt wurden, sind unter anderem **Shared Subscriptions, Statuscodes, sicherheitsbedingte Kontrollpakete, Topic-Aliases** und **Ablaufdatum von Nachrichten**.


Diese würden jedoch den Rahmen dieses Whitepapers sprengen und können bei Bedarf im Detail in der Spezifikation nachgelesen werden.

4.2 OPC UA

OPC UA steht für „*Open Plattform Communication Unified Architecture*“ und ist ein Industriestandard für die Kommunikation und den Datenaustausch in der Automatisierungstechnik sowie im Industrial Internet of Things (IIoT). Es handelt sich um einen offenen, plattformunabhängigen und herstellernerneutralen Standard, der die Interoperabilität zwischen verschiedenen Geräten, Systemen und Anwendungen ermöglicht. OPC UA ist Nachfolger von OPC mit der Zielsetzung der Plattformunabhängigkeit und ist damit unabhängig von Microsoft und deren COM/DCOM.

Der große Mehrwert von OPC UA liegt in den **Daten- und Informationsmodellen**: Technische Arbeitsgruppen definieren sogenannte Companion Specifications, die auf branchenübergreifende Harmonisierung und Kommunikation aller Geräte hinarbeiten. So haben beispielsweise zahlreiche Hersteller von Spritzgussmaschinen in einem gemeinsamen Konsortium ein einheitliches Datenmodell für Spritzgussmaschinen ausdefiniert und die sogenannte **Euromap77-Schnittstelle** veröffentlicht, welche mit einer fertigen Companion Specification gleichzusetzen ist.

Für einen Maschinenbetreiber bedeutet dies folgendes: Eine Spritzgussmaschine, welche die Euromap77-Schnittstelle implementiert, verfügt über ein einheitliches Datenmodell und die Daten sämtlicher Spritzgussmaschinen, die diese Schnittstelle implementieren, können einheitlich abgefragt werden.

 Wie sieht ein solches **Datenmodell** im Detail aus?

Die OPC-UA-Datenmodellierung basiert auf zwei grundlegenden Elementen: *Knoten* und *Referenzen*. Jede Dateneinheit wird als Knoten bezeichnet und diese Knoten sind über Referenzen miteinander verbunden. Es gibt verschiedene Typen von Knoten und Referenzen. Beispielsweise kann ein Knoten den Typ „Object“, „Variable“, „Method“, „ObjectType“ oder „DataType“ haben. Die Referenzen haben ebenfalls Typen und sind in hierarchischen und nichthierarchischen Gruppen organisiert. Die Knoten und Referenzen sind innerhalb eines Adressbereichs platziert, und jeder Knoten kann über eine eindeutige Knoten-Kennung, die als „Nodeld“ bezeichnet wird, angesprochen werden. Die Nodeld fungiert als Adresse, unter der die entsprechenden

Dateninformationen des Knotens abgerufen werden können.

 Was ist unter einem **Informationsmodell** zu verstehen?

Informationsmodelle stellen Standardknotendefinitionen bereit, um eine bessere Harmonisierung zwischen Geräten zu ermöglichen. Es gibt verschiedene Informationsmodelle, von denen die OPC Foundation grundlegende Modelle wie „Data Access (DA)“, „Historical Data Access (HDA)“, „Alarms and Conditions“ und „Programs“ definiert hat. Diese Modelle geben Auskunft über die unterstützten Funktionen einer OPC-UA-Anwendung. Das *Data Access (DA)-Modell* ermöglicht den Standardzugriff auf Prozess- und Parameterdaten. Das *Historical Data Access (HDA)-Modell* erweitert die Funktionalität, indem es das Speichern von mehreren Messungen desselben Knotens für statistische Zwecke ermöglicht. Das Alarms and Conditions-Modell ermöglicht das Senden von Ereignissen bei Notfallbenachrichtigungen, und das Programs-Modell ermöglicht die Definition einfacher Zustandsmaschinen. Die Skalierbarkeit von OPC UA ermöglicht es, nur das Data Access-Informationsmodell zu implementieren, was besonders für Geräte mit begrenztem Speicher und Rechenleistung vorteilhaft ist. Diese Informationsmodelle tragen zur Harmonisierung von Geräten bei, da sie dieselben Informationen auf denselben Zugriffspunkten (Nodelds) implementieren. Dadurch wird der Austausch von Geräten, selbst von verschiedenen Anbietern, vereinfacht.

Nachdem die Grundlagen der Daten- und Informationsmodelle erklärt sind stellt sich die Frage nach der Art der Kommunikation via OPC UA.

Dabei ist zwischen dem *Request/Response* und dem *Pub/Sub* Muster zu unterscheiden:

Request/Response

Das Request/Response-Muster ist ein gängiges Kommunikationsmuster in Client/Server-Architekturen. Es basiert auf einer einfachen Funktionsweise: Der Client sendet eine Anfrage an den Server, und der Server antwortet mit einer entsprechenden Antwort. Typischerweise wird dies synchron umgesetzt, was bedeutet, dass der Client auf die Antwort des Servers wartet, bevor er weitermacht.

Pub/Sub

Seit 2018 wurde OPC UA um die Pub/Sub Kommunikation erweitert. Dabei kommt u.a. MQTT zum Einsatz und asynchrone Kommunikation, wie bereits oben im Publish-Subscriber-Modell beschrieben, wird ermöglicht. Dies wird oft als "OPC UA over MQTT" bezeichnet.

Die nachfolgende Abbildung zeigt eine OPC UA Applikation, die mit einem OPC UA Client im *Request/Response Modell* kommuniziert und gleichzeitig im *Pub/Sub System* Daten an einen MQTT Broker published.

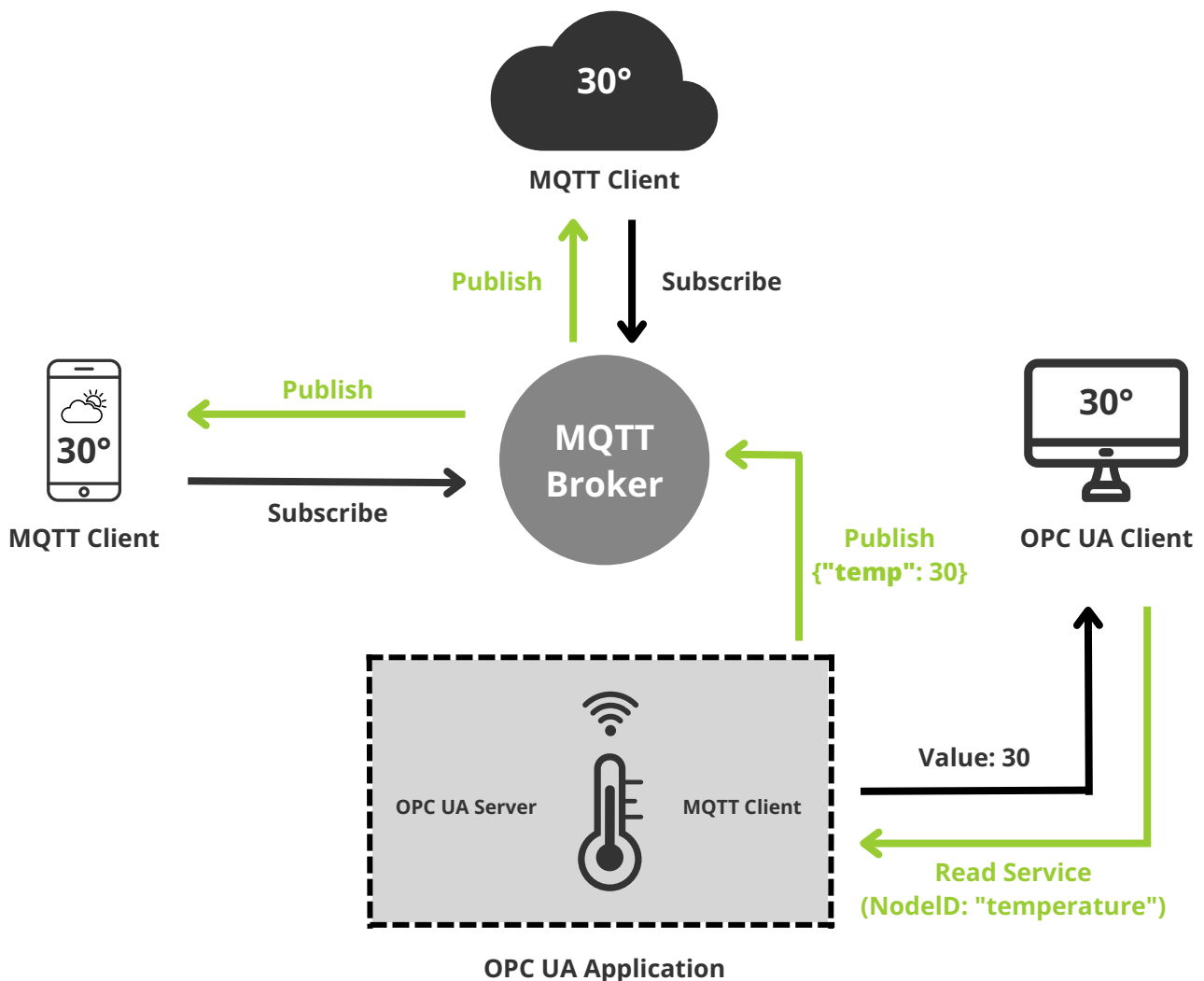


Abb. 6 Kommunikation einer OPC UA Applikation

Frägt der OPC UA Client den Wert der NodeID "temperature" an, erhält er eine Antwort gemäß Request/Response Modell. Gleichzeitig veröffentlicht der Temperatursensor in regelmäßigen Abständen seine Werte an einen MQTT Broker. Die beiden Kommunikationswege sind dabei mithilfe zweier separater TCP-Verbindungen realisiert.

4.3 Protokollvergleich

Nachdem die beiden Protokolle im Detail erklärt wurden, sollen sie in diesem Unterkapitel miteinander verglichen werden, um ein Gefühl für deren optimale Einsatzbereiche zu entwickeln.

Dabei werden die Kriterien *Interoperabilität*, *Echtzeitverhalten*, *Implementierung* und *Use Cases* untersucht.



Unter *Interoperabilität* ist sowohl die Maschine-Maschine Kommunikation als auch die Mensch-Maschine Kommunikation zu verstehen.

MQTT

Interoperabilität

Für die Maschine-Maschine Kommunikation ist das Protokoll unzureichend, da keine semantische Standardisierung definiert ist. Die falsche Schreibweise eines Topics oder das Senden von Daten in einem anderen Format als gewünscht wird nicht als Fehler betrachtet. Eine Maschine-Maschine Kommunikation ist also nur möglich wenn zuvor alle Topics und Payloads von Menschenhand festgelegt werden.

Für eine Mensch-Maschine Kommunikation sollte der Mensch zunächst in einem Handbuch nachschlagen, welche Topics verfügbar sind und wie ihre Daten strukturiert sind, bevor er mit der Maschine interagiert. Die Maschine kann nicht selbst festlegen, welche Topics abonniert werden können.

OPC UA

OPC UA ist in der Lage, Maschine-Maschine-Interaktionen zu ermöglichen. Es bietet eine Vielzahl von Funktionen zur syntaktischen und semantischen Interoperabilität. *Syntaktisch* verwenden alle Geräte die gleichen Typen und Dienste für die Kommunikation. Sie verwenden Informationsmodelle, die in Nodeset-Dateien im gleichen Format gespeichert werden. Diese Dateien enthalten eine Sammlung von Knoten und Referenzen, die die Daten gemäß den OPC UA-Modellierungsregeln definieren und in einem maschinenlesbaren Format bereitstellen. *Semantisch* arbeiten verschiedene Arbeitsgruppen daran, die Parameter- und Prozessdatennamen, die Verfügbarkeit von Methoden, Ereignissen usw. zu harmonisieren, um sicherzustellen, dass unterschiedliche Geräte ähnliche Namen und Verhaltensweisen aufweisen.

Durch die Verwendung von Visualisierungstools zur Darstellung des Informationsmodells wird außerdem eine Mensch-Maschine-Interaktion ermöglicht.

Echtzeitverhalten

MQTT ist ein sehr schlankes Protokoll und erreicht dadurch sehr kurze Transportzeiten. Falls der Broker erreichbar ist (kann durch Redundanz gewährleistet werden) und der Subscriber mit dem Publisher verbunden ist, kann die theoretisch versprochene beinahe Echtzeit erreicht werden.

Durch die Pub/Sub Erweiterung kann auch OPC UA im Echtzeitbereich mithalten. Eine interessante Entwicklung ist für OPC UA TSN (Time-Sensitive Networking) zu beobachten. TSN bezeichnet dabei eine Reihe von Unterstandards des Ethernet-Standards, wodurch die Datenübertragung über Standard-Ethernet echtzeitfähig wird. Teile der Automobilbranche setzen bereits auf diesen Standard.

Implementierung

- Sehr geringer Speicherbedarf
- Geringe Komplexität der Implementierung

-
- Verhältnismäßig großer Speicherbedarf, der jedoch etwas reduzierbar ist durch Begrenzung der möglichen TCP-Verbindungen sowie Beschränkung der Informationsmodelle auf Data Access (DA)
 - Hohe Komplexität der Implementierung, v.a. wenn ein vollständiges Informationsmodell aufgebaut werden soll

Use Cases

- Einfache, kompakte Lösungen
- Geringer Leistungs- und Speicherbedarf

-
- Standardisierung und Harmonisierung gemeinsam genutzter Daten
 - Einfache Austauschbarkeit von Geräten durch die Nutzung identischer Informationsmodelle
 - Wenn eine Mischung zwischen Request/Response und Pub/Sub Kommunikation benötigt wird

1.5 Zusammenfassung und Ausblick

In diesem Whitepaper wurde das Thema *Connectivity* aus verschiedenen Flugebenen und Perspektiven beleuchtet. Dazu wurde zunächst auf der höchsten Flugebene erläutert, warum es überhaupt sinnvoll ist, lokale Maschinendaten in die Cloud zu bringen. Danach wurde auf die verschiedenen strategischen Arten der Datenintegration sowie deren Vor- und Nachteile eingegangen. Die darauffolgende Hardware- und Software-Überblick hat gezeigt, welche Komponenten und Protokolle notwendig sind, um Konnektivität zu ermöglichen. Die relevantesten Protokolle wurden dann im Detail analysiert und gegenübergestellt.

Insgesamt ist das Thema Konnektivität ein riesiges Feld und der Weg zur tatsächlich intelligenten Fabrik, bei der Echtzeit-Datenanalysen zur Selbstüberwachung und Optimierung gesamter Fabriken genutzt werden, ist noch recht weit. Die nötigen Visionen und Geschäftsmodelle sind jedoch bereits vorgedacht und auch die notwendige Basis Innovation bzgl. Hard- und Software ist bereits vorhanden. Es bleibt mit Spannung zu betrachten, ob sich neben den in diesem Whitepaper genannten Technologien noch weitere Ansätze durchsetzen werden oder ob das weitere Rennen eher in den nachgelagerten Datenplattformen und vor allem KI-Services stattfindet.

2 DATA PLATFORMS

2.1 Der Plattform-Ansatz für die Welt der digitalen Daten

Plattformen für Software sind aus unserer heutigen Welt nicht mehr wegzudenken. Sie bilden die Basis zur Erstellung und Ausführung von Applikationen, Services und anderen Softwarekomponenten und bündeln diverse Komponenten und Tools zur Softwareentwicklung oder deren Nutzung.

Durch den Einsatz von Softwareplattformen wird Entwickler:innen die Möglichkeit gegeben, auf bereits vorhandene Funktionen aufzubauen und sich auf die Umsetzung ihrer eigenen Ideen und Lösungen zu konzentrieren, anstatt grundlegende technische Details neu zu erfinden. Die Verwendung einer Plattform kann damit Zeit- und Kosteneinsparungen ermöglichen, indem sie Entwicklungszyklen verkürzt und die Wartung erleichtert. Ein prominentes Beispiel sind Appstores auf mobilen Endgeräten, die es Anwender:innen ermöglichen diverse Applikationen durch nur wenige Klicks auf ihrem Gerät zu installieren, aber auch Entwickler:innen bei der Bereitstellung neuer Applikationen unterstützen.

Mit der zunehmenden Erfassung und Verarbeitung von Daten, insbesondere im industriellen Umfeld, steigt der Bedarf an vergleichbaren Plattformen für die Sammlung und Verarbeitung digitaler Daten. Dabei meint der oft synonym für spezifische technologische Lösungen verwendete Begriff „Datenplattform“ per se ein Konglomerat von Technologien und Tools wie Datenbanken, Aufbereitungstools und Analysewerkzeugen von Daten. Eine solche Datenplattform dient

der Sammlung, Transformation und Vereinheitlichung digitaler Daten aus diversen Quellen und ermöglicht so, dass Daten in aufbereiteter Form direkt für Endanwender:innen verfügbar gemacht werden oder aber durch Softwareapplikationen oder Business Intelligence Lösungen genutzt werden können.

Demnach kann eine Datenplattform also mehrere konkrete Mehrwerte bieten. Durch einheitliche Möglichkeiten zum Datenzugriff wird sichergestellt, dass jede:r Anwender:in auf die gleiche Weise auf die Daten zugreifen kann, ohne die unterschiedlichen Konfigurationen und Strukturen der verschiedenen Datenquellen kennen zu müssen. Dies ist im Vergleich zu direkten Zugriffen auf verschiedene Purpose-Datenbanken oder Datendumps von enormem Vorteil.

Darüber hinaus werden durch eine Datenplattform viele Analyseabfragen erst möglich, da Purpose-Datenbanken in der Regel nicht für analytische Anfragen ausgelegt sind. Außerdem kann eine Datenplattform dazu beitragen, dass das Produktivsystem nicht zusätzlich belastet wird. Data Scientists müssen sich keine Sorgen machen, dass ihre Analysen das System verlangsamen oder sogar lahmlegen. Ein weiterer Vorteil einer Datenplattform ist, dass sie als Single Source of Truth dient, sodass alle auf die gleichen, zuverlässigen Daten mit gewährleisteter Aktualität zugreifen.

Eine Datenplattform kann dabei diverse Formen und Arten digitaler Daten beinhalten, die sich zum Beispiel anhand ihrer Struktur, ihrer Entstehungsquelle oder nach ihrer Generierungsrate klassifizieren lassen. Eine Unterscheidungsform, die speziell für industrielle Anwendungsfälle sinnvoll ist, ist die Differenzierung nach der Art von Informationen, die sie abbilden: Dabei unterscheidet man zwischen *Zeitreihendaten* und *deskriptiven Daten*, die in den nachfolgenden Abschnitten genauer definiert werden.

Zeitreihendaten

Zeitreihendaten sind eine Sammlung von Beobachtungen, die in bestimmten Zeitintervallen aufgezeichnet werden, in der Regel mit gleichmäßigem zeitlichen Abstand. Sie werden üblicherweise verwendet, um das Verhalten eines Systems oder Prozesses im Laufe der Zeit zu verstehen. In industriellen Anlagen werden z.B. häufig Temperaturen, Drücke, Vibrationen oder Durchflussmengen kontinuierlich erfasst und überwacht. Durch die Analyse dieser Zeitreihendaten können Muster, Trends und Anomalien eines Systems oder Prozesses beobachtet werden, was nachgelagerte Services wie automatische Fehlererkennung, vorausschauende Wartung oder allgemeine Effizienzoptimierungen ermöglicht (siehe Kapitel 3 *KI Services*).

Sofern sie in Echtzeit erfasst, verarbeitet und analysiert werden, können sie Aufschluss über die Leistung und den Betriebszustand von industriellen Prozessen und Anlagen geben. Echtzeit-Datenströme sind in industriellen Umgebungen von entscheidender Bedeutung, da sie es den Betreiber:innen ermöglichen, schnell auf Veränderungen in der Umgebung zu reagieren und proaktive Maßnahmen zu ergreifen, um Probleme zu verhindern, bevor sie auftreten.

Konkrete Beispiele für Zeitreihendaten sind:

Maschinenzustandsdaten: Daten zu verschiedenen Parametern, wie Temperatur, Druck, Feuchtigkeit, Durchflussmenge und Vibration von Maschinenbauteilen oder -anlagen. Die von diesen Sensoren im Laufe der Zeit ermittelten Messwerte können zur Überwachung des Zustands von Anlagen verwendet werden, um potenzielle Probleme zu erkennen, bevor sie kritisch werden. Entscheidend für den Wertgehalt dieser Daten ist die Identifikation aussagekräftiger Messpunkte, was wiederum tiefgehendes Ingenieurwissen über die Maschine beziehungsweise die Anlage erfordert.

Energieverbrauchsdaten: Informationen über den Energieverbrauch in Industrieanlagen. Dies kann Daten über den Strom-, Kraftstoff- und andere Formen des Energieverbrauchs umfassen. Durch die Analyse dieser Daten können Unternehmen Bereiche identifizieren, in denen der Energieverbrauch gesenkt werden kann, was zu Kosteneinsparungen und verbesserter Nachhaltigkeit führt.

Umweltdaten: Industrieanlagen können Daten über Umweltfaktoren wie die Luftqualität, Wasserqualität und Schallpegel erzeugen. Das Sammeln dieser Daten im Laufe der Zeit kann helfen, Trends und Muster zu erkennen, die auf zu behobende Umweltprobleme hinweisen können.

Lieferkettendaten: erfassen die Leistung von Lieferketten, einschließlich Durchlaufzeiten, Versand- und Empfangsaktivitäten und Lagerbestände. Durch die Analyse dieser Daten können Unternehmen Bereiche identifizieren, in denen die Leistung der Lieferkette verbessert werden kann, was zu mehr Effizienz und Kosteneinsparungen führen kann.

Deskriptive Daten

Deskriptive Daten im Kontext der industriellen Analyse können als Daten definiert werden, die die betrieblichen Prozesse und die in industriellen Umgebungen verwendeten Geräte beschreiben. Dazu gehören CRUD-Daten (*Create, Read, Update, Delete*), die über die Erstellung, Änderung und Löschung von Daten in einer Datenbank gesammelt werden. Sowie Betriebsdaten, die sich auf Daten beziehen, die über die Leistung von Industrieanlagen und -prozessen über beispielsweise Schnittstellen, wie Application Programming Interfaces (APIs) gesammelt werden. Hinzu kommen Metadaten, die zusätzliche Informationen über die erfassten Betriebsdaten liefern, wie z.B. deren Datenquelle, Datenqualität und andere relevante Details.

Insgesamt können deskriptive Daten ein breites Spektrum an Informationen abbilden, z.B. Hardwarespezifikationen, Produktionsraten, Ausfallzeiten von Anlagen, Wartungsprotokolle, Maschinenschwellenwerte und andere wichtige Leistungsindikatoren.

Durch die Analyse von deskriptiven Daten können Unternehmen somit Trends, Muster und Anomalien in ihren Abläufen erkennen, was zu einer verbesserten Effizienz, geringeren Kosten und einer höheren Rentabilität führen kann.



Deskriptive Daten spielen auch eine wichtige Rolle bei der prädiktiven und präskriptiven Analyse, bei der historische Daten verwendet werden, um Vorhersagen zu treffen und Maßnahmen für zukünftige Leistungsverbesserungen zu definieren.

Einige Beispiele für deskriptive Daten sind:

Metadaten eines Maschinensensors: Beschreibende Daten, die Informationen über Maschinen und Sensoren liefern, wie z. B. deren Typ, Standort und Konfiguration, die zur Identifizierung und Analyse von Sensordaten verwendet werden.

Metriken für Produktionslinien: Beschreibende Daten zu Produktionsprozessen, wie z.B. Ausgabearten, Qualitätsmetriken und Materialverbrauch, die zur Ermittlung von Trends, Mustern und Leistungsproblemen verwendet werden.

Wartungshistorie: Beschreibende Daten über die Wartung von Geräten und Systemen, einschließlich Aufzeichnungen über Reparaturen, Austausch und vorbeugende Wartung, die dazu dienen, die Leistung zu verfolgen, Probleme zu erkennen und Wartungspläne zu optimieren. Besonders relevant im Hinblick auf den Ansatz der vorausschauenden Wartung.

Betriebs-/Finanzdaten: Maschinenleistung und z.B. Umsatz/Wertströme können Maschinen und Kostenstellen zugeordnet werden. Wartungspläne können anhand der Marktnachfrage optimiert werden. Solche organisatorischen Daten sind notwendig, um bei der Analyse die Werkstatt mit dem Unternehmen zu verbinden.

2.2 Anforderungen und Ziele einer Datenplattform

Eine moderne Datenplattform erfüllt mehrere Ziele für die Verarbeitung und Verwendung von Daten in einem Unternehmen. Daneben gibt es mehrere funktionale und nicht-funktionale Anforderungen, die eine Datenplattform erfüllen sollte. Die wichtigsten Ziele und Anforderungen werden nachfolgend beschrieben.

- **Single Source of Truth:** Eine Datenplattform als "Single Source of Truth" bietet einen zentralen Speicherort für alle Daten in einem Unternehmen und stellt sicher, dass alle Mitarbeitenden mit denselben Daten arbeiten. Sie gewährleistet Konsistenz, Genauigkeit und Zuverlässigkeit von Daten, die für fundierte Geschäftsentscheidungen unerlässlich sind. Umgekehrt können bei einer fehlenden Single Source of Truth Daten über verschiedene Systeme, Abteilungen und Personen verstreut sein, was zu Inkonsistenzen, Fehlern und Ungenauigkeiten führt. Dies kann schnell zu fehlendem Vertrauen in Daten und Datenprodukten führen. Außerdem kann es schwierig werden, Probleme bzgl. der Datenqualität zu erkennen, sowie deren Ursachen zu finden und zu beheben.
- **Self-Service:** Eine wichtige Anforderung an eine Datenplattform ist die Möglichkeit für Benutzer:innen, einfach und flexibel auf die Daten zuzugreifen und Analysen oder Reports zu erstellen. Dies ist besonders wichtig für nicht-technische Rollen, wie z.B. Business Analysts, die möglicherweise nicht über die Fähigkeiten oder das Wissen verfügen, um komplexe Datenabfragen durchzuführen. Um diese Anforderungen zu erfüllen, müssen Tools zur Verfügung
- gestellt werden, die für alle Benutzer:innen einfach zu bedienen sind, wie z.B. Microsoft Excel, PowerBI oder andere gängige Analysetools.
- **Datendemokratisierung:** Eine Datenplattform unterstützt das Konzept der Demokratisierung von Daten, indem sie jedem im Unternehmen potenziell Zugang zu Daten und Analysen ermöglicht. Durch einheitliche Möglichkeiten zum Datenzugriff wird sichergestellt, dass jede:r Anwender:in auf die gleiche Weise auf die Daten zugreifen kann, ohne die unterschiedlichen Konfigurationen und Strukturen der verschiedenen Datenquellen kennen zu müssen. Dies erleichtert den Datenzugang im Vergleich zu direkten Zugriffen auf verschiedene Purpose-Datenbanken oder Datendumps. Jedoch muss sichergestellt werden, dass die Daten in Bezug auf Datenschutz und Vertraulichkeit geschützt sind und nur diejenigen Benutzer:innen darauf zugreifen können, die die entsprechenden Zugriffsberechtigungen haben.
- **Transparenz:** Im Sinne der Transparenz sollen Daten dabei helfen, das Unternehmen und seine Prozesse zu visualisieren und damit eine faktenbasierte Steuerung zu ermöglichen. Das Ziel ist es, Entscheidungen auf der Grundlage von verlässlichen Daten zu treffen, statt sich auf das Bauchgefühl zu verlassen. Eine Datenplattform sollte daher ein Dashboard oder ein Berichtswesen bieten, das es den Benutzer:innen ermöglicht, die wichtigsten Geschäftskennzahlen auf einen Blick zu sehen.

- **Agilität:** In unserer schnelllebigen Welt müssen Unternehmen in der Lage sein, auf sich ändernde Marktbedingungen reagieren zu können. Dies erfordert, dass neue Datenquellen schnell und einfach in die Plattform integriert werden können. Eine Datenplattform muss daher agil und anpassbar sein, um schnell auf Veränderungen reagieren zu können.
- **Performance:** Durch Einsatz einer Datenplattform werden viele Analyseabfragen erst möglich, da Purpose-Datenbanken in der Regel nicht für analytische Anfragen ausgelegt sind. Außerdem kann eine Datenplattform dazu beitragen, dass das Produktivsystem nicht zusätzlich belastet wird. Data Scientists müssen sich keine Sorgen machen, dass ihre Analysen das System verlangsamen oder sogar lahmlegen.

Neben den genannten Anforderungen gibt es auch weitere Nebenforderungen, wie die Sicherheit der Daten, die Datenqualität und -aktualität. Eine Datenplattform muss sicherstellen, dass die Daten vor unbefugtem Zugriff geschützt sind, dass die Datenqualität hoch ist und dass die Daten jederzeit aktuell sind.

Eine gute Dokumentation der Daten und Möglichkeiten zur Exploration (*Data Catalogue* und *Data Lineage*) sind ebenfalls wichtige Nebenanforderungen, um sicherzustellen, dass die Benutzer:innen verstehen, woher die Daten stammen und wie sie genutzt werden können.



Insgesamt kann eine Datenplattform dazu beitragen, dass Unternehmen ihre Daten effektiver nutzen und bessere Entscheidungen auf der Grundlage von verlässlichen Daten treffen können. Durch die Einhaltung der oben genannten Anforderungen kann eine Datenplattform als Single Source of Truth fungieren und das Vertrauen in die Daten im gesamten Unternehmen steigern.

2.3 Komponenten und Funktionen einer Datenplattform

Eine Datenplattform umfasst die Aufgaben der Integration (*Source*), Aufnahme (*Ingestion*), Speicherung und Verarbeitung (*Storage & Compute*), sowie Nutzung und Verteilung (*Consumption*) digitaler Daten. Darüber hinaus stellt sie Funktionalitäten zur Verwaltung und Orchestrierung (*Governance & Orchestration*) für die Verarbeitung von Daten bereit (siehe Abb. 1). Jede dieser Funktionen wird durch sogenannte Layer abgebildet. Im nachfolgenden Kapitel werden diese Layer detailliert beschrieben.

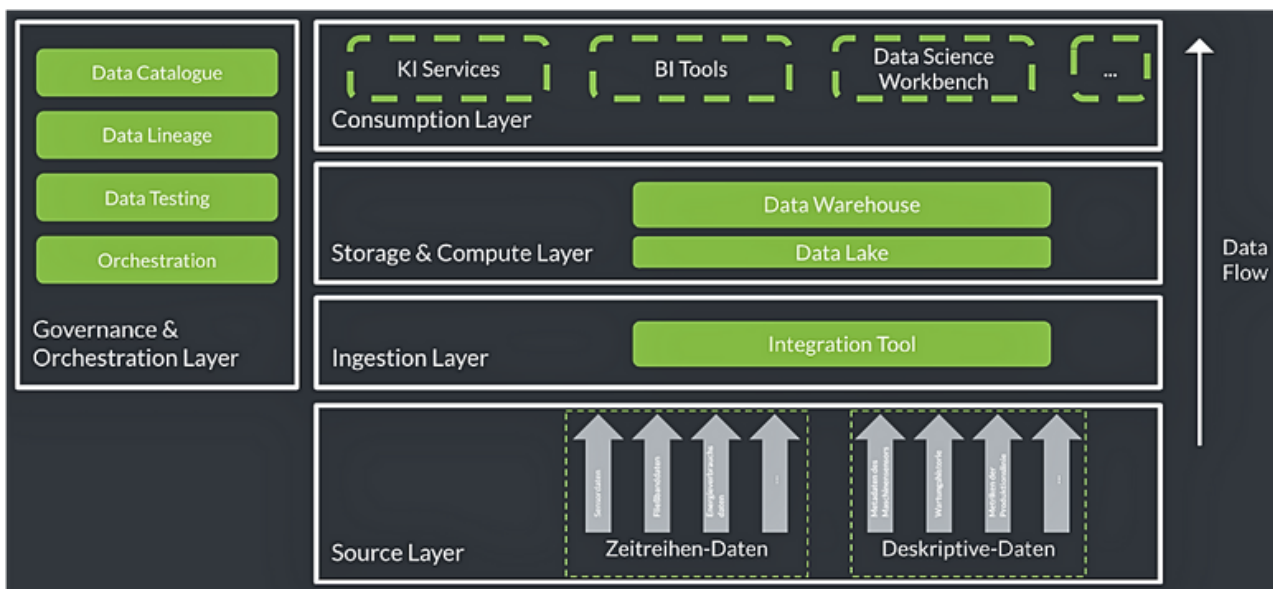


Abb. 1 Übersicht über die nach Funktionen unterteilten Layers einer Datenplattform

2.3.1 Source & Ingestion Layer

Eine zentrale Funktionalität einer Datenplattform besteht in der Integration digitaler Daten in verschiedensten Formen aus diversen Quellen. Hierzu zählen neben strukturierten Daten auch semi-strukturierte, unstrukturierte und deskriptive Daten, sowie Zeitreihendaten, die in Quellen, wie Datenbanken, Flat Files oder Sensoren erzeugt werden und angebunden werden müssen.

Um Daten aus Quellsystemen in eine Datenplattform zu übertragen, können zwei verschiedene Ansätze zum Einsatz kommen: **(1)** Die **Batch-Verarbeitung**, bei der Daten aus einer Quelle zu einem vordefinierten Zeitpunkt oder in vordefinierten Gruppierungen aufgenommen werden und **(2)** die **Stream-Verarbeitung**, bei der Daten (nahezu) in Echtzeit verarbeitet werden.

(1) Batch-Verarbeitung:

Die Batch-Verarbeitung ist eine Methode, um große Datenmengen, die über einen bestimmten Zeitraum gesammelt und gespeichert wurden, auf einmal zu verarbeiten und an ein Analysesystem weiterzuleiten. Sie erfordert eine Art von Speicher (Datenbank oder Dateisystem) zum Laden und Verarbeiten von Daten, deren Größe begrenzt ist. Bei dieser Technik werden Transaktionen oder Datensätze gruppiert und nicht einzeln, sondern als Ganzes verarbeitet (siehe Abb.2).

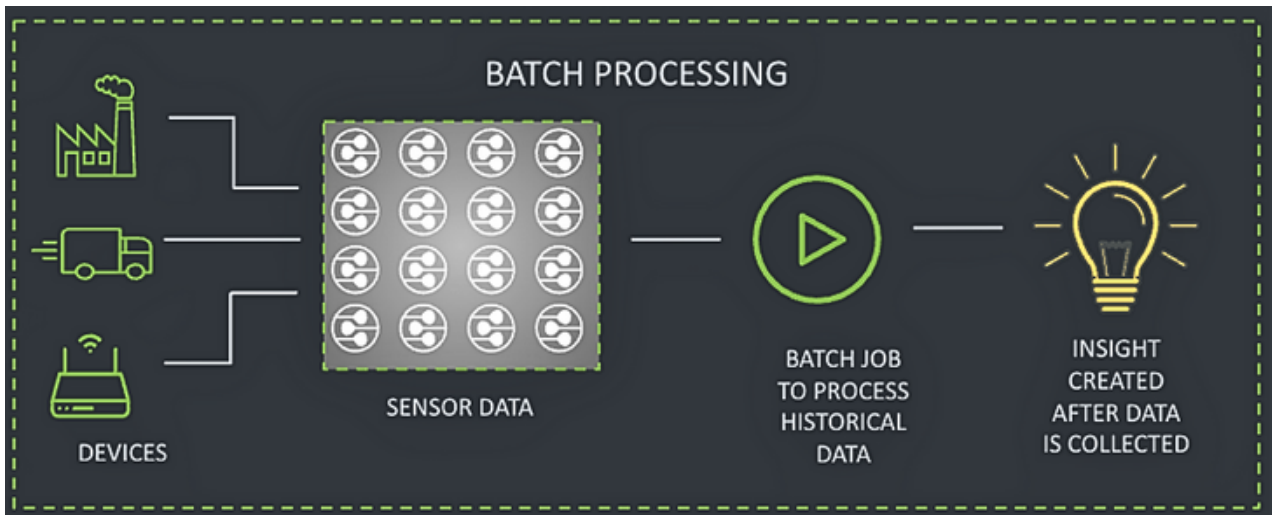


Abb.2 Prozessuale Darstellung einer Batch-Verarbeitung

In der Vergangenheit war die Batch-Verarbeitung die einzige Methode zur Verarbeitung großer Datenmengen, da die Computer nicht leistungsfähig genug waren, um sie in Echtzeit zu verarbeiten.

Heutzutage wird die Batch-Verarbeitung immer noch für einige Aufgaben in der industriellen Analyse verwendet, aber für die meisten KI-Dienste, die eine Datenanalyse in Echtzeit erfordern, wurde sie weitgehend durch die Stream-Verarbeitung ersetzt.

Anwendungsfälle für die Batch-Verarbeitung in der industriellen Analyse sind:

- Datensicherung und -archivierung (z.B. Rohdaten-Backups über Nacht)
- ETL-Prozesse (Extrahieren, Transformieren, Laden) (z.B. Datenmigration zwischen Systemen)
- Berichterstellung (z.B. monatliche Wartungsberichte)
- Analysetools zur Gewinnung von Erkenntnissen aus Daten (z.B. Segmentierung von Industrieanlagen)
- maschinelles Lernen oder Data Mining (z.B. Training von Modellen zur vorausschauenden Wartung)

(2) Stream-Verarbeitung:

Stream-Verarbeitung meint eine Verarbeitungstechnik, bei der Daten dann verarbeitet werden, wenn sie für eine bestimmte Verwendung benötigt werden oder wenn sie erstellt werden. Das bedeutet, dass Daten gesammelt und dann sofort oder sehr bald nach der Sammlung verarbeitet werden. Dies ermöglicht die Analyse von Stream-Daten in Echtzeit, die für viele Anwendungen, insbesondere für KI-basierte Dienste, unerlässlich ist.

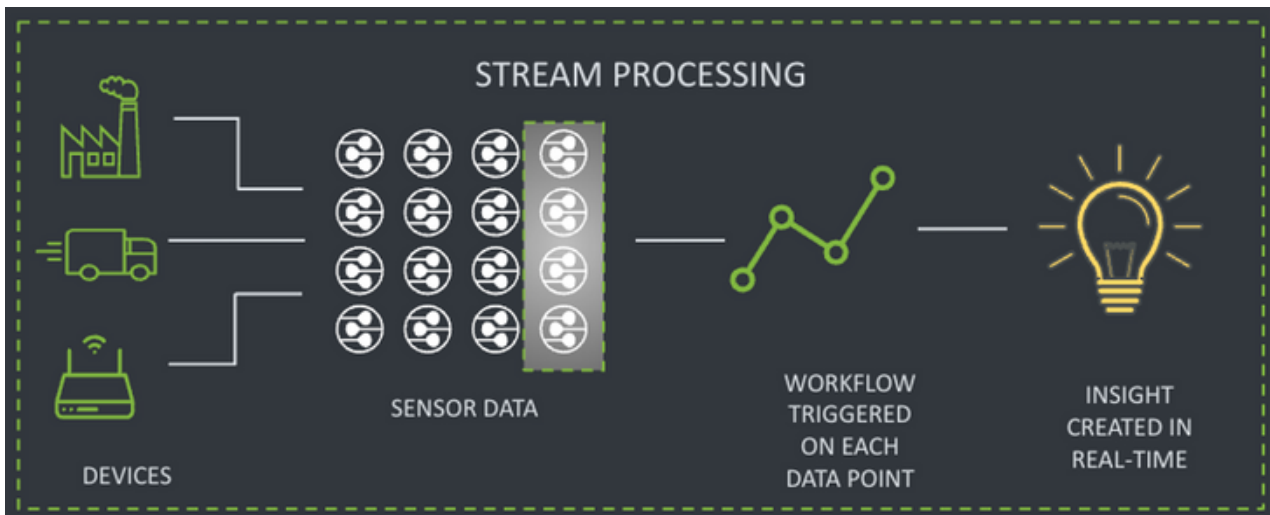


Abb.3 Prozessuale Darstellung einer Stream-Verarbeitung

Die Stream-Verarbeitungstechnologie wird eingesetzt, um eine konstante Datenzufuhr in (nahezu) Echtzeit zu verarbeiten, um sie weiterzunutzen, Berichte zu erstellen oder automatische Reaktionen auszulösen, ohne dass sie zuerst heruntergeladen werden müssen. Zudem sind sie durch ihre minimale Latenz in Situationen unerlässlich, in denen Verzögerungen zu negativen Ergebnissen führen könnten.

Echtzeitverarbeitung bedeutet, dass die Daten fast sofort, innerhalb von Millisekunden, verarbeitet werden. Die Architektur des Datenstroms gibt die Möglichkeit, Daten in Bewegung aufzunehmen, zu verarbeiten, zu speichern, anzureichern, zu strukturieren und zu analysieren. Hierzu liest und verarbeitet ein Stream-Prozessor kontinuierlich Datenströme aus Eingabequellen nach bestimmten Regeln oder Logiken und schreibt die Ergebnisse in Ausgabeströme.

Der Prozessor kann einen oder mehrere Threads verwenden, um Parallelität zu ermöglichen und die Leistung zu verbessern.

Industrielle Analyse Anwendungsfälle für die Stream-Verarbeitung sind:

- Sensordaten (z.B. Erkennung von Anomalien in Sensordaten oder Verkehrsüberwachung in Echtzeit)
- Energienutzungsdaten (z.B. Optimierung des Energieverbrauchs)
- Qualitätskontrolle in der Industrie (z.B. visuelle Qualitätskontrolle)
- Andere KI-basierte Anwendungen (jede Lösung, die historische und Echtzeit-Datenquellen vergleichen und analysieren muss)

2.3.2 Storage & Compute Layer

Nachdem Daten aus diversen Quellen eingespeist wurden, können diese in verschiedenen Speichersystemen einer Datenplattform persistiert werden. Anschließend können Daten vor der weiteren Nutzung in dem Consumption Layer transformiert, verarbeitet oder bereinigt werden. In den nachfolgenden Absätzen werden konkrete Lösungsansätze und Technologien für diese beiden Einsatzgebiete vorgestellt.

a) Datenhaltung (Storage):

Für die Datenhaltung koexistieren verschiedene technische Lösungen. Die drei meist genutzten Varianten sind *Data Lake*, *Data Warehouse*, sowie deren hybride Mischung *Data Lakehouse*:

- **Data Lake:** skalierbare Repositories, die viele Arten von Daten in roher und nativer Form speichern können, insbesondere für semistrukturierte und unstrukturierte Daten. Sie stellen grundsätzlich unbegrenzt Speicherkapazität zur Verfügung und sind kostengünstiger als andere Lösungen zur Datenhaltung. Um wirklich nützlich zu sein, müssen sie eine benutzerfreundliche Erkundung mit gängigen Methoden und Abfragesprachen, wie z.B. SQL ermöglichen, routinemäßige Datenverwaltungsaktivitäten automatisieren und eine Reihe von Analyseanwendungen unterstützen.
- **Data Warehouse:** in der Regel nur zur Speicherung strukturierter Daten, die durch ein relationales Datenbankschema definiert sind. Die Rohdaten müssen oft transformiert werden, um dem Schema zu entsprechen. Moderne Data Warehouse sind für die Verarbeitung von Tausenden oder sogar Millionen von Abfragen pro Tag optimiert und können spezifische Geschäftsanwendungen unterstützen.

- **Data Lakehouse:** Cloud-basierte Versionen dieser Datenspeicheroptionen haben zu hybriden Lösungen geführt, die die besten Eigenschaften von Data Lake und Data Warehouse miteinander kombinieren. Einige dieser neueren Optionen sind so konzipiert, dass sie sowohl strukturierte und halbstrukturierte Daten als auch die wichtigsten Attribute unstrukturierter Daten speichern und verarbeiten können.

b) Datentransformation (Compute):

Die Datentransformation ist der Prozess der Vorbereitung von Daten für verschiedene Arten der Nutzung. Sie kann die *Standardisierung* (Konvertierung aller Datentypen in dasselbe Format), das *Cleansing* (Beseitigung von Inkonsistenzen und Ungenauigkeiten), das *Mapping* (Kombination von Datenelementen aus zwei oder mehr Datenmodellen) oder das *Enrichment* (Hinzufügen von Daten aus anderen Quellen) umfassen.

Eine Datentransformation ist meistens aus vielen verschiedenen Gründen erforderlich, die sich insbesondere aus den konkreten Anwendungsfällen ergeben. Daneben steht jedoch auch die Gewährleistung von Datenqualität im Vordergrund. So müssen Daten bereinigt werden, um Fehler und Duplikate zu beseitigen, z.B. unterschiedliche Schreibweisen von Namen, Adressen und anderen Anomalien. Eine gute Datenqualität sorgt für genaue Berichte und Analysen und ermöglicht letztlich erweiterte Anwendungsfälle, wie maschinelles Lernen.

Um Daten aus verschiedenen Quellen zu extrahieren, zu transformieren und in ein anderes Ziel zu laden, können zwei verschiedene Methoden verwendet werden: Hier gibt es den sog. **ETL-Ansatz** (kurz für "Extract, Transform, Load") und den **ELT-Ansatz** (kurz für "Extract, Load, Transform").

Eigenschaften von ETL:

- Extrahierte Daten werden in eine einheitliche Struktur gebracht, bereinigt und aufbereitet, um für das Ziel-System optimiert zu sein.
- Die transformierten Daten werden in das Ziel-System geladen, wie z.B. Data Warehouse oder Data Mart.
- Tendenziell einfacher zu implementieren als ELT, da die Transformation der Daten in einem separaten Schritt erfolgt.
- Wird oft bei der Integration von Daten in Data Warehouses eingesetzt.

Eigenschaften von ELT:

- Extrahierte Daten werden direkt in das Ziel-System geladen, ohne dass sie im Voraus transformiert werden.
- Transformation der Daten findet im Ziel-System statt, indem sie mit speziellen Tools oder Skripten transformiert werden.
- Bietet eine höhere Skalierbarkeit, da die Verarbeitung der Daten direkt im Ziel-System stattfindet und somit dessen Leistung genutzt wird.
- Wird oft bei Cloud-basierten Systemen eingesetzt.

2.3.3 Consumption Layer

Der Consumption Layer in Datenplattformen bietet Zugriff auf die verarbeiteten Daten für Benutzer:innen und Anwendungen. Sie umfasst Analysetools wie Business Intelligence-Lösungen, die den Benutzer:innen die Möglichkeit bieten, Daten auf verschiedene Weise zu visualisieren, zu analysieren und zu verstehen. Business Intelligence-Lösungen ermöglichen es Benutzer:innen, schnell und einfach Dashboards und Berichte zu erstellen, um wichtige Geschäftskennzahlen zu verfolgen und Entscheidungen zu treffen. Darüber hinaus umfassen sie KI-basierte Services, die einen produktbezogenen Mehrwert durch fortschrittliche Analysefunktionen auf Basis maschinellen Lernens bieten ([mehr dazu erfahren Sie in Kapitel 3 KI Services](#)).

So ermöglicht der Consumption Layer es Benutzer:innen, die verarbeiteten Daten zu nutzen und daraus Wert zu generieren.

2.3.4 Governance & Orchestration Layer

Um eine effektive Data-Engineering-Praxis zu etablieren, ist es wichtig, ein Gleichgewicht zwischen Agilität und Governance zu finden. Aus diesem Grund sind Datenorchestrierung und -governance entscheidende Säulen beim Aufbau einer robusten Datenplattform.

Während umfassende Kontrollen notwendig sind, um die Sauberkeit, Genauigkeit und Aktualität der Daten zu gewährleisten, können übermäßig aufwändige Governance-Praktiken die Produktivität der Benutzer-Community behindern. Letztendlich geht es darum, eine agile Umgebung bereitzustellen, die für eine Vielzahl von Interessengruppen zugänglich ist. Darunter erfahrene Data Engineers, Data Scientists und gelegentliche Nutzer:innen, die ihre Daten mithilfe von Self-Service-Tools untersuchen, verbessern, ändern und bereinigen möchten.

In diesem Abschnitt wird erläutert, warum es wichtig ist, dass eine vielfältige Gruppe von Personen gute Regeln für den Umgang mit Daten durchsetzt, was ein sicheres und kreatives Umfeld schaffen soll und gleichzeitig die Einhaltung von Datenschutzgesetzen sicherstellen muss.

Für den Aufbau einer erfolgreichen Datenplattform ist es entscheidend, Daten als ein Produkt zu betrachten, sowie geeignete Tools und Technologien einzusetzen. In jüngster Zeit wurden die Grundsätze von DevOps, die ursprünglich zur Förderung der agilen Softwareentwicklung entwickelt wurden, auf die Datenmodellierung und -gestaltung übertragen. Daraus ist der neue Begriff DataOps (*Data Operations*) entstanden, der sich auf die Praktiken zur Automatisierung des Datenentwicklungszyklus bezieht.

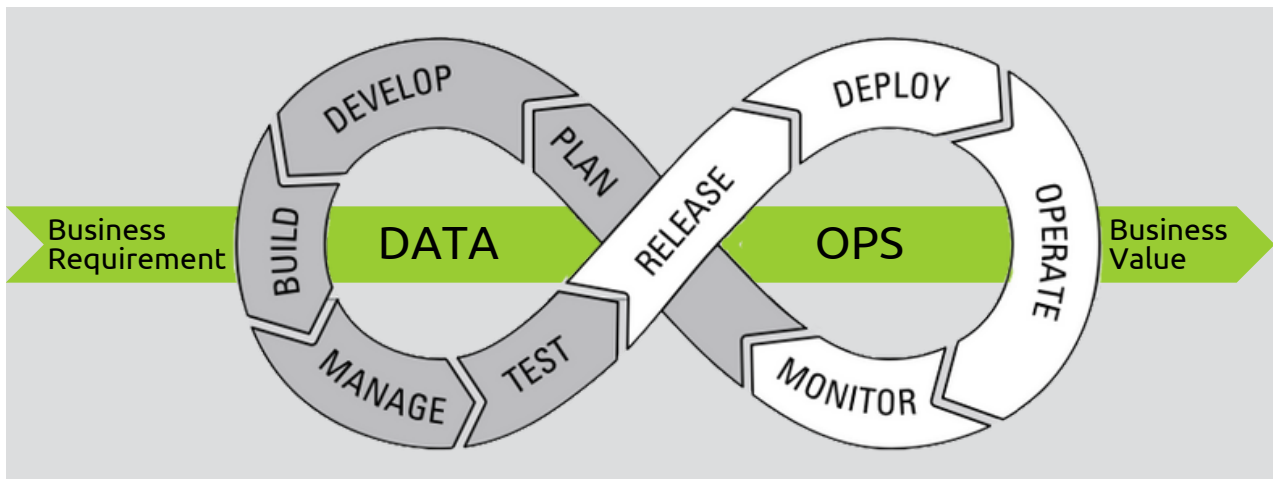


Abb. 4 Der DataOps Prozess (Figure Source: Datalytx DataOps)

Da sich die Daten- und Analyseanforderungen ständig weiterentwickeln, sind verwaltete Self-Service-Verfahren und kontinuierliche DataOps-Bereitstellungsmethoden unabdingbar, um sicherzustellen, dass Daten korrekt und gut verwaltet durch die Pipeline fließen.

Die Grundlage von DataOps liegt in der Data Governance, die mehrere wichtige Aspekte umfasst. Dazu gehören:

- **Lineage:** Das Verständnis des Ursprungs, der Verwendung und der Änderungen von Daten im Laufe der Zeit ist wichtig für die Einhaltung von Vorschriften, für Audits, Fehlererkennung und Dokumentationszwecke. Zudem müssen Data Scientists möglicherweise die Datenherkunft nachvollziehen, um ihre Modelle für maschinelles Lernen zu erklären.
- **Datenqualität:** Das Vertrauen in Daten ist entscheidend für den erfolgreichen Einsatz von Daten. Unzuverlässige Daten können zu fehlerhaften Geschäftsentscheidungen, Umsatzeinbußen und Geldstrafen für den rechtswidrigen Umgang mit personenbezogenen Daten führen.
- **Data Catalog:** Ein Datenkatalog mit Metadaten und Suchwerkzeugen hilft Benutzer:innen, relevante Daten zu finden und auszuwerten.
- **Datenzugriff:** Es müssen Regeln festgelegt werden, um zu bestimmen, wer auf Daten zugreifen, mit ihnen arbeiten und sie ändern darf. Dies gilt insbesondere für sensible Informationen, wie personenbezogene Daten und Finanzdaten, die ggf. anonymisiert, oder pseudonymisiert werden müssen, oder deren Zugriff durch eine Multifaktor-Authentifizierung (MFA) gesichert werden muss, um den Datenschutzbestimmungen zu entsprechen.

2.4 Zusammenfassung und Ausblick

Mit diesem Kapitel haben Sie einen Überblick über die zentralen Ziele und Funktionen einer modernen Datenplattform erhalten. Hierzu wurde der Plattform-Gedanke aus Sicht von Softwareanwendungen auf die Welt der Daten übertragen. Anschließend wurde durch die Definition der Ziele und Anforderungen beschrieben, was bei der Realisierung einer Datenplattform zu beachten ist. Eine umfangreiche Beschreibung der einzelnen Layer einer Data Plattform, inklusive der möglichen technischen Optionen können helfen zu

bewerten, wo sich Ihr eigenes Unternehmen auf der Reise zu einem datengetriebenen Unternehmen befindet. Denn eine moderne Datenplattform ist die Grundlage für datengetriebene Entscheidungen und KI-basierte Services. Jedoch existiert keine „Einheitslösung“, sondern jedes Unternehmen muss anhand der spezifischen Anforderungen und Gegebenheiten festlegen, welche Komponenten für den Aufbau der eigenen, modernen Datenplattform sinnvoll sind.

3 KI SERVICES

3.1 Industrial Analytics und datengetriebene Wertschöpfung

Industrial Analytics, zu Deutsch etwa „industrielle Analyse“, meint die Anwendung von Data Analytics (Datenanalyse) für die Wertschöpfung im industriellen Umfeld.

Ziel ist es, Daten aus der Industrie (z.B. aus Maschinensensoren oder Anlagenüberwachungssystemen) aufzubereiten und zu analysieren, um so Erkenntnisse über Produkte, Prozesse und Services zu erhalten oder neu zur Verfügung zu stellen. Nähere Informationen zur technischen Integration verschiedener Datenquellen gibt es in [Kapitel 2 Data Platforms](#).

Dabei verknüpft Data Analytics als Ansatz diverse Konzepte, Technologien und Tools, die ein tiefgreifendes Verständnis und die Entdeckung von nutzbaren Erkenntnissen aus Daten ermöglichen. So umfasst diese Domäne alle Arten von datenbasierten Analysetechniken:

Neben klassischem Reporting und Online Analytical Processing (OLAP), bis hin zu Data Mining auf Basis maschinellen Lernens. Analysetechniken, die über klassisches Reporting und OLAP hinausgehen, werden dabei unter dem Oberbegriff Advanced Analytics subsumiert. Dazu zählen auch Methoden der künstlichen Intelligenz, insbesondere Techniken des Deep Learnings.

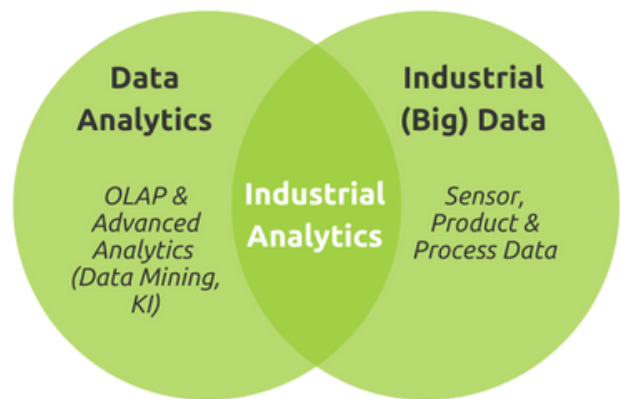


Abb.1. Begriffseinordnung Industrial Analytics

Das Ziel von Industrial Analytics ist es, den geschäftlichen Wert industrieller Daten in Form datengetriebener Wertschöpfung zu nutzen. Datengetriebene Wertschöpfung bietet eine Möglichkeit, wie Unternehmen die vielfältigen Technologien rund um Industrial Analytics für einen unternehmerischen Mehrwert einsetzen können.

Möglich wird dies neben der Verbesserung bestehender Prozesse und Leistungen auch durch Bereitstellung neuer, datenbasierter Services und Produkte. Die Vielzahl an möglichen Services und Produkten auf Basis von Daten und künstlicher Intelligenz (KI) wird im nächsten Kapitel behandelt.

3.2 Typische Use Cases (Problem - Ansatz - Lösung)

Unabhängig davon, welcher Use-Case später realisiert werden soll, erfolgt das Vorgehen bei der Umsetzung von Anwendungsfällen für Industrial Analytics nach einem festen Schema. Dabei wird wie folgt vorgegangen:

Zunächst ist es erforderlich, digitale Daten von diversen Quellen, wie beispielsweise Sensoren an industriellen Maschinen und anderen Geräten oder externen Datenbanken, zu erfassen.

Anschließend müssen die erfassten Daten transformiert und aufbereitet werden, damit sie für eine Analyse in mathematischen Vorhersagemodellen eingesetzt werden können, welche mit Algorithmen des maschinellen Lernens wie Regression, Clustering oder Klassifizierungen arbeiten.

Bei der Datenaufbereitung werden die Datensätze bereinigt und in Formate umgewandelt, die sich für das Modelltraining eignen. Hierzu werden Daten standardisiert und ggf. normalisiert. Sobald dieser Schritt abgeschlossen ist, kann der aufbereitete Datensatz verwendet werden, um ein Modell für einen spezifischen Anwendungsfall - wie beispielsweise Predictive Maintenance, Automated Quality Control oder Machine Efficiency Optimization - zu trainieren.

Nachdem das sogenannte Modelltraining abgeschlossen und die Leistung des Modells gemäß standardisierter Bewertungsmetriken als ausreichend bewertet wurde, kann das trainierte Modell für den produktiven Einsatz herangezogen werden.

Dabei gilt es zu beachten, dass sowohl die benötigten Daten, als auch die definierten Bewertungsmetriken der Modelle sich je nach Anwendungsgebiet stark unterscheiden können, sodass es notwendig ist, den späteren Use-Case bereits frühzeitig zu definieren.

3.2.1 Predictive Maintenance (PdM)

Predictive Maintenance (PdM) beschreibt eine Wartungsstrategie, die darauf abzielt, durch intelligente Überwachung von Systemen und Anlagen potenzielle Ausfälle zu erkennen, bevor diese auftreten. Durch die Vorhersage, wann eine Wartung erforderlich ist, kann PdM demnach dazu beitragen, Anlagenausfälle zu verhindern, Ausfallzeiten zu reduzieren und die betriebliche Effizienz zu steigern. Dabei beruht PdM auf der Möglichkeit, zukünftige Werte von Größen, die ein System - wie z.B. eine Maschine, eine Anlage oder ein Produktionsprozess - charakterisieren, durch mathematische Modelle zu modellieren, um somit Anomalien oder Fehler frühzeitig zu identifizieren.

Übliche Beispiele des PdM sind:

- Maschinenvibrationen können auf eine Abnutzung von Lagern oder die Verformung bestimmter mechanischer Bauteile hinweisen.
- Temperaturen eines Motors und Kriechstrom können darauf hinweisen, dass Reibung und mögliche mechanische Fehlfunktionen die Funktionalität beeinträchtigen.

- Die Messung von Partikeln in einem Schmiermittel kann einen Rückschluss auf eine Verschlechterung von reibenden Kontaktstellen zulassen. Mit geeigneten Sensoren lässt sich die Zusammensetzung des Schmieröls messen und damit der Zustand der Maschine überprüfen.

Das grundsätzliche Vorgehen in PdM umfasst die folgenden Schritte:

- Messung von physikalischer Größen in Echtzeit mittels angebrachter Sensoren.
- Schätzung der messbaren (oder nicht messbaren) Parameter zu einem beliebigen Zeitpunkt mittels mathematischer Modelle auf Basis von maschinellem Lernen.
- Identifizierung von Systemzuständen, die als anormal oder fehlerhaft gelten.
- Planung und Definition von vorbeugenden sowie korrigierenden Maßnahmen zur Reaktion auf erkannte fehlerhafte oder anormale Zustände von Komponenten, bevor das gesamte System einen kritischen Zustand erreicht.



Anomale oder fehlerhafte Zustände von Maschinen durch Predictive Maintenance frühzeitig erkennen.

3.2.2 Automated Quality Control (AQC)

Allgemein meint Quality Control (Qualitätskontrolle) Verfahren und Maßnahmen von Fertigungsunternehmen zur Sicherstellung, dass erzeugte Produkte und die in deren Herstellung durchgeführten Prozesse Normen entsprechen, die von Unternehmen, Kunden oder Dritten festgelegt wurden. Automated Quality Control (AQC) beschreibt hingegen den Einsatz von Methoden des maschinellen Lernens zur automatisierten Überwachung und Gewährleistung der Qualität von Produktionsprozessen und Produkten im industriellen Umfeld. Dabei werden Sensoren, Datenanalysetools sowie Algorithmen eingesetzt, um Produktionsprobleme in Echtzeit zu erkennen und zu diagnostizieren, um so das Risiko von Fehlern, Ausschuss und kostspieligen Ausfallzeiten zu minimieren. Bereits hier zeigt sich die Überschneidung zu PdM, die eine spezifische Methodik von AQC sein kann.

Übliche Anwendungsbeispiele von AQC sind:

- PdM: *siehe oben*
- Automated Visual Inspection: Verfahren, bei dem Computer Vision und Bildverarbeitungstechniken zur automatischen Inspektion und zur Erkennung von Fehlern oder Anomalien in Produkten oder Komponenten während des Herstellungsprozesses eingesetzt werden.
- Process Optimization: Analyse von digital abgebildeten Prozessen, um so Ineffizienten und Bottlenecks zu erkennen.

Vorgehen für Automated Visual Inspection:

- Errichtung einer Kamera-Infrastruktur, die in der Lage ist, Objekte akkurat und in geeigneter Qualität in Form von Videos oder Bildern zu erfassen.
- Vorbereitung der Bilder mithilfe von Bildfiltern, wie beispielsweise Hochpass- oder Tiefpassfiltern. Entwicklung von mathematischen Modellen auf Basis der vorbereiteten Daten.
- Identifizierung und Kennzeichnung von Zuständen, die als anormal oder fehlerhaft gelten anhand von Aufnahmen von normgemäßen Produkten während des Herstellungsprozesses auf Basis von Größe, Form, Position oder Kolorierung einzelner Bauteile und Komponenten.
- Planung und Definition von vorbeugenden und korrigierenden Maßnahmen zur Reaktion auf erkannte fehlerhafte oder anormale Produkte während des Herstellungsprozesses, um Ausschuss und Qualitätseinbußen zu vermeiden.

Vorgehen für Process Optimization:

- Modellierung und digitale Abbildung relevanter Geschäftsprozesse, z.B. mithilfe einer Prozess/Workflow Engine.
- Erkennung von Engpässen oder Ineffizienzen in Prozessen oder den entsprechenden Workflows mithilfe von ML-gestützter Mustererkennung im Process Mining.
- Steigerung der Prozesseffizienzen durch Anpassung des Prozesses via Parallelisierung, Entkopplung oder Schaffung erweiterter Verarbeitungsressourcen.



3.2.3 Machine Efficiency Optimization (MEO)

Machine Efficiency Optimization (MEO) ist ein Prozess zur Maximierung der Leistung und Produktivität industrieller Maschinen. Dabei werden Faktoren identifiziert und beseitigt, die sich negativ auf die Effizienz von Maschinen auswirken. Durch die Optimierung der Maschineneffizienz können Unternehmen ihre Produktion steigern, Kosten senken und die Produktqualität verbessern.

Typische Anwendungsfälle zur Steigerung der Maschineneffizienz sind:

- Senkung von Ausfallzeiten: Zur Reduzierung von Zeiten, in denen eine Maschine aufgrund von Wartung, Reparatur oder anderen Faktoren nicht in Betrieb ist, kann PdM das Mittel der Wahl sein.
- Vermeidung von Produktdefekten: Defekte im Fertigungsprozess können zu Zeitverlusten und Materialverschwendung führen. Die Minimierung von Defekten ist der Schlüssel zur Maximierung der Maschineneffizienz. Hier kann AQC im Sinne der Produktqualitätskontrolle das Mittel der Wahl sein.

- Reduzierung von Zykluszeiten: Langsame Zykluszeiten können die Gesamtproduktivität und den Ausstoß verringern. Hier kann AQC im Sinne der Prozessüberwachung Abhilfe schaffen.
- Steigerung von Wirkungsgraden: Durch die Identifikation ineffizienter Betriebszustände, wie z.B. ungünstige Betriebspunkte eines Motors, können Wirkungsgrade erhöht und damit Energie eingespart werden. Nicht selten sind ineffiziente Betriebszustände darauf zurückzuführen, dass Motoren nicht optimal auf geänderte Anwendungen abgestimmt sind. So werden Motoren zwar gemäß dem initial beabsichtigten Einsatzgebiet dimensioniert, im späteren Verlauf zu anderen Zwecken verwendet und dann gegebenenfalls in sehr ineffizienten und verschleißintensiven Betriebspunkten betrieben. Werden ineffiziente Betriebszustände erkannt, können meist bereits durch Einsatz eines Frequenzumrichters bereits deutliche Effizienzsteigerungen erreicht werden.

Vorgehen für die Steigerung von Maschinenwirkungsgraden:

- Auswahl eines repräsentativen Teilsystems, das sich isoliert untersuchen lässt.
- Erlernen von Verständnis über die theoretischen Hintergründe des Teilsystems.
- Messung von physikalischen Größen in Echtzeit mittels angebrachter Sensoren und Vergleich mit den zu erwartenden Werten der physikalischen Größen an den entsprechenden Betriebspunkten.
- Analyse von Abweichungen zwischen den erwarteten und den tatsächlichen gemessenen Kenngrößen mithilfe von Modellen auf Basis von maschinellem Lernen. Für diese Art der Ursachenanalyse sollten gut erklärbare ML-Modelle verwendet werden.
- Ableitung von Maßnahmen zur Vermeidung der ineffizienten Betriebspunkte.



3.3 Vorgehensweise und Rollen – Von der Idee zum digitalen Produkt

Dieses Kapitel gibt Einblick in das Vorgehen und die involvierten Rollen bei der Einführung eines KI-basierten Service in industriellen Unternehmen. Hierzu wird erklärt, wie zunächst explorativ Daten gesammelt und analysiert werden und sich ein erster Prototyp des Services implementieren lässt. Im weiteren Verlauf wird beleuchtet, wie es gelingt, den entstandenen Prototypen in die unternehmerische IT-Umgebung einzugliedern, zu einem digitalen Service weiterzuentwickeln und diesen letztendlich als Produkt zu vermarkten.

3.3.1 Allgemeines Vorgehen

1. Auswahl eines Use Cases:

Das vorangegangene Kapitel hat gezeigt, welche Anwendungsfälle von KI in der Industrie häufig sind. Dennoch gilt es für jedes Unternehmen auf Basis strategischer Rahmenbedingungen und Entscheidungen selbst zu definieren, welches potenzielle Einsatzgebiet von KI aufgrund der eigenen Gegebenheiten infrage kommt. Gleichzeitig muss dabei auch berücksichtigt werden, wie sich die Kosten für die Entwicklung einer solchen Lösung amortisieren: Soll der entwickelte Service später in Form eines kostenpflichtigen Produktes angeboten werden oder handelt es sich um eine, für Kunden unentgeltlich nutzbare, Erweiterung der angebotenen physischen Produkte? Dabei gilt, dass der Break-even-Point aufgrund der geringen Nutzungsgebühren im Vergleich zu den anfänglich hohen Investitionskosten für KI-basierte Services erst nach einigen Jahren erreicht wird. Nichtsdestotrotz können die mithilfe der digitalen Services gesammelten Daten, für unternehmerische Entscheidungen wichtige Informationen liefern und zusätzliche Customer-Touchpoints generieren (siehe Kapitel 1 *Connectivity*).

Bei der Auswahl geeigneter Anwendungsfälle kann ein Value Proposition Canvas in Kombination mit einem Business Model Canvas helfen.

2. Exploratives Data Mining als Start in die Entwicklung KI-basierter Services:

Fakt ist, dass digitale Daten die elementare Grundlage sämtlicher KI-basierter Services bilden. Steht fest, welcher Use-Case umgesetzt werden soll und sind die im Unternehmen verfügbaren Daten bereits in geeigneter und standardisierter Form in der hauseigenen Data Platform verfügbar (siehe Kapitel 2 *Data Platforms*), so können sie unkompliziert für explorative Analysen herangezogen werden. Das Ziel dieser Analyse besteht darin, erste Erkenntnisse aus den Daten im Sinne von Data Mining zu gewinnen. Hierzu kann nach dem Cross-Industry Standard Process for Data Mining, kurz CRISP-DM, vorgegangen werden, um das Data Mining Projekt gut zu strukturieren (siehe Abb. 2).

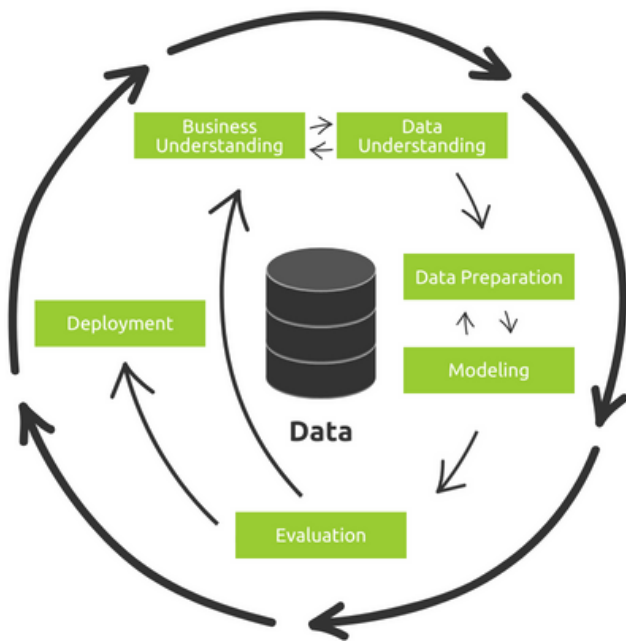


Abb. 2 CRISP-DM Diagramm

Dabei ist zunächst wichtig zu definieren, welche konkreten betriebswirtschaftlichen Ziele mithilfe des KI-Service zu erreichen sind (**Business Understanding**). Geht es beispielsweise um die Reduzierung von Kosten, die Effizienzsteigerung von Anlagen oder aber nur um das Angebot einer zusätzlichen Kundenleistung in digitaler Form?

Dabei gilt es, in einem iterativen Prozess die für die betriebswirtschaftlichen Ziele benötigten Daten zu definieren und zu analysieren (**Data Understanding**). Dies kann zur Folge haben, dass die Ziele ggf. angepasst oder weitere Daten in die hauseigene Data Platform aufgenommen werden müssen. Nicht selten kann es aber auch dazu führen, dass es eigene betriebliche Zusammenhänge und Prozesse noch einmal klarer zu verstehen gilt, bevor die erforderlichen Informationen aus den Daten gewonnen werden können.

Sind die Daten und unternehmerischen Gesichtspunkte so weit analysiert, kann

damit begonnen werden, die Daten für den konkreten KI-Anwendungsfall aufzubereiten (**Data Preparation**).

Dabei geht es insbesondere auch um die fachgerechte Aufbereitung in Abhängigkeit vom später zu verwendenden Machine Learning (ML) Algorithmus: so erfordern Recommender Systeme zur Produktempfehlung beispielsweise eine andere Aufbereitung von Daten als eine klassische Zeitreihenanalyse.

Sobald die Daten in geeigneter Form vorliegen, kann im nächsten Schritt mit der Auswahl eines ML-Modells, dessen Training und damit die korrekte Parametrisierung begonnen werden (**Modeling**). Sobald das Modelltraining abgeschlossen ist, kann die Leistung des Modells hinsichtlich der definierten Zielmetrik evaluiert werden (**Evaluation**). Hierzu wird auf standardisierte, aus der Statistik bekannte Bewertungsmetriken zurückgegriffen, wie beispielsweise die Genauigkeit (Recall) oder die Präzision (Precision) im Falle einer Klassifizierung, bzw. der Mean Absolute Error (MAE) oder der Mean Squared Error (RMSE) im Falle einer Regression.

Auf Basis der gewonnenen Informationen und Erkenntnissen gilt es im nächsten Schritt zu entscheiden, ob die entwickelte Lösung bereits zu einem für Kunden und andere Anwender zugänglich nutzbaren Service umgewandelt werden soll (**Deployment**). Dieses Vorgehen wird in den nachfolgenden Unterkapiteln weiter im Detail beschrieben. Sind die gewonnenen Erkenntnisse noch nicht ausreichend zur Weiterentwicklung als nutzbarer Prototyp, kann der o.g. Prozess im Sinne eines iterativen Vorgehens weitergeführt oder die Erkenntnisse lediglich für interne Forschungszwecke herangezogen werden.

3. Von der explorativen Analyse zum Software-Prototypen:

Sobald erste vielversprechende Erkenntnisse in einem explorativen Setup - wie in einer lokalen Entwicklungs- oder Notebook-Umgebung - erarbeitet wurden, kann im nächsten Schritt die Entwicklung eines Prototypen als Minimum Viable Product (MVP) erfolgen.

Im Sinne des MVP-Ansatzes steht dabei die schnelle Bereitstellung von ersten grundlegenden Funktionen des Prototypen bei möglichst geringem Entwicklungsaufwand im Vordergrund. Fortgeschrittene Methoden und Konzepte der Softwareentwicklung wie Continuous Integration / Continuous Delivery (CI/CD) oder Clean Code Prinzipien sind hier noch nicht zu berücksichtigen.

Durch die Bereitstellung eines MVP kann bereits frühzeitig Feedback von (potenziellen) Kunden eingeholt werden, welches für die iterative Weiterentwicklung des Prototypen herangezogen werden kann. In der Kommunikation mit Verantwortlichen aus dem Produktmanagement oder dem Vertrieb muss insbesondere darauf geachtet werden, klarzustellen, dass das primäre Ziel des MVP darin besteht, Feedback einzuholen. Der Prototyp erfüllt zwar bereits einige funktionale, jedoch keinesfalls weitergehende, nicht-funktionale Softwareanforderungen und ist in diesem Entwicklungsstand demnach noch nicht für die großflächige Vermarktung geeignet.

4. Vom Software-Prototyp zum wartbaren, ubiquitären Service:

Bei der Weiterentwicklung des Software-Prototypen in einen wartbaren, ubiquitären Service empfiehlt sich die Anwendung nachfolgender Best Practices:

a) Überführen des funktional programmierten Prototyps in eine modulare, individuell testbare Struktur

Auf dem Weg der Weiterentwicklung des vorhandenen Service-Prototypen zu einem "qualitativen" Softwareprodukt muss sichergestellt werden, dass der Service diverse nicht-funktionale Anforderungen erfüllt. Hierzu zählen beispielsweise Anforderungen an die Wartbarkeit sowie die Änder- und Erweiterbarkeit des Services. Hier stößt die rein funktionale Programmierung an ihre Grenzen und es ist empfehlenswert, die entwickelte Anwendung im nächsten Entwicklungsschritt in lose-gekoppelte, modulare und damit individuell testbare Einheiten aufzuteilen und diese durch Softwaretests hinsichtlich eigens spezifizierter Anforderungen auf ihre Funktionalität zu überprüfen.

Auch in früheren Entwicklungsphasen des KI-basierten Services kann es sinnvoll sein, erste Tests zur Funktionalität des Services im Sinne des Test-Driven-Development (TDD) durchzuführen. Da sich die Funktionalität und der Aufbau des Services in dieser explorativen Phase jedoch meist im Stunden- oder Tagesrhythmus weiter verändert, bieten die Definition und der Aufbau umfangreicher Softwaretests in frühen Projektphasen meist noch keinen Mehrwert. Analog hierzu könnte auch in früheren Projektphasen bereits mit der objektorientierten Strukturierung des Softwarecodes begonnen werden. Allerdings erfordert dies von den Entwicklern fundierte Kenntnisse im Bereich der Softwareentwicklung, die von den meist primär mathematisch-fundierten Data Scientists nicht erwartet werden können. Hinzu kommt, dass schon die Tools selbst, die für explorative Analysen im Bereich Data Science häufig genutzt werden, wie beispielsweise Jupyter Notebooks, für die funktionale

Programmierung und weniger für die modulare Softwareentwicklung ausgelegt sind. Entsprechend empfiehlt es sich, die Überführung des Anwendungscode in losegekoppelte, modulare Strukturen in eine spätere Projektphase unter Unterstützung erfahrener Software-Entwickler zu verlagern.

b) Trennung von Anwendungs- und Infrastrukturlogik (inkl. Datenzugriffen)

Grundsätzlich ist es ratsam, die Logik der Anwendung und die der Infrastruktur (inklusive Datenzugriffe) des Services strikt voneinander zu trennen. Auch, wenn diese Strukturierung anfänglich mit einem höheren Zusatzaufwand zur Umsetzung einhergeht, führt sie langfristig zu zahlreichen Vorteilen, wie beispielsweise:

- Es ist möglich, mit der hohen Geschwindigkeit der Neuerungen in Bezug auf Datenspeicher und Performance Schritt zu halten, da die zugrunde liegenden Datenspeicher einfach austauschbar bleiben.
- Die Weiterentwicklung der Anwendungslogik und der Infrastruktur kann unabhängig voneinander erfolgen, was bedeutet, dass beispielsweise der Data Scientist seine Arbeit verbessern kann, ohne auf den Data Engineer angewiesen zu sein und umgekehrt.
- Falls zu einem späteren Zeitpunkt entschieden wird, dass die Anwendungslogik beispielsweise on the edge anstatt in der Cloud ausgeführt werden soll, kann diese einfach extrahiert werden. Im Idealfall, wenn die Laufzeitumgebung identisch ist, kann der Anwendungscode 1:1 wiederverwendet werden.

Es gibt verschiedene Ansätze, um diese Trennung zu erreichen. Im Folgenden sollen zwei davon genauer erläutert werden: Der Ansatz der Clean Architecture sowie die

Auslagerung der Anwendungslogik in eigenständige Softwarepakete.

b i.) Clean Architecture

Der Clean Architecture Ansatz definiert Organisationsprinzipien zur strukturellen Unterteilung eines Softwaresystems in die vier Komponenten "Entities", "Use-Cases", "Interface-Adapter" sowie "Frameworks & Drivers". Dies sollte sich auch in der Verzeichnisstruktur eines Softwareprojektes widerspiegeln. So sollten Repositories stets durch entsprechende Ordner und Unterordner die Unterteilung der genannten Komponenten abbilden. Durch Einsatz der Clean Architecture Prinzipien lässt sich die Trennung von Anwendungs- und Infrastrukturlogik realisieren. Darüber hinaus kann die Strukturierung der Repositories gemäß des Clean-Architecture-Ansatzes insbesondere in größeren Teams zu einer schnelleren Einarbeitung führen. Wird festgelegt, dass Repositories stets nach standardisierten Prinzipien organisiert werden, erleichtert dies die Navigation in neuen Repositories oder in anderen Projekten erheblich.

b ii.) Auslagerung von Algorithmenlogik in eigene Softwarepakete

Sobald dieselbe Algorithmenlogik für mehr als einen Service verwendet wird oder dies zukünftig in Betracht gezogen wird, empfiehlt es sich, diese Logik in ein eigenständiges Softwarepaket auszulagern. Im Unterschied zum zuvor beschriebenen Paradigma der Clean Architecture wird die Geschäftslogik in diesem Ansatz nicht in einen eigenen Unterordner eines Projektes deployed. Stattdessen wird sie in einem vollständig unabhängigen Code-Repository deployed, das als Softwarepaket bereitgestellt und über ein Artificactory für verschiedene Services zugänglich gemacht wird.

Die Verwendung eines solchen Pakets bietet den Vorteil einer Single Source of Truth und ermöglicht zudem die Kennzeichnung mit Versionsnummern (sog. version tags), sodass verschiedene Services unterschiedliche Versionen eines Softwarepakets nutzen können. Bei der Auslagerung von Logik in eigenständige Softwarepakete ist es wichtig sicherzustellen, dass keine Abhängigkeiten zu den zugrundeliegenden Daten im Softwarepaket enthalten sind. Das bedeutet, dass Eingabedaten in Form von Datensätzen zur Laufzeit übergeben werden sollten und das Paket selbst keine Funktionalität zum Lesen oder Schreiben von Daten besitzen sollte.

Hierbei ist es von Bedeutung, den Übergang zwischen den einzelnen Stages zu implementieren, wobei neben dem Einsatz umfangreicher Test-Frameworks vor allem die Prozesse hinter den Stages eine wichtige Rolle spielen. Bei der Verwendung von Stages wie "Development", "Testing", "QA" und "Produktion" ist es erforderlich, den Übergang auf Knopfdruck durchzuführen, sobald in einer Stage alle Tests erfolgreich bestanden wurden. Mindestens genauso wichtig sind jedoch klare Definitionen und Verantwortlichkeiten für die Prozesse, die innerhalb der einzelnen Stages getestet werden sollen, um ein reibungsloses und effektives Deployment zu gewährleisten.

c) Aufbau von CI/CD Pipelines inklusive Infrastruktur als Code

Das Ziel dieses Schrittes ist es, Änderungen am Code schnell, regelmäßig und unter intensiven Testmaßnahmen zu veröffentlichen.

Im beschriebenen Ansatz, bei dem die Algorithmenlogik in ein Softwarepaket ausgelagert wird, können die beiden CI/CD-Pipelines beispielsweise realisiert werden, wie in Abbildung 3 dargestellt.

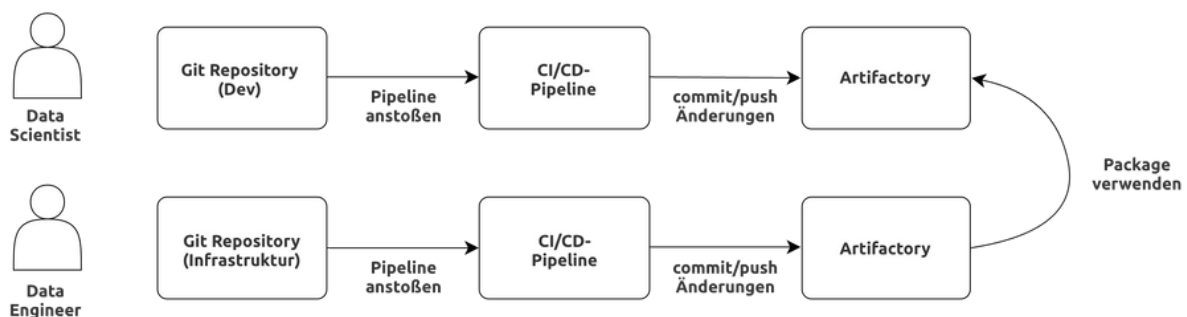


Abb. 3 CI/CD-Pipelines unter Einsatz zweier, dedizierter Repositories (Development / Infrastruktur)

In diesem Fall werden neue Versionen der Anwendung als auch der Infrastruktur mithilfe zwei eigenständiger CI/CD-Pipelines samt dedizierten Artifactories verwaltet und die Deployments somit technisch voneinander getrennt. Dieser Ansatz bietet den Vorteil, dass zum einen Änderungen an der Infrastruktur vorgenommen werden können, ohne dass die gesamte CI-Pipeline zur Formatierung, Testen und Bauen der Anwendung ausgeführt werden muss.

Analog hierzu können zum anderen aber auch Anpassungen der Funktionalität des Services durchgeführt werden und in ein Deployment Artifact geschrieben werden, ohne dass der bereits im Einsatz befindliche Code der Infrastruktur erneut deployed werden muss. Dieser Ablauf ist prozessual in Abbildung 3 dargestellt. Des Weiteren lassen sich durch diesen „Zwei-Repository-Ansatz“ Zugriffsberechtigungen granularer verwalten.

So kann beispielsweise festgelegt werden, dass sich die Entwickler, die Zugriffe auf das Infrastruktur-Repository besitzen, von jenen, die die Funktionalität des Services entwickeln, unterscheiden sollen und kein Entwickler über Zugriff auf beide Repositories verfügen darf.

d) Einsatz von Monitoring zur kontinuierlichen Überwachung der Servicefunktionalitäten

Sobald ein Service alle funktionalen und nicht-funktionalen Anforderungen erfüllt, kann er für den Einsatz im produktiven Umfeld bereitgestellt werden. Es gilt, die Funktionalität durch den Einsatz von Monitoringansätzen kontinuierlich zu überwachen, um somit Fehler bei der Anwendung des entwickelten Service frühzeitig zu erkennen.

Hierin liegt der essenzielle Unterschied des Softwaremonitoring zum Softwaretesting: Während Softwaretest einzelne Module oder die gesamtheitliche Funktionalität von Software während des Entwicklungsprozesses untersuchen, meint Monitoring den Prozess der Evaluierung von Softwarefunktionalitäten während des produktiven Einsatzes.

Grundsätzlich können im Produktivbetrieb von KI-basierten Services zwei Arten von Fehlern auftreten: Zum einen gibt es klassische Softwarefehler (Error), die bereits in der Entwicklung der Software entstanden sind, jedoch erst während der Laufzeit in Form eines Softwaredefekts (Defect, umgangssprachlich oft als "Bug" bezeichnet) in Erscheinung treten und im schlechtesten Fall zum Softwareversagen (Failure) führen. Solche Defekte sind auf explizite Fehler in der entwickelten Software zurückzuführen und müssen durch eine entsprechende Anpassung des Quellcodes korrigiert werden.

Im Gegensatz dazu gibt es sogenannte Software-Entscheidungsfehler, die sich nicht direkt auf Unstimmigkeiten in der entwickelten Software zurückführen lassen. Ursächlich ist ein Phänomen von KI-basierten Algorithmen, das sogenannte Model Drift (oft auch als Model Decay bezeichnet), welches sich ausschließlich durch aufmerksame Überwachung eingesetzter Bewertungsmetriken aufdecken lässt.

So äußern sich Software-Entscheidungsfehler durch eine im Laufe der Zeit gesunkene Genauigkeit der von KI-basierten Algorithmen getroffenen Klassifizierungen oder abgegebenen Vorhersagen. Damit führen sie zu einer meist signifikant schlechteren Performance hinsichtlich statistischer Metriken, wie etwa der Genauigkeit (Recall) oder der Präzision (Precision) im Falle einer Klassifizierung, bzw. des Mean Absolute Errors (MAE) oder des Mean Squared Errors (RMSE) im Falle einer Regression.

Ein Model Drift wird in der Regel durch ein zugrunde liegendes Konzeptdrift oder Datendrift oder auch durch Änderungen in der vorgelagerten Infrastruktur oder Datenpipelines verursacht.



Durch die Definition und Einrichtung geeigneter Maßnahmen und Messgrößen im Rahmen der Operationalisierung von ML-basierter Software (MLOps), lassen sich Model Drifts im produktiven Betrieb meist zuverlässig erkennen und rechtzeitig beheben.

5. Vom Service zum digitalen Endprodukt:

Meist steht am Ende der Entwicklung ein Service in digitaler Form, welcher den Kunden entweder in Kombination mit den bisher angebotenen, physischen Produkten, oder aber auch als individuelle Leistung verkauft werden kann. Um diesen Service jedoch als vollwertig marktreifes Produkt zu bewerben, gilt es zunächst, dessen Verfügbarkeit sicherzustellen. Anschließend ist es von enormer Relevanz, die Vertriebsmitarbeitenden hinsichtlich der Verknüpfung des digitalen Services als Ergänzungsleistung zu schulen und damit im

Sinne des Cross-Sellings zusätzliche Gewinne zu erzielen. Gleichzeitig aber auch darin, den Service als ein eigenständiges Produkt zu bewerben. Nur so lässt sich das digitale Endprodukt erfolgreich vermarkten.

3.3.2 Involvierte Rollen

So heterogen die Gegebenheiten bei verschiedenen Industrieunternehmen auch sind, so gilt dennoch, dass in der Regel eine bestimmte Auswahl an Rollen in einem Projektteam zur Entwicklung KI-basierter Services notwendig sind, um erfolgreich zu sein (siehe Abb. 2).

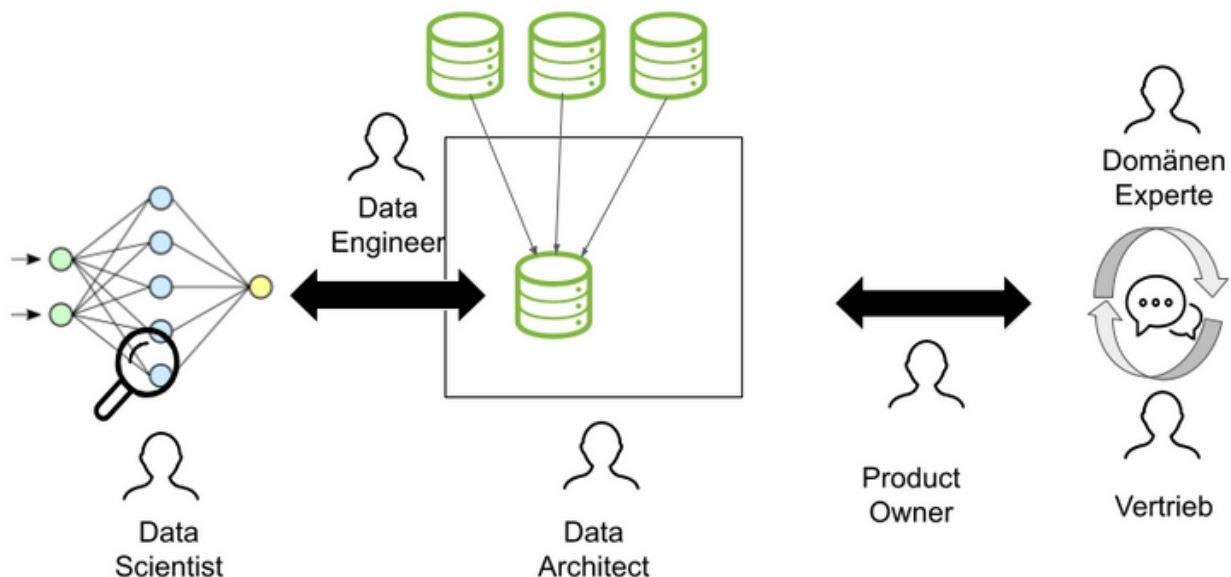


Abb. 4 Übersicht der Projektrollen zur Entwicklung von Daten- bzw. KI-basierten Services

Domänenexperte:

Dieser ist Expert:in für die durch den KI-Service zu lösende Problemstellung oder das zugehörige physische Produkt und vermittelt dieses Wissen im Projektteam. Er kennt betriebswirtschaftliche Zusammenhänge, technische Besonderheiten und ggf. bereits verfügbare Daten.

Product Owner:

Der Product Owner ist das Bindeglied zwischen den Data Experts und dem restlichen Projektteam sowie von außenstehenden Stakeholdern. Er kennt die Anforderungen an den KI-basierten Service, definiert To-Dos und priorisiert sie gemeinsam mit dem Entwicklungsteam. Des Weiteren sucht er die aktive Kommunikation mit Stakeholdern außerhalb des Projektteams.

Data Engineer:

Data Engineers sind für die Aufbereitung von Daten und Umsetzung von Datenpipelines verantwortlich. Sie bauen Datenplattformen auf und entwickeln diese weiter. Dies umfasst die Anbindung von Quellsystemen sowie die Modellierung und Aufbereitung der Daten auf unterschiedlichen Ebenen. Durch die Bereitstellung zusätzlicher Development-Umgebungen innerhalb der Datenplattformen fungieren Data Engineers außerdem als Enabler für explorative Datenanalysen durch Data Scientists, ohne dass gleichzeitig die Produktivsysteme durch Datenabfragen belastet werden. Ähnlich wie Data Scientists sind sie zudem mit der Arbeit im agilen Projektumfeld vertraut und bedienen sich diverser Werkzeuge der Software- und Datenentwicklung.

Data Architect:

Data Architects sind, im Gegensatz zu Data Engineers, primär konzeptionell tätig. Sie sind verantwortlich für den Entwurf ganzheitlicher Datenarchitekturen in Unternehmen. Hierzu definieren sie, aus welchen Quellen Daten bezogen werden und wie diese zu integrieren sind. Des Weiteren sind sie auch für die Account- und Benutzerrechteverwaltung zur Zugriffssteuerung auf unternehmensweit verfügbare Datenressourcen verantwortlich. Data Architects besitzen fundierte Kenntnisse im Bereich des Informationsmanagements und der Datenmodellierung.

Data Scientist:

Data Scientists sind für die explorative Analyse von Daten und die Entwicklung von ML-Modellen verantwortlich. Damit decken sie Zusammenhänge und Strukturen in den Daten auf und verknüpfen diese mit dem jeweiligen Wissen über die fachliche Domäne. Außerdem unterstützen sie bei der Operationalisierung von ML, indem sie ihre entwickelten Algorithmen und Modelle in eine produktive Laufzeitumgebung überführen, um diese auf neue, bisher unbekannte Datenpunkte anzuwenden (Inference). Dabei greifen sie sowohl auf Projektmethoden der agilen Softwareentwicklung, wie Scrum und Kanban, als auch auf Tools zur Entwicklung und Versionierung von Software, wie Jupyter, PyCharm, Git, Bitbucket sowie Programmiersprachen, wie SQL und Python zurück.

Vertriebsmitarbeitende:

Oft unterschätzt, nehmen die Vertriebsmitarbeitenden eine zentrale Rolle bei der Entwicklung KI-basierter Services ein. Mit ihrem Wissen über Kundenbedürfnisse und Marktanforderungen können sie dazu beitragen, dass das entwickelte digitale Produkt sowohl als Ergänzungsleistung als auch als eigenständiges Produkt die vorhandenen Bedürfnisse deckt. Gleichzeitig lernen die Mitarbeitenden im Vertrieb durch ihre Einbindung in das Entwicklungsprojekt die Bedeutung von KI-basierten Services kennen und verstehen, wie sie diese sowohl durch Cross-Selling als auch Up-Selling vermarkten können.

3.4 Zusammenfassung und Ausblick

In diesem Kapitel haben Sie einen Einblick in die potenziellen Einsatzgebiete (Use-Cases) von Industrial Analytics bekommen. Es zeigt sich, dass die grundlegende Vorgehensweise und Rollen sich unabhängig vom späteren Anwendungsfall standardisieren lassen.

Allerdings ist es sowohl im Hinblick auf die zur Verfügung stehenden Daten und Bewertungskriterien, als auch für die Auswahl des richtigen Algorithmus unerlässlich, sich bei der Umsetzung eines Industrial Analytics bereits frühzeitig um den konkreten Use Case Gedanken zu machen.

Sind die Vorgehensweisen, Methoden und Rollen in einem Unternehmen für den Einsatz von Industrial Analytics jedoch einmal implementiert, so ist das Unternehmen zukünftig in der Lage, weitere Use Cases effizient und schnell umsetzen zu können.

Jedoch ist mit der Umsetzung eines Use-Cases noch nicht die ganze Arbeit getan. Sobald ein KI-Service mal im produktiven Betrieb ist, stellt sich auch schon wieder die Frage, wie man ohne zusätzlichen, händischen Mehraufwand dafür sorgen kann, dass ML-Modelle stets zuverlässig zur Verfügung stehen und gleichzeitig bei Vorhandensein neuer Datensätze weiter trainiert werden.

In unserem Whitepaper zum Thema MLOps, das wir Ihnen zeitnah zur Verfügung stellen, erklären wir, welche Praktiken und Werkzeuge es braucht, um produktiv genutzte Modelle für maschinelles Lernen fachgerecht zu warten sowie effizient und zuverlässig im Betrieb zu halten. Seien Sie gespannt.

Kontakt

Wir hoffen, dass Sie einige Erkenntnisse durch unser Whitepaper gewinnen konnten.

Sie haben Fragen oder möchten die Vorteile für Ihr Unternehmen diskutieren?

Ich freue mich, von Ihnen zu hören!



SIMON KNELLER

Head of Industrial
Analytics & IoT



+49 160 96764804



simon.kneller@esentri.com

Wir gestalten Zukunft – digital, nachhaltig und menschenzentriert.

Digitalisierung bedeutet für uns nicht einfach die Modernisierung des Bestehenden, sondern die Zukunft neu zu gestalten. Deshalb haben wir es uns zum Ziel gemacht, unsere Kunden bei der Digitalisierung der gesamten Wertschöpfungskette nachhaltig zu unterstützen und dabei vor allem eines in den Mittelpunkt zu setzen: den Menschen.

Unser Firmenname esentri setzt sich aus 2 Bestandteilen zusammen: **easy und entry**. Das spiegelt den Kern unserer Vision wider, denn wir schaffen unseren Kunden einen einfachen Zugang zur nachhaltigen Digitalisierung.

Wir denken digital

Gemeinsam mit unseren Kunden erarbeiten wir Lösungen, die das Digitale aus verschiedenen Perspektiven betrachten. Diese Denkweise ist Teil unserer DNA und macht sich in unserem Handeln zu jedem Zeitpunkt bemerkbar.

Wir handeln nachhaltig

Nachhaltiges Handeln bedeutet für uns viel mehr als Ökologie, denn für uns hat Nachhaltigkeit drei Perspektiven: die ökonomische, die ökologische und die soziokulturelle. Bei unserer Arbeit legen wir das Augenmerk darauf, all diesen Aspekten gerecht zu werden.

Im Mittelpunkt steht der Mensch

Nachhaltige Digitalisierung im Sinne des Menschen. Das ist kein Slogan, sondern unsere Antriebskraft. In unserer Zukunftswerkstatt arbeiten wir gemeinsam mit unseren Kunden an Lösungen, die eine nachhaltige Veränderungen für den Menschen schaffen.

KONTAKT

ESENTRI AG
PFORZHEIMER STRASSE 128B
76275 ETTLINGEN
MARKETING@ESENTRI.COM