Hochschule Darmstadt

Fachbereiche Mathematik und
Naturwissenschaften & Informatik

Echtzeit Gerätedaten Nachrichtenübermittlung als Grundlage für datengetriebene Lösungen wie das IIoT.

Abschlussarbeit zur Erlangung des akademischen Grades

Master of Science (M.Sc.)

im Studiengang Data Science

vorgelegt von

Lennart Severin

Referent	:	Prof. Dr. Michael von Rüden
Korreferent	:	Prof. Dr. Jutta Groos

Ausgabedatum	:	12.04.2021
Abgabedatum	:	24.09.2021

ERKLÄRUNG

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Freiburg im Breisgau, 24. September 2021

Lennart Severin

ZUSAMMENFASSUNG

Die kontinuierlich steigende Abhängigkeit industrieller Unternehmen von digitalen Infrastrukturen führen zur Notwendigkeit, disparate und abhängige Applikationen in einer einheitlichen Struktur zu verbinden. Enterprise Application Integration ist ein Prozess um Systeme in eine Struktur zu integrieren, sodass Informationen und Ressourcen geteilt werden können.

Zur Entkopplung von Applikationen in der bestehenden Infrastruktur ist *Musterunternehmen (MU)* an einer eventgetrieben und nachrichtenbasierten Integrationsarchitektur interessiert. In dieser Arbeit wird eine solche Architektur konzipiert und durch Lasttests auf ihre Echtzeitfähigkeit und Skalierbarkeit geprüft. Zusätzlich werden Änderungen im Lebenszyklus von *MU*-Geräten, die zu duplikativen Einträgen im Datenbestand führen, erkannt, eingeordnet und für das weitere Prozessieren zur Verfügung gestellt.

Die Konzeptionierung der Integrationsarchitektur erfolgt durch eine Analyse der bestehenden Infrastruktur und Selektion einer geeigneten Enterprise Messaging-Lösung mithilfe einer Nutzwertanalyse. Zur Einordnung von Duplikaten werden Hypothesen mittels Stichprobenuntersuchungen über Zusammenhänge der Merkmale und Unterschiede der Ausprägungen formuliert. Durch eine Faktoranalyse und einem Clustering-Verfahren werden die aufgestellten Hypothesen überprüft und evaluiert. Um die Echtzeitfähigkeit und Skalierbarkeit der erarbeiteten Architektur zu belegen, werden Lasttests durchgeführt. Dabei wird die Lösung anhand steigender Nachrichten pro Minute und parallel aktiven Nachrichtenkonsumenten getestet.

Es konnte gezeigt werden, dass die entwickelte Architektur skalierbar und in allen Variationen echtzeitfähig mit Latenzzeiten von unter 20 Millisekunden ist. Die Architektur weist zudem eine hohe Kompatibilität für die bestehende Infrastruktur auf. Mit einer Übereinstimmung von 87.5% zwischen der Hypothese und dem Clustering-Verfahren, konnten 12.16% aller relevanten Duplikate als automatisch zusammenführbar identifiziert werden. Zusammenfassend ermöglicht die entwickelte Architektur und Datenverarbeitung eine nahtlose Einbindung in bestehende Integrationsprozesse und eine Minimierung duplikativer Einträge für den zukünftigen Datenbestand.

Schlagworte— Enterprise Application Integration, Enterprise Messaging, Editierabstand, Faktoranalyse, K-Means Clustering

ABSTRACT

Due to companies being increasingly dependent on their technological infrastructure, the need to connect disparate and dependent applications into a unified structure is growing. Enterprise Application Integration is a process of integrating systems into a structure so that information and resources are shared.

In order to decouple applications within their existing infrastructure, *Musterunternehmen (MU)* is interested in an event-driven and message oriented integration architecture. The aim of this work is the conceptual design and evaluation of a real-time capable and scalable integration architecture. Furthermore, life cycle changes in *MU* devices, that lead to duplicative entries in databases, are recognised, categorized and made available for further processing.

The conception of the architectural design is based on an analysis of the existing infrastructure and a selection of a suitable Enterprise Messaging solution using a decision analysis. The categorization of duplicative entries is carried out by formulating hypotheses based on sample examinations of correlations between features and differences in characteristics. The hypotheses are tested and evaluated using factor analysis and a clustering algorithm. Load tests are carried out to evaluate the real-time capability and scalability of the architecture, based upon increasing messages per minute as well as parallel consumers.

The results show that the developed architecture is scalable, real-time capable in all variations with latencies below 20 milliseconds and highly compatible with the existing infrastructure. Moreover, an agreement of 87.5% between the hypothesis and the clustering method is reached, identifying 12.16% out of all relevant duplicates as automatically mergeable. In summary, the developed architecture and data processing method allows for seamless integration into existing integration processes and a minimization of duplicate entries for future data.

Keywords— Enterprise Application Integration, Enterprise Messaging, Edit Distance , Factor Analysis, K-Means Clustering

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

TABELLENVERZEICHNIS

LISTINGS

ABKÜRZUNGSVERZEICHNIS

API	Application Programming Interface
EAI	Enterprise Application Integration
P/S	Publish-Subscribe
PTP	Point-to-Point
BDH	Business Data Hub
EM	Enterprise Messaging
PaaS	Platform-as-a-Service
WSS	Web-Socket-Subscription
BDL	Business Data Lake
BTP	Business Technology Platform
DI	Data Intelligence
CPI	Cloud Platform Integration
Npm	Nachrichten pro Minute
RU	Ressource Unit
IoT	Internet of Things
QoS	Quality of Service
HTTP	Hypertext Transfer Protocol
MQTT	Message Queuing Telemetry Transport
AMQP	Advanced Message Queueing Protocol
MU	Musterunternehmen

Teil I

THESIS

1 EINLEITUNG

1.1 MOTIVATION

Das konstante Wachstum der Digitalisierung in allen Bereichen eines Unternehmens, stellt diese vor immer größer werdende Herausforderungen. Systeme können dezentral, von unterschiedlichen Anbietern und für diverse Zwecke wie die Datenverarbeitung oder -speicherung genutzt werden. Wodurch die Integration dieser Systeme ein hohes Maß an Komplexität beinhaltet.

Im Allgemeinen lassen sich produzierende Unternehmen in mindestens drei Bereiche gliedern, Lager und Produktion, Einkauf und Vertrieb und Personal. Der Vertrieb nimmt Bestellungen entgegen und versendet Produkte und Rechnungen. Im Einkauf wird mit dem Lager und der Produktion interagiert um Materialien, Ersatzteile oder Teilprodukte zu bestellen und zu verrechnen. Mitarbeiter verteilen und bearbeiten Aufgaben und erhalten einen Lohn. All diese Tätigkeiten werden durch unterschiedliche Systeme und Oberflächen unterstützt.

Bei der Integration ist der wichtigste Aspekt, dass der Integrationsfluss zwischen unterschiedlichen Entitäten durchgängig und optimiert abläuft. Entitäten können Kunden, Zulieferer oder auch interne Abteilungen sein.

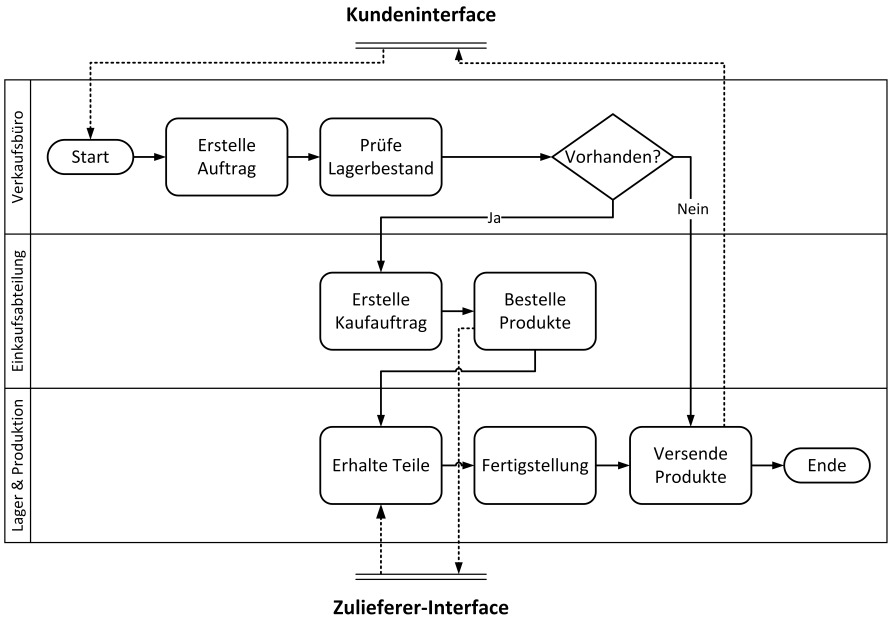


Abbildung 1.1: Vereinfachtes Beispiel eines Bestell- und Fertigungsprozesses.

In Abb. 1.1 ist ein solcher Prozess als vereinfachtes Beispiel aufgezeigt. Der dargestellte Prozess benötigt Informationen und Prozesse unterschiedlicher Bereiche des Unternehmens.

Über das Kundeninterface wird ein Kaufantrag für ein bestimmtes Produkt gestellt. Zunächst wird im Lager überprüft, ob das geforderte Produkt bzw. die erforderlichen Bauteile für dieses Produkt vorhanden sind. Sollte dem nicht so sein, wird ein neuer Prozess in der Einkaufsabteilung gestartet, es werden die benötigten Teile bei einem Zulieferer bestellt. Nach Erhalt der Teile kann das Produkt fertiggestellt, versandt und abschließend eine Bestätigung an den Käufer gesendet werden.

Selbst in einem simplen Prozess wie diesem, ist ein Informationsfluss zwischen diversen Abteilungen, Systemen und Prozessen notwendig:

1. Das Verkaufsbüro erhält eine Anfrage aus dem Kundenportal und erstellt einen Auftrag.
2. Es wird geprüft, ob das geforderte Produkt bzw. die erforderlichen Bauteile für dieses Produkt im Lager vorhanden sind. Verwendet wird hierzu ein Lagersystem.
3. Sollte das Produkt nicht vorhanden sein, erstellt die Einkaufsabteilung einen Kaufauftrag im Buchhaltungssystem und bestellt die nötigen Teile.
4. In dem Lager werden die neuen Teile des Zulieferers in das Lagersystem gebucht und bei Fertigstellung angepasst.
5. Das Packen und Versenden des Produktes kann ein eigenes System sein.
6. Nach Versand des Produktes wird eine Rechnung in dem Buchhaltungssystem erzeugt, die der Kunde im Kundeninterface erhält.

Wie in dem Beispiel in Abb. 1.1 illustriert, sind bei jedem Geschäftsprozess diverse heterogene Systeme involviert, um diesen Prozess zu unterstützen. In diesem Fall wird ein iterativer Prozess aufgezeigt. Die unterschiedlichen Systeme benötigen die Informationen des vorherigen Systems, um eigene Operationen durchzuführen. Die Integration dieser Systeme begünstigt nicht nur den Datenfluss, sondern ermöglicht auch eine Automatisierung des Vorgehens.

M. Fowler definiert *Enterprise Application Integration (EAI)* als [1]:

"Enterprise Integration is the task of making disparate applications work together to produce a unified set of functionality."

EAI versetzt folglich Applikationen in die Lage, Daten und Prozesse zu teilen ohne maßgebliche Eingriffe in die Applikationen oder Datenstrukturen durchzuführen [2].

1.2 ZIEL DER ARBEIT

Ziel dieser Arbeit ist es, basierend auf den Prinzipien von *EAI*, eine entkoppelte, eventgetrieben und nachrichtenbasierte Integrationsarchitektur zu entwerfen. Diese Architektur soll in die bestehende Infrastruktur von *Musterunternehmen (MU)* integrierbar sein. Weiterhin sollen Änderungen im Lebenszyklus von Geräten, welche von *MU* produziert werden, erkannt, eingeordnet und zur Verfügung gestellt werden. Solche Änderungen können beispielsweise die Produktion, der Verkauf oder eine Qualitätskontrolle eines Geräts sein und werden in dieser Arbeit als Events bezeichnet.

Zur Umsetzung dieses Ziels wird zunächst die bestehende Infrastruktur analysiert und eine geeignete nachrichtenbasierte Lösung identifiziert.

Anschließend werden vorhandene Events, welche zu duplikativen Einträgen im Datenbestand führen, analysiert und mittels ausgewählter statistischer Methode eingeordnet. Im Fokus stehen dabei Duplikate, die auf einen systematischen Fehler oder einen Tippfehler zurückzuführen sein könnten. Es soll bestimmt werden, ob Abweichungen zwischen den Duplikaten so klein sind, dass eine automatische Zusammenführung angebracht ist.

Schließlich wird eine Architektur konzipiert und Methoden zur Publizierung von Events identifiziert. Die erarbeitete Methode zur Einordnung der Events wird in der Architektur implementiert und die Ergebnisse zur Initiierung weiterer Prozessschritte persistiert. Darüber hinaus wird die entwickelte Architektur auf ihre Echtzeitfähigkeit und Skalierbarkeit unter verschiedenen Last- und Verarbeitungsszenarien geprüft. Diese Prüfung wird anhand dreier Testreihen durchgeführt.

Die drei Schwerpunkte lassen sich wie folgt zusammenfassen:

1. Entwicklung einer in die bestehende Infrastruktur integrierbaren, eventgetrieben und nachrichtenbasierten Integrationsarchitektur.
2. Analyse und Einordnung von Duplikaten anhand ausgewählter statistischer Methoden.
3. Umsetzung einer Architektur anhand eines Anwendungsfalles und die Prüfung der Echtzeitfähigkeit und Skalierbarkeit unter verschiedenen Last- und Verarbeitungsszenarien.

1.3 GLIEDERUNG

In Kapitel 2 werden die Grundlagen von *EAI* und die statistischen Methoden erläutert, welche in dieser Arbeit verwendet werden. In Kapitel 3 wird die aktuelle Systemlandschaft vorgestellt. Mithilfe einer Nutzwertanalyse wird eine geeignete, nachrichtenbasierte Lösung ausgewählt. Anschließend wird die Analyse und Einordnung der Duplikate durchgeführt. Zuletzt wird ein Konzept für die erarbeitete Architektur vorgestellt. Kapitel 4 untersucht das vorgestellte Konzept anhand dreier Testreihen. Es wird unter verschiedenen Bedingungen geprüft, ob die Architektur echtzeitfähig und skalierbar ist. Abschließend erfolgt in Kapitel 5 eine Zusammenfassung dieser Arbeit gefolgt von einem Ausblick über Folgeuntersuchungen.

2 GRUNDLAGEN

In diesem Kapitel werden Grundlagen zu *Enterprise Application Integration (EAI)*, sowie ausgewählte statistische Methoden vorgestellt.

2.1 ENTERPRISE APPLICATION INTEGRATION

EAI sollte jeden Aspekt der digitalen Infrastruktur eines Unternehmens wie die Architektur, verfügbare Hardware, Software und die bereits vorhandenen Prozesse berücksichtigen [3]. Darüber hinaus sollte für die Integration von Unternehmensprozessen bekannt sein, was die Unternehmensprozesse ausmacht und inwieweit sie automatisierbar sind.

Abhängig von dem zugrunde liegenden Ziel der Integration wie beispielsweise dem Verschieben von Daten oder einem Prozessaufruf, werden unterschiedliche Verfahren oder Technologien zur Umsetzung benötigt. Data-, Application Interface-, Method- und User Interface-Level sind vier Integrationstypen die in *EAI* Anwendung finden. Für diese Arbeit sind insbesondere Data- und Application Interface-Level relevant und werden im Folgenden erläutert [4].

Data-Level beschreibt den Prozess des Datenverschiebens. Dabei werden die Datenbanksysteme der involvierten Applikationen über Pull oder Push Methoden integriert. Applikationen können folglich Daten im Datenbanksystem einer anderen Applikation anfragen und in der eigenen Datenbank persistieren.

Application Interface-Level verwendet die bereits bestehende Integrationsinfrastruktur einer Applikation über bestehende Integrationskanäle oder eine *Application Programming Interface (API)*. Dadurch kann eine Applikation Zugriff auf verwendete Methoden und Daten ermöglichen, ohne dass anfragende Applikationen identische Maßnahmen ergreifen müssen.

Für Informationen zu den Typen Method- und User Interface-Level wird auf [4] und [1] verwiesen.

2.1.1 *Integrationsarchitekturen*

Unabhängig von dem *EAI* Typ, stehen für den Daten- und Informationsfluss zwei Integrationsarchitekturen zur Verfügung: Point-to-Point oder Middleware basierte [2].

Point-to-Point

Bei einer *Point-to-Point (PTP)*-Architektur werden die Applikationen direkt miteinander verbunden. Es gilt jedoch zu beachten, dass für eine steigende Anzahl an Applikationen, die Zahl der Verbindung wächst.

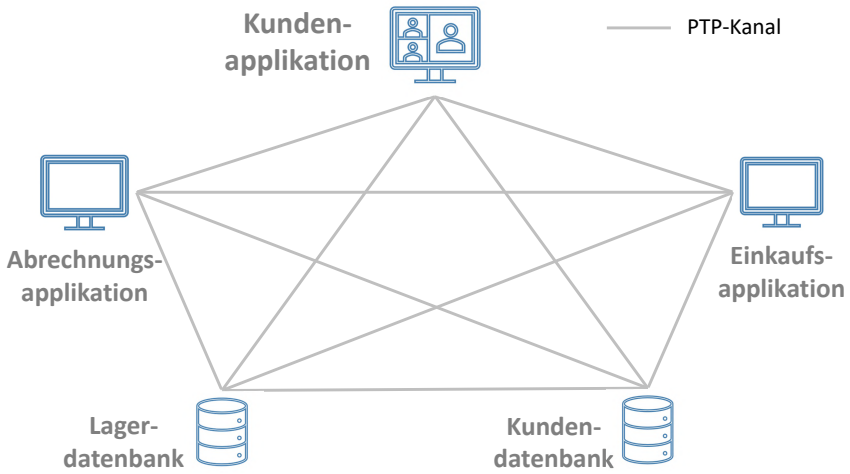


Abbildung 2.1: Applikationsverknüpfung Point-to-Point

Der in Abschnitt 1.1 dargelegte Geschäftsprozess erfordert eine Kundendatenbank, eine Kundenapplikation, eine Einkaufsapplikation, eine Lagerdatenbank sowie eine Applikation zur Rechnungserfassung. Um alle Applikationen miteinander zu integrieren werden 10 Integrationskanäle benötigt, siehe Abbildung 2.1. Dies entspricht einem vollständigen Graphen, woraus für weitere Applikationen nach der Graphentheorie für die Anzahl der Kanten folgt: Sei n die Anzahl der Knoten des vollständigen Graphen K_n, dann gilt für die Anzahl der Kanten [5]:

$$n(n-1)/2 \tag{2.1}$$

Es ist leicht zu erkennen, dass die Skalierbarkeit dieser Architektur begrenzt ist. Um diese Einschränkung zu umgehen, kann eine Vermittlungsschicht implementiert werden.

Middleware

Die Middleware hat die Aufgabe als Vermittler zwischen Applikationen zu fungieren.

Abb. 2.2 zeigt, dass beispielsweise ein Nachrichtenkanal als Middelwaretechnologie eingesetzt werden kann, um die Kommunikation zwischen Applikationen und Datenbanken zu steuern. Diese Technologie ist erweiterbar und fundamental für diese Arbeit. Aufgrund der Bedeutsamkeit dieser Technologie, folgt in Abschnitt 2.1.2 eine detaillierte Einführung.

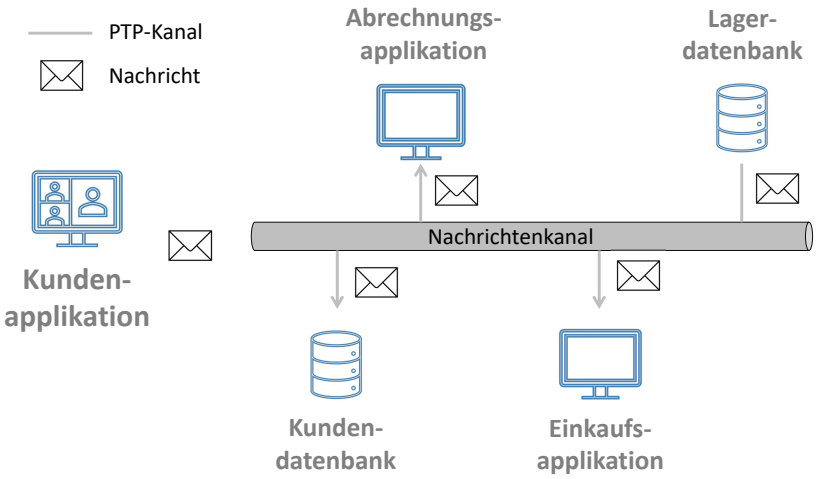


Abbildung 2.2: Applikationsverknüpfung über einen Message Bus

2.1.2 *Middleware*

Innerhalb von *EAI* fungiert die Middleware als die Technologie, welche Informationen und Logik zwischen Applikationen steuert. Folglich kann die Middleware als Schicht zwischen Applikationen und Betriebssystemen aufgefasst werden. Mittels Middleware erfolgt eine Transformation der Komplexität für eine möglichst vereinfachte Nutzung der zugrunde liegenden Betriebssysteme und Netzwerke. Verbindungen können dabei in den Formen One-to-One, One-to-Many oder Many-to-Many auftreten [4].

Integrationskriterien

Die Auswahl der zugrundeliegenden Technologie basiert auf multiplen Kriterien, die bei der Integration zu beachten sind. Im Folgenden werden insbesondere die Kriterien benannt, die für diese Arbeit relevant sind [2]:

APPLIKATIONSVERKNÜPFUNG Bei der Integration von Applikationen gilt es die Abhängigkeit der Applikationen zueinander zu berücksichtigen. Eng gekoppelte Applikationen setzen Annahmen über die verbundene Applikation voraus, welche zu Problemen bei Änderungen einer Applikation führen können.

TECHNOLOGIESELEKTION Verschiedene Integrationstechniken erfordern meist unterschiedliche Hardware. Dabei gilt es zu berücksichtigen, dass Abhängigkeiten zu den ausgewählten Technologie-Anbietern entstehen können.

DATENFORMAT Integrierte Applikationen müssen ein gemeinsames konzeptionelles Datenformat akzeptieren. Dieses Datenformat muss um neue Datenfelder und -typen erweiterbar sein, was bspw. durch Extensible Markup Language oder JavaScript Object Notation realisiert werden kann.

DATENAKTUALITÄT Der Integrationsmechanismus sollte die Zeit zwischen Bereitstellung der Daten durch eine Applikation und das Konsumieren dieser Daten von einer anderen Applikation minimieren. Zu beachten ist, ob es sich bei der Bereitstellung um Batch oder Live Daten handelt.

Unter Berücksichtigung dieser Kriterien erfolgt die Wahl der Middleware Technologie.

Middleware Technologien

Anhand der Komplexität des Integrationsvorganges wird differenziert, welche Art von Integrationsmechanismus notwendig ist. Die Middleware besteht hierfür meist aus einer oder multiplen Technologien, welche einen Lösungsansatz für unterschiedliche Probleme bieten. Die im Folgenden aufgeführten Technologien basieren aufeinander und können jeweils als Integrationsmuster aufgefasst werden [2]:

FILE TRANSFER Jede Applikation produziert und nutzt geteilte Dateien. Diese Dateien können unter Berücksichtigung des Datenformat-Kriteriums in unterschiedlichen Formaten, wie Extensible Markup Language, JavaScript Object Notation oder Comma-separated values gemeinsam genutzt werden.

SHARED DATABASES Jede Applikation in dem Integrationsprozess greift auf eine gemeinsame Datenbank zu.

REMOTE PROCEDURE CALL Jede Applikation stellt interne Prozesse für andere Applikationen zur Verfügung, sodass die internen Prozesse extern aufgerufen werden können. Ziel dieses Musters ist es, dass eine Applikation eine Funktion einer anderen Applikation aufruft, Daten bereitstellt, die Prozessierung definiert und eine Funktion zur Bestätigung des Transfers invokiert.

MESSAGING Jede Applikation ist mit einem gemeinsamen Messaging System verbunden. Dieses steuert den Datenaustausch zwischen den Applikationen.

In dieser Arbeit wird insbesondere die Messaging Technologie herangezogen. Daher folgt in Abschnitt 2.1.3 eine detaillierte Beschreibung dieser Technologie. Für weiterführende Informationen bezüglich dem File Transfer, Shared Databases sowie Remote Procedure Calls wird auf [1] und [2] verwiesen.

2.1.3 *Messaging*

Nach Abschnitt 2.1.2, kann die enge Kopplung von Applikationen bei verteilten Systemen zu Komplikationen in der Integration führen. Die asynchrone Messaging-Technologie ermöglicht Applikationen eine Kommunikation ohne auf den Konsumenten der Message warten zu müssen.

Wie in Abb. 2.2 und Abb. 2.1 dargestellt, beinhalten Middleware Systeme wiederkehrende Komponenten wie beispielsweise unterschiedliche Channel oder Messages. Diese und weitere Komponenten werden als Pattern bezeichnet. Im Folgenden werden ausgewählte Pattern kurz erläutert [2]:

CHANNELS Ein Channel verbindet den Sender der Message mit dem Empfänger. Die Art des Channels muss basierend auf den Applikationen festgelegt werden. Möglich sind hierbei *PTP* oder *Publish-Subscribe (P/S)*.

MESSAGES Messages (Nachricht) sind Datenpakete die über einen Channel verschickt werden.

PIPES AND FILTERS Pipes und Filter sind für komplexere Kommunikationsszenarien geeignet. Im einfachsten Fall wird die Nachricht unverändert an den Empfänger gesendet. Pipes und Filter werden eingesetzt, sofern eine Transformation oder Validierung durchgeführt werden muss, bevor die Nachrichten dem Empfänger gesendet wird.

ROUTING In komplexen Systemen können unterschiedliche Applikationen bei der Nachrichtenübertragung involviert sein. Ein Message Router stellt sicher, dass der Sender und Empfänger die Zwischenwege nicht bekannt sein müssen und steuert den Kommunikationsfluss.

TRANSFORMATION Unterschiedliche Applikationen benötigen gegebenenfalls unterschiedliche Nachrichtenformate. Der Message Translator bildet Dateiformate aufeinander ab und stellt somit die Lesbarkeit der Nachricht sicher.

ENDPOINTS Endpunkte ermöglichen der Applikation, das Empfangen und Versenden von Nachrichten.

In dieser Arbeit steht der *P/S*-Channel im Fokus und wird im Folgenden näher erläutert.

2.1.3.1 *Channel*

Die zwei Schlüsseltechnologien des Channel (Kanal), um den Datenverkehr zu steuern, sind *Point-to-Point (PTP)* und *Publish-Subscribe (P/S)*. *PTP* stellt eine Direktverbindung zwischen Produzent und Konsument der Nachricht her. *P/S* hingegen ermöglicht einem einzelnen Produzenten, Nachrichten an *n* Konsumenten zu übermitteln, wobei n unbekannt Groß sein kann [6]. Im Folgenden werden beide Technologien näher erläutert [7].

Point-to-Point

Neben der Verbindung zweier Applikationen, beinhaltet ein *PTP*-Kanal einen weiteren Mechanismus. Eine Queue bewahrt die Nachrichten die mithilfe des Kanal ausgetauscht werden auf und ermöglicht eine asynchrone Kommunikation. Die Nachrichten werden solange aufbewahrt bis ein Überlauf der Queue oder eine vorzeitige Löschung erfolgt.

Dabei gibt es unterschiedliche Prinzipien nach der eine Nachricht abgearbeitet wird bspw. First-in First-out, Last-in First-out oder random. Es wird nach der Reihenfolge des Zugriffs unterschieden.

PTP Messaging nutzt einen dedizierten Kanal für den Datentransfer zwischen dem Sender und dem Empfänger. Die Art der Transaktion und damit die Qualität des Datentransfers muss berücksichtigt werden. Transaktionen können in *PTP*-Systemen auf zwei Arten verarbeitet werden:

- Die Transaktion garantiert, dass die Nachricht in die Queue geschrieben wurde.
- Die Transaktion garantiert, dass die Nachricht von dem Zielsystem empfangen wurde. Dies erfordert eine Empfangsbestätigung des Zielsystems.

Publish-Subscribe Channel

Ein *P/S*-Kanal arbeitet folgendermaßen: Eine Nachricht die eine publizierende Applikation (Publisher) in den *P/S*-Kanal sendet, wird abhängig von der Anzahl der subskribierten Applikationen (Subscriber) vervielfältigt. Der *P/S*-Kanal sendet diese Kopien mithilfe dedizierter *PTP*-Kanäle an jeden Subscriber. Diese können jede Nachricht genau einmal empfangen. Abschließend wird die Nachricht von dem *P/S*-Kanal gelöscht.

2.1.4 *Publish-Subscribe Paradigma*

Das *P/S*-Paradigma nutzt einen *P/S*-Kanal, um multiplen Applikationen Nachrichten zukommen zu lassen.

Ziel ist es, einen *P/S*-Kanal für ein dediziertes Ereignis (Event) oder eine bestimmtes Thema (Topic) zur Verfügung zu stellen. Ein Event kann eine neue Kundenanfrage oder auch ein neuer Datensatz in der Datenbank sein. Ein Topic kann bspw. ein bestimmter Kunde oder eine Datenbanktabelle sein. Es kann Push oder Pull Technologie verwendet werden. Die Nachrichten können von dem *P/S*-System übermittelt oder von einem Subscriber angefragt werden. Dabei kann die Queue Mechanismen nutzen wie [4]:

ONLY-ONCE Die Nachricht wird von der Queue entfernt, nachdem sie einmalig konsumiert wurde.

MESSAGE EXPIRATION Die Nachricht wird nach einem bestimmten Zeitraum von der Queue gelöscht. Dieser Typ wird vorwiegend zur Prävention eines Speicherüberlaufs verwendet und garantiert nicht, dass die Nachricht von der Applikation konsumiert wurde.

P/S-Lösungen können mit folgenden Methoden implementiert werden [8]:

HUB-N-SPOKE Diese Methode nutzt einen zentralen Knotenpunkt (Hub), welcher als Nachrichtenverteiler fungiert. Als Spoke wird ein Adapter bezeichnet.

Dieser Adapter stellt die Verbindung, Sicherheit und Datenformate von Applikationen zu dem Knotenpunkt sicher. Der Knotenpunkt ist verantwortlich für das Kopieren und Verteilen von Nachrichten an interessierte Subscriber. Nachrichten an inaktive Applikationen werden hält der Adapter mithilfe einer Queue vor. Durch den Adapter können Inhalte einer Queue gelöscht werden.

MESSAGE BUS Ein Producer übermittelt eine Nachricht an einen Message Bus. Dieser vervielfältigt und übermittelt die Nachricht an alle im Netzwerk verfügbaren Subscriber. Im Gegensatz zu der Hub-n-Spoke Methode müssen bei einem Message Bus die Subscriber nicht benötigte Nachrichten verwerfen. Es werden alle übermittelten Nachrichten verteilt.

Eine für diese Arbeit relevante Methodik in Verbindung mit den genannten Verfahren ist eine *Web-Socket-Subscription (WSS)*. Dabei wird eine aktive *PTP*-Verbindung zwischen dem Server und einer Applikation aufgebaut (Socket). Eine sichere Verbindung wird mithilfe eines Handshake aufgebaut. Dazu wird eine Identifikation und Authentifizierung der involvierten Systeme durchgeführt. Solange die Verbindung aktiv ist, sind keine weiteren Authentifizierungsschritte nötig. Dieses Verfahren kann als Push Variante für die Verbindung zwischen einem *P/S*-System und den Subscribern verwendet werden.

2.2 MATHEMATISCHE GRUNDLAGEN

In diesem Kapitel werden ausgewählte statistische Methoden und Begriffe eingeführt.

Zunächst werden in Abschnitt 2.2.1 einige Grundbegriffe der Lage- und Streuungsmaße erläutert.

Anschließend werden in Abschnitt 2.2.2 ausgewählte statistische Tests vorgestellt. Alle vorgestellten Tests werden bei der Auswertung der Testreihen in Kapitel 4.2 angewandt.

Die in Abschnitt 2.2.3 vorgestellten Distanzmaße und die in Abschnitt 2.2.4 erläuterten Methoden werden in Kapitel 3.2 verwendet. Ziel dieser Methoden ist es, Daten unterschiedlicher Merkmalsausprägungen zu transformieren und partitionieren um eine Kategorisierung zu ermöglichen.

2.2.1 *Lage- und Streuungsmaße*

Lagemaße geben verschieden gewichteten Verteilungszentren an. Streuungsmaße sind Maßzahlen die angeben, wie dicht die Beobachtungen um den Lageparameter liegen. Zunächst wird das arithmetische Mittel und die Quantilmethode definiert. Anschließend werden die empirische Varianz, die Standardabweichung und der Inter-Quartilsabstand erläutert. Zuletzt wird die Schiefe, eine Maßzahl für die Gestalt einer Verteilung, eingeführt.

Arithmetisches Mittel

Sei $x_1, x_2, ..., x_n$ eine Stichprobe vom Umfang n. Das arithmetische Mittel ist definiert durch

$$\overline{x} = \overline{x}_n = \frac{1}{n} \sum_{i=1}^{n} x_i. \tag{2.2}$$

Das arithmetische Mittel $\overline{x}$ ist sensitiv gegenüber Ausreißern und wird folglich durch kleine und große Werte stark beeinflusst [9]. In solchen Fällen ist der Median dem arithmetischen Mittel vorzuziehen.

Quantilmethoden

Sei $x_1, x_2, ..., x_n$ eine Stichprobe vom Umfang n und $0 < p < 1$. Weiterhin sei $x_{(1)} \leq ... \leq x_{(n)}$ die zugehörige geordnete Stichprobe. Dann ist das p-Quantil definiert durch

$$\tilde{x}_p = \begin{cases} x_{(np)} & \text{falls } np \notin \mathbb{N} \\ \frac{1}{2}(x_{(np)} + x_{(np+1)}) & \text{falls } np \in \mathbb{N} \end{cases} \tag{2.3}$$

Dabei gibt np an, dass mindestens np% Merkmalswerte kleiner oder gleich $x_{(np)}$ und mindestens $1 - np$% der Merkmalswerte größer oder gleich $x_{(np)}$ sind.

Übliche Bereiche sind das erste Quantil $\tilde{x}_{0.25}$, der Median $\tilde{x}_{0.5}$ und das dritte Quantil $\tilde{x}_{0.75}$. Insbesondere der Median $\tilde{x}_{0.5}$ ist von größerem Interesse, da dieser die Merkmalswerte genau in dem Verteilungszentrum trennt [9].

Inter-Quartilsabstand

Sei $x_1, x_2, ..., x_n$ eine Stichprobe vom Umfang n. Weiterhin sei $x_{(1)} \leq ... \leq x_{(n)}$ die zugehörig geordnete Stichprobe. Für den Quartilsabstand gilt

$$IQR = \tilde{x}_{0.75} - \tilde{x}_{0.25}. \tag{2.4}$$

Der IQR beschreibt den mittleren 50% Bereich der Daten[10].

Varianz und Standardabweichung

Sei eine Stichprobe $x_1, ..., x_n$ gegeben, mit arithmetischen Mittel $\overline{x}$, dann steht

$$s^2 = s_n^2 = \frac{1}{n-1} \sum_{k=1}^{n} (x_k - \overline{x})^2 \tag{2.5}$$

für empirische Varianz. Die entsprechende Wurzel

$$s = s_n = \sqrt{\frac{1}{n-1} \sum_{k=1}^{n} (x_k - \overline{x})^2}, \tag{2.6}$$

entspricht der Standardabweichung der Stichprobe.

Dabei weist die Standardabweichung s_n eine durchschnittliche Abwecheiung der Merkmalswerte von dem arithmetischen Mittel $\overline{x}$ aus. Die Varianz ist invariant gegenüber einer Verschiebung des Nullpunktes [10].

Schiefe

Die Schiefe ist eine Maßzahl zur Beschreibung der Gestalt einer Verteilung.

Sei eine Stichprobe $x_1, x_2, ..., x_n$ gegeben. Modus D entspricht der Merkmalsausprägung mit maximaler Häufigkeit. Für die Schiefe gilt nach der Fechnerschen Lageregel [10]:

$$\begin{aligned} &D = \tilde{x} = \bar{x} \Rightarrow \text{symmetrische Verteilung} \\ &D < \tilde{x} < \bar{x} \Rightarrow \text{(linkssteil) rechtsschiefe Verteilung} \\ &D > \tilde{x} > \bar{x} \Rightarrow \text{(rechtssteil) linksschiefe Verteilung.} \end{aligned} \tag{2.7}$$

Als Maß für die Schiefe wird das Schiefemaß von Pearson verwendet [10]

$$g_p = \frac{\bar{x} - D}{s}. \tag{2.8}$$

2.2.2 *Statistische Hypothesentests*

In diesem Abschnitt wird der Shapiro-Wilk, Levene- oder Brown-Forsythe- und der Mann-Whitney-U Test definiert.

Bei statistischen Hypothesentests geht es um das Aufstellen und Prüfen einer Annahme über die Gültigkeit eines bestimmten Sachverhalt. Dies könnte beispielsweise der Vergleich zweier Verteilungen auf gewisse Lage- oder Streuungsmaße sein. Die von Neyman und Pearson entwickelte Methodik des Hypothesentests nimmt die Existenz einer Nullhypothese H_0 und einer Gegenhypothese H_1 an [11]. Anhand einer oder multipler Stichproben wird mithilfe einer maximalen Irrtumswahrscheinlichkeit α (Signifikanzniveau), die Nullhypothese H_0 auf einen vorher definierten Sachverhalt getestet. Dabei schließen sich die gegenseitigen Aussagen von H_0 und H_1 aus, sodass in der Grundgesamtheit nur die Aussage einer Hypothese gültig sein kann [12].

Mögliche Tests sind beispielsweise ein Test auf Verteilungsanpassung, Tests für Lagealternativen oder Variabilitätstests. In dieser Arbeit werden die Hypothesentests von Shapiro-Wilk, der Mann-Whintey-U und Levene oder Brown-Forsythe Test verwendet.

Shapiro-Wilk Test

Der Shapiro-Wilk Test berechnet eine sogenannte W Statistik und prüft, ob eine zufällige Stichprobe aus einer Normalverteilung stammt [13].

Sei $x_1, \leq ... \leq x_n$, $x \in \mathbb{R}$ eine geordnete Stichprobe mit Verteilungsfunktion F_x und $F \sim N(\mu, \sigma)$ eine Normalverteilung. Für W gilt

$$W^2 = \frac{(\sum_{i=1}^{n} a_i x_{(i)})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \tag{2.9}$$

wobei a_i einem Gewicht entspricht, siehe Koeffiziententabelle [13].

Die zu prüfenden Hypothesen werden wie folgt definiert:

$$H_0 : F(x) = F_x(x) \text{ vs } H_1 : F(x) \neq F_x(x) \tag{2.10}$$

Sofern $W^2 \leq W_\alpha^2$ wird die Nullhypothese H_0 verworfen und es kann keine Normalverteilung angenommen werden.

Levene oder Brown-Forsythe Test

Der Levene oder Brown-Forsythe Test wird genutzt um zu testen, ob k Stichproben die gleiche Varianz s^2 haben [14].

Sei X eine Stichprobenvariable mit Stichprobengröße N, unterteilt in k Gruppen mit n_i Stichproben. Weiterhin seien $s_1^2, s_2^2, ..., s_k^2$ die zugehörigen Stichproben-Varianzen. Für die Teststatistik W gilt dann

$$W = \frac{(N-k)\sum_{i=1}^{k} n_i(\bar{Z}_i - \bar{Z})^2}{(k-1)\sum_{i=1}^{k}\sum_{j=1}^{n_i}(Z_{ij} - \bar{Z}_i)^2}, \tag{2.11}$$

wobei $\bar{Z}$ der Mittelwert über alle Z_{ij} und $\bar{Z}_i$ dem Mittelwert der i_{ten} Gruppe entspricht.

Für diese Arbeit ist insbesondere die Definition von $Z_{ij} = |X_{ij} - \tilde{X}_i|$ relevant. Durch Verwendung des Median $\tilde{X}_i$ wird der Levene Test robuster gegenüber nicht Normalverteilten Stichproben [15].

Die zu prüfende Hypothese wird wie folgt definiert:

$$\begin{aligned} &H_0 : s_1^2 = s_2^2 = ... = s_k^2 \\ &H_1 : s_i^2 \neq s_j^2 \text{ für mindestens ein Paar}(i, j). \end{aligned} \tag{2.12}$$

Sofern $W > F_{\alpha,k-1,N-k}$ wird die Null-Hypothese H_0 verworfen und es kann keine Gleichheit der Varianzen (Homoskedastizität) angenommen werden.

Mann-Whitney-U Tests

Der Mann-Whitney-U Test wird zur Überprüfung von Hypothesen über die Lage oder Form zweier statistischer Verteilungen verwendet. Es handelt sich um einen nicht-parametrischen Test, es muss also keine Normalverteilung angenommen werden. Der Mann-Whintey-U Test setzt voraus, dass die Beobachtungen $x_1, ..., x_m$, $y_1, ..., y_n$ mindestens ordnial skaliert sind. Weiterhin müssen die Variablen $X_1, ..., X_m$ und $Y_1, ..., Y_n$ unabhängig sein und eine stetige Verteilungsfunktion F bzw. G haben. Durch Hinzunahme zweier schwachen Bedingungen der Homoskedastizität und gleichen Verteilung von F und G, wird eine Verschiebung der Verteilung getestet [16], [17].

Seien $X_1, ..., X_m$, $Y_1, ..., Y_n$ unabhängige Zufallsvariablen, X_i, $i = 1, ...n$ stetig mit Verteilungsfunktion F und Y_i $i = 1, ...n$ stetig mit Verteilungsfunktion G. Für die Teststatistik des Mann-Whitney-U-Tests gilt

$$U_{F>G} = mn + \frac{n(n+1)}{2} - \sum_{i=1}^{n} R(Y_i), \quad U_{F<G} = mn + \frac{m(m+1)}{2} - \sum_{i=1}^{m} R(X_i),$$

wobei $R(X_i)$, $R(Y_i)$ den Rängen der gemeinsamen Beobachtungen $R(X_1)$, $R(X_2), ..., R(X_n), R(Y_1), R(Y_2), ..., R(Y_m)$ entsprechen.

$R(X_i)$ entspricht der Anzahl aller Werte der gemeinsamen Stichprobe, die kleiner oder gleich x_i sind ($R(Y_i)$ analog)[18].

Die zu prüfende Hypothese für den zweiseitigen Test ist definiert als

$$H_0 : F(z) = G(z) \text{ vs } H_1 : F(z) = G(z + \theta), \forall z \in \mathbb{R}, \theta \neq 0. \tag{2.13}$$

Für die einseitige Hypothese $F < G$ (X stochastisch größer als Y) gilt

$$H_0 : F(z) = G(z) \text{ vs } H_1 : F(z) = G(z + \theta), \forall z \in \mathbb{R}, \theta < 0. \tag{2.14}$$

und für $F > G$ (X stochastisch kleiner als Y) gilt

$$H_0 : F(z) = G(z) \text{ vs } H_1 : F(z) = G(z + \theta), \forall z \in \mathbb{R}, \theta > 0. \tag{2.15}$$

Die Null-Hypothese H_0 wird verworfen falls: $min(U_{F>G}, U_{F<G}) \leq U_{\alpha/2}$ bei einem zweiseitigen Test, $U_{F<G} \leq U_\alpha$ nach Formel 2.14 und $U_{F>G} \leq U_\alpha$ nach Formel 2.15.

2.2.3 *Distanzmaße*

In diesem Abschnitt werden ausgewählte Distanzmaße vorgestellt, die im Laufe dieser Arbeit angewendet werden. Das Ziel der hier verwendeten Distanzmaße besteht darin, die Editier-Ähnlichkeit von zwei Datenpunkten zu messen. Insbesondere wird ein Fokus auf die Ähnlichkeit von zwei *String* Werten, also zwei Wörtern gelegt.

Hamming-Distanz

Die Hamming-Distanz ist ein Maß für den binären Unterschied an der selben Stellen zweier Wörter [19], [20]. Die Distanz wird definiert anhand der Anzahl an Stellen an denen sich zwei Vektoren unterscheiden.

Sei $x = (x_1, x_2, ..., x_n)$ und $y = (y_1, y_2, ..., y_n)$ zwei Wörter gleicher Länge gegeben. Die Hamming-Distanz $d_h(x, y)$ ist definiert als die Anzahl an Stellen, bei denen sich x und y unterscheiden. Es gilt

$$d_h(x, y) = \sum_{i=1}^{n} \mathbb{1}_{x_i = y_i}, \tag{2.16}$$

wobei $\mathbb{1}_{x_i = y_i} = 1$, wenn $x_i = y_i$ sonst 0.

In dieser Arbeit wird die relative Hamming-Distanz $d_h^{rel}(x, y)$ herangezogen für die gilt

$$d_h^{rel}(x, y) = \frac{1}{n} d_h(x, y), \tag{2.17}$$

wobei n der Länge eines Wortes entspricht.

Damerau-Levenshtein-Distanz

Die Levenshtein-Distanz beschreibt die kleinste Anzahl von nötigen Änderungsoperationen zur Transformation eines Strings in einen anderen. Änderungenoperationen sind Löschen, Einsetzen oder Ersetzen. Die Damerau-Levenshtein-Distanz ist eine erweiterte Levenshtein-Distanz und berücksichtigt zusätzlich die Transposition als weitere Änderungsoperation [21].

Seien Strings x und y gegeben. Für die Damerau-Levenshtein-Distanz $d_{x,y}(|x|, |y|)$ gilt

$$d_{x,y}(i, j) = min \begin{cases} 0 & \text{wenn } i = j = 0 \\ d_{x,y}(i-1, j) + \mathbb{1} & \text{wenn } i > 0 \\ d_{x,y}(i, j-1) + \mathbb{1} & \text{wenn } j > 0 \\ d_{x,y}(i-1, j-1) + \mathbb{1}_{x_i \neq y_j} & \text{wenn } i, j > 0 \\ d_{x,y}(i-2, j-2) + \mathbb{1} & \text{wenn } i, j > 1 \text{ und } x_i = y_{j-1}, x_{i-1} = y_j, \end{cases} \tag{2.18}$$

und Indikatorfunktion $\mathbb{1}_{(x_i \neq y_j)} = 0$ wenn $x_i = y_j$.

Dabei steht jeder Fall für eine Operation, die an den Strings x, y durchgeführt wurden. Es gilt: Buchstaben sind identisch (wenn $i = j = 0$), Löschoperation (wenn $i > 0$), Einfügeoperation (wenn $j > 0$), Substitution (wenn $i, j > 0$) oder eine Transposition (wenn $i, j > 1$).

In dieser Arbeit wird die relative Damerau-Levenshtein-Distanz verwendet mit

$$d_{x,y}^{rel} = \frac{d_{x,y}(|x|, |y|)}{max(|x|, |y|)} \tag{2.19}$$

2.2.4 *Mustererkennung*

Mustererkennung beschreibt das Feld der automatischen Erkennung von wiederkehrenden Ereignissen oder Regelmäßigkeiten in Daten. Basierend auf gewonnenen Zusammenhängen dieser Regelmäßigkeiten, können Daten eingeordnet, klassifiziert und interpretiert werden [22].

2.2.4.1 *Faktoranalyse*

Die Faktoranalyse fungiert als Methode zur Dimensionalitätsreduktion. Dabei wird versucht die Anzahl an latenten Variablen und die zugrunde liegenden Faktorstrukturen aus einer Reihe miteinander korrelierten Variablen zu identifizieren. Diese latenten Variablen sind nicht direkt messbar und werden mit einem Verfahren wie der Explorativen-Faktoranalyse geschätzt. Sie drücken das Verhältnis verschiedener Variablen zueinander aus [23].

Sei $x_1, x_2, ..., x_p, x \in X$ ein messbarer Vektor mit Mittelwert μ und Korrelationsmatrix R. Die Faktoranalyse nimmt an, dass X eine lineare Abhängigkeit von gemeinsamen Faktoren $F_1, ..., F_q$ besitzt sowie Variationen $\epsilon_1, ..., \epsilon_p$. Es gilt

$$\begin{aligned} X_1 - \mu_1 &= l_{11}F_1 + ... + l_{1q}F_q + \epsilon_1 \\ &\vdots \\ X_p - \mu_p &= l_{p1}F_p + ... + l_{pq}F_q + \epsilon_p, \end{aligned}$$

oder in Matrixform

$$X - \mu = LF + \epsilon, \tag{2.20}$$

l_{ij} wird als Ladung der i_{ten} Variable auf den j_{ten} Faktor bezeichnet. Unter der Annahme das der Erwartungswert $E(F) = 0$ und die Kovarianz $Cov(F) = E(FF') = I$ der Einheitsmatrix entspricht. F und ϵ unabhängig sind, $Cov(\epsilon, F) = E(\epsilon F) = 0$, also $Cov(\epsilon) = R$ einer Diagonalmatrix entspricht, folgt für die Kovarianz von X:

$$\begin{aligned} Cov(X, F) &= L \\ Cov(X_i, F_j) &= l_{ij} \end{aligned} \tag{2.21}$$

Der Anteil der Varianz der i_{ten} Variable wird dabei als i_{te} Kommunalität bezeichnet und trägt anteilig zu den Faktoren bei. Dieser Anteil der Varianz $Var(X_i) = \sigma_{ii}$, wird als spezifische Varianz bezeichnet und es gilt:

$$Var(X_i) = \sigma_{ii} = h_i^2 + \psi_i, \quad i = 1, 2, ..., p \tag{2.22}$$

wobei $h_i^2 = l_{i1}^2 + ... + l_{iq}^2$ und $\psi_i = Cov(\epsilon_i)$ entspricht.

Die Methode nimmt an, dass die Varianzen und Kovarianzen von X durch die pq Faktorladungen l_{ij} und der spezifischen Varianz ψ_i reproduziert werden können.

Explorative-Faktoranalyse

Im Folgenden wird ein Schätzmodell für die Kommunalitäten, die Explorative-Faktoranalyse (EFA), vorgestellt [23], [24].

Angenommen es sei eine initiale Schätzung ψ_i^* der spezifischen Varianz verfügbar. Für $h_i^{*2} = 1 - \psi_i^*$ folgt für die Korrelationsmatrix R

$$R_r = \begin{bmatrix} h_i^{*2} & r_{12} & \dots \\ \vdots & \ddots & \\ r_{1p} & & h_p^{*2} \end{bmatrix}, \tag{2.23}$$

insbesondere gilt $R_r = L_r^* L_r^{*t}$. Es werden die Schätzungen für L_r^* wie folgt angewandt

$$\begin{aligned} L_r^* &= \begin{bmatrix} \sqrt{\hat{\lambda}_1^*}\hat{\epsilon}_1^* & \dots & \sqrt{\hat{\lambda}_q^*}\hat{\epsilon}_q^* \end{bmatrix} \\ \psi^* &= 1 - \sum_{j=1}^{q} l_{ij}^{*2} \end{aligned} \tag{2.24}$$

mit den größten Eigenwert-Eigenvektor Paare aus R_r, $(\hat{\lambda}_1^*, \hat{\epsilon}_1^*)$, $i = 1, 2, ..., q$. Für die Kommunalitäten folgt iterativ:

$$\hat{h}_i^{*2} = \sum_{j=1}^{q} l_{ij}^{*2} \tag{2.25}$$

Nach jeder Iteration werden die berechneten Kommunalitäten als neuer Startpunkt definiert, bis die restlichen Parametermatritzen L, F und $Cov(\epsilon)$ den Testdaten angepasst wurden.

2.2.4.2 *K-Means Clustering*

Der K-Means Clustering Algorithmus ist ein partitionierender Algorithmus. Das Ziel besteht in der Identifikation von K Gruppen innerhalb eines multidimensionalen Merkmalsraumes [25]. Bei einem Algorithmus basierend auf Partitionierung, wird anhand eines zentralen Objekts (Zentroid), die Partitionierung der Daten iterativ vorgenommen. Der K-Means Algorithmus wird in dieser Arbeit verwendet, da keine vorherige Kategorisierung der Daten vorhanden ist. Im Folgenden wird eine formale Definition des Algorithmus gegeben [22], [26].

Sei $x_1, x_2..., x_n$ ein Datensatz mit N Beobachtungen eines D-dimensionalen Vektor x, wobei $x_i \in \mathbb{R}$. Das Ziel besteht darin, k Zentroide and die zugehörigen Gruppen (Cluster) c_i zu bestimmen. Dazu werden zunächst zufällige Cluster-Zentroide $\mu_1, \mu_2, ..., \mu_k \in \mathbb{R}$ initialisiert. Anschließend werden die folgenden Schritte wiederholt, bis eine Konvergenz hergestellt ist und die Zentroide ihre Position nicht ändern.

Für jedes i gilt

$$c_i = \min_j dist(x_i, \mu_j) \tag{2.26}$$

und für jedes j gilt

$$\mu_j = \frac{\sum_{i=1}^{n} \mathbb{1}_{c_i=j} x_i}{\sum_{i=1}^{n} \mathbb{1}_{c_i=j}} \tag{2.27}$$

mit $\mathbb{1}$ einer Indikatorfunktion, welche bei minimaler $dist(x_i, \mu_k)$ 1 einnimmt. Als Distanzmaß kann die beispielsweise die Euklidische-Distanz $\|x - y\|$ verwendet werden, dann gilt

$$c_i = \min_j \|x_i - \mu_j\|^2. \tag{2.28}$$

Der Algorithmus teilt folglich die Daten in K Gruppen auf, in denen die Abstände der Gruppen zueinander maximiert und die Summe der quadratischen Abstände der Daten innerhalb einer Gruppe minimiert werden.

Zuletzt wird ein Maß für die Güte der erzeugten Cluster eingeführt, welches kein Apriori-Wissen über die Gruppenzugehörigkeit der Daten voraussetzt.

Silhouette-Breite-Kriterium

Das Silhouette-Breite-Kriterium (SSWC) ist eine Maßzahl für die Kompaktheit der Daten innerhalb eines Clusters und der Separation zwischen den Clustern [27]. Je höher der SSWC Index, desto eindeutiger ist die berechnete Aufteilung eines Partitionierungsalgorithmus [28].

Sei $M = \{\mu_1^*, ..., \mu_k^*\}$ die generierten Clusterzugehörigkeiten eines Clusteringverfahrens, wobei $x_i \in \mu_r^*$ gilt. Weiterhin sei $c_1^*, ..., c_k^*$ die Zentroide der K Cluster. Für den SSWC Index gilt

$$\frac{1}{m} \sum i = 1^m \frac{\beta_{ir} - \alpha_{ir}}{max(\alpha_{ir}, \beta_{ir}}, 1 < k < m, \tag{2.29}$$

mit $\alpha_{ir} = dist(x_i, c_r^*)$, $\beta_{ir} = \min_{q \neq r} dist(x_i, c_q^*)$.

Der SSWC-Index gibt folglich für eine Beobachtung x_i an, wie gut die Zuordnung dieser Beobachtung zu den beiden nächstgelegenen Clustern r, q ist.

3 KONZEPTION

Dieses Kapitel ist in drei Teile untergliedert.

Im ersten Teil Abschnitt 3.1, wird eine Übersicht über die vorhandene Systemlandschaft gegeben. Anschließend wird in Abschnitt 3.1.2 eine Nutzwertanalyse, zur Bestimmung einer geeigneten Messaging Lösung, durchgeführt.

Im zweiten Teil, Abschnitt 3.2, werden die zu verarbeitenden Daten ausgewertet. Zunächst wird in Abschnitt 3.2.1 die Herkunft und Menge der Daten definiert. Anschließend werden die Merkmale der Daten in Abschnitt 3.2.2 untersucht und selektiert. In Abschnitt 3.2.3 werden die selektierten Daten für die weitere Analyse transformiert. Abschließend werden in Abschnitt 3.2.4 Hypothesen formuliert die mithilfe einer Faktoranalyse in 3.2.4.2 und einem partitionierenden Clustering-Verfahren in 3.9 untersucht und bewertet werden.

Im letzten Teil, Abschnitt 3.3, wird basierend auf den gewonnenen Erkenntnisse ein Konzept für eine eventgetriebene und nachrichtenbasierte Integrationsarchitektur vorgestellt.

3.1 SYSTEMLANDSCHAFT UND LÖSUNGSSELEKTION

Basierend auf der Aufgabenstellung: Konzeptionierung einer in die aktuelle Systemlandschaft integrierbaren, eventgetrieben und nachrichtenbasierten Integrationslösung, bildet die aktuelle Systemlandschaft das Fundament für die folgenden Kapitel.

Zunächst werden für diese Arbeit relevante Teile der Gesamtarchitektur vorgestellt. Anschließend wird der *Business Data Hub (BDH)* und die darin vorhandenen Applikationen und Services erläutert. Abschließend folgt eine Zielarchitektur, für die es eine Middleware in Form einer Messaging-Lösung (*Enterprise Messaging (EM)*-Lösung) nach Abschnitt 2.1.3 mit möglichst hoher Kompatibilität zu identifizieren gilt.

3.1.1 *Beschreibung der Systemlandschaft*

Die Systemlandschaft von *MU* kann in vier Domänen unterteilt werden: Nutzerapplikationen (User), *Internet of Things (IoT)*-Geräte (Things), lokalen Server (On-Premise) und Cloud Plattformen (Cloud). Die Domänen werden auch als Integrationsdomänen bezeichnet und beschreiben ein typisches Integrationsgebiet in einer hybriden Systemlandschaft. Eine hybride Systemlandschaft liegt vor, wenn sowohl lokale als auch Cloud basierte Bereiche miteinander interagieren.

Die Integration zwischen den Domänen kann für alle vier Bereiche erforderlich sein, um den Daten- und Prozesstransfer zwischen On-Premise Systemen, Things, User und Cloud Plattformen zu ermöglichen. *MU* nutzt diverse Cloud-Services von unterschiedlichen Anbietern wie SAP, Micrsoft und Salesforce. Für diese Arbeit sind insbesondere ausgewählte SAP Services wie der *BDH* relevant und werden im Folgenden näher erläutert.

Business Data Hub

Der *BDH* beschreibt eine Gruppe ausgewählter Applikationen, welche in der SAP *Business Technology Platform (BTP)* zur Verfügung stehen [29].

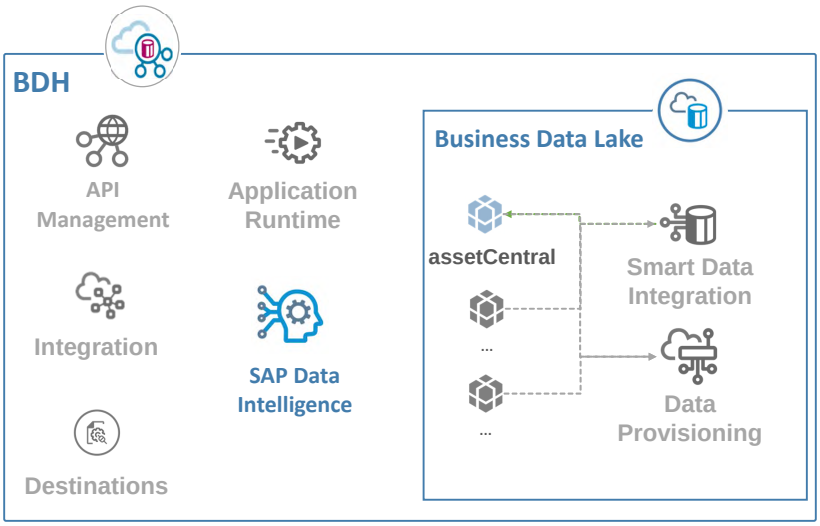


Abbildung 3.1: Darstellung von Software-Bestandteilen sowie ausgewählter Datenbank Schemata des Business Data Hub.

In Abb. 3.1 sind die Applikationen des *BDH* dargestellt. Relevante Applikationen sind blau hervorgehoben.

Die ***Data Intelligence (DI)*** ist eine Managementlösung für Datenmengen- und ströme, die diverse Applikationen wie bspw. den Pipeline Modeler bereitstellt. Mithilfe des Pipeline Modeler können durch containerisierte Laufzeitumgebungen diverse Programmiersprachen zur Verbindung und Verarbeitung von Datenmengen- und strömen genutzt werden. Die Lösung verwendet Kubernetes, um die Skalierbarkeit des Systems zu garantieren [30].

Das ***Business Data Lake (BDL)*** beinhaltet eine SAP HANA Cloud Datenbank und etwaige Integrations- und Provisionierungsapplikationen. Die SAP HANA ist eine In-Memory-Datenbank und nutzt den Arbeitsspeicher als Speichermedium. Dies ermöglicht signifikant schnellere Lese- und Schreibzeiten im Vergleich zu konventionellen Datenbankmanagementsystemen [31].

Grundgerüst der Zielarchitektur

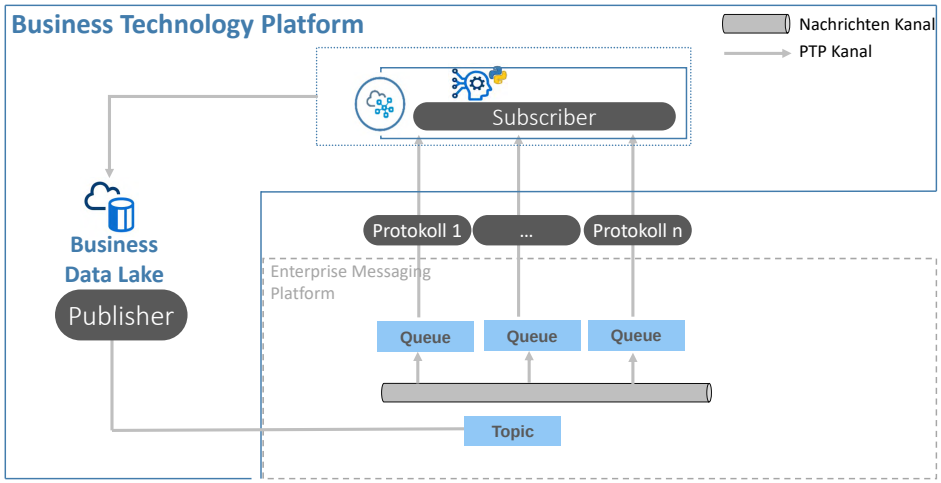


Abbildung 3.2: Grundgerüst einer Lösungsarchitektur mit einer variablen Darstellung eines Enterprise Messaging Systems.

In Abb. 3.2 ist das Grundgerüst der zu erarbeitenden Integrationsarchitektur dargestellt. In dem dargestellten Szenario werden Events in Form von Datenbestandänderungen im *BDL* detektiert und publiziert. Die erzeugten Event werden von einer *EM*-Lösung empfangen und verteilt. Die *DI* wird als Subscriber eingesetzt und empfängt, transformiert und kategorisiert die Inhalte der Nachricht. Nach erfolgreiche Einordnung der Inhalte, wird das Ergebnis und persistiert. Anhand dieser Ergebnisse können anschließend zur Erhöhung der Datenqualität, Operationen wie Update oder Delete in dem *assetCentral*-Schema durchgeführt werden.

Wie aus Abb. 3.2 hervorgeht, ist die *EM*-Lösung dieser Architektur nicht eindeutig definiert. Aufgrund dessen werden in Abschnitt 3.1.2 ausgewählte Messaging-Lösungen mithilfe einer Nutzwertanalyse verglichen und eine geeignete Lösung selektiert.

3.1.2 *Auswahl einer geeigneten Lösung*

3.1.2.1 *Einführung in die Nutzwertanalyse*

Eine Nutzwertanalyse ermöglicht die systematische Bewertung von Alternativen Lösungen für eine bestimmte Zielsetzung. Es werden Alternativen anhand harter und weicher Kriterien miteinander verglichen. Ziel ist es, jene Alternative auszuwählen, welche der Zielsetzung am besten dient [32]. Es wird die Alternative mit dem größten Nutzwert gewählt.

Zur Durchführung der Nutzwertanalyse wird die erweiterte Form von Kepner verwendet [33].

1. Entscheidungserklärung

Im Folgenden soll die bestmögliche *EM*-Lösung zur Integration in die Zielarchitektur aus Abb. 3.2 ausgewählt werden. Hierbei werden die in Frage kommenden Lösungen anhand von qualitativen Entscheidungskriterien wie die Zukunftsperspektive, Integrier- und Nutzbarkeit, sowie die Funktionalität miteinander verglichen. Mögliche *EM*-Lösungen sind Apache Kafka, Amazon Simple Notification Service, Microsoft Event Hub, Google Pub/Sub, IBM MQ und SAP Event Mesh. Aufgrund der von *MU* bereits eingesetzten Software-Lösungen, wird die Nutzwertanalyse auf einen Vergleich zwischen den Lösungen Apache Kafka, Microsoft Event Hub und SAP Event Mesh begrenzt.

2. Entscheidungskriterien

Das Lösungssystem wird anhand folgender Kriterien bewertet:

MARKTPOSITIONIERUNG : Die Lösung wird von einem Unternehmen bereitgestellt oder genutzt, welches aus Sicht des Marktes bestand hat.

REFERENZWERTUNG GARTNER: Die Lösung wird von einem Unternehmen bereitgestellt oder genutzt, dass laut Gartner zukunftsfähige Software-Lösungen bereitstellt.

SCHNITTSTELLENEXISTENZ: Die Lösung verfügt über eine *API* Schnittstelle, die mit üblichen POST, GET Anfragen umgehen kann.

CLOUD BASIERT: Die Lösung kann cloudbasiert eingesetzt werden. Es sind keine lokalen Rechenkapazitäten erforderlich.

UPDATE ZYKLEN: Die Lösung wird mindestens einmal im Jahr upgedatet, um aktuelle Sicherheitsstandards zu erfüllen.

AUTHENTIFIZIERUNGSMÖGLICHKEITEN: Die Lösung verfügt sowohl für den Publisher als auch den Subscriber über mindestens zwei Authentifizierungsmöglichkeiten.

BENUTZERFREUNDLICHKEIT: Die Lösung erfordert für eine Erstimplementierung einen möglichst geringen Zeitaufwand. Dieser Zeitraum darf 8 Zeitstunden nicht überschreiten.

NACHRICHTENPARAMETER: Die Lösung muss pro Nachricht eine Mindestgröße von 250 KiloByte zulassen.

KOMMUNIKATIONSPROTOKOLLE: Die Lösung ermöglicht die Nutzung diverser Protokolle oder Methoden zur Kommunikation zwischen dem Lösungssystem und den Publish- und Subscribern.

QUALITÄTSOPTIONEN : Die Lösung muss über Qualitätsoptionen verfügen, mithilfe derer mindestens eine Bestätigung des Nachrichtenempfangs möglich ist.

3. Unterteilung der Kriterien

In diesem Schritt erfolgt eine Unterteilung der Kriterien in zwei Kategorien: harte- (*MUST*) und weiche (*WANT*)-Kriterien. *MUST*-Kriterien müssen von dem Zielsystem erfüllt werden. *WANT*-Kriterien sind optional.

	Kriterium	MUST	WANT
A	Marktpositionierung		x
B	Referenzwertung Gartner		x
C	Schnittstellenexistenz	x	
D	Cloudbasiert	x	
E	Update Zyklen		x
F	Authentifizierungsmöglichkeit		x
G	Benutzerfreundlichkeit		x
H	Nachrichtenparameter	x	
I	Kommunikationsprotokolle	x	
J	Qualitätsoptionen		x

Tabelle 3.1: Unterteilung der Entscheidungskriterien nach MUST und WANT.

In Tbl. 3.1 ist die Zuordnung der Kriterien dargestellt. Es wurden vier *MUST* und sechs *WANT* Kriterien definiert. Die Wahl der *MUST*-Kriterien beruht auf den Anforderungen der Systemlandschaft und wird im Folgenden erläutert.

Aufgrund des Quellsystems *BDL* muss eine Schnittstellenexistenz (**C**) gegeben sein, die eine Kommunikation unabhängig von plattforminternen Kommunikationskanälen bereitstellt. Es muss eine *API* Schnittstelle verfügbar sein, um externen Applikationen die Kommunikation zu ermöglichen.

Die Lösung muss cloudbasiert (**D**) oder in einer Cloud-Plattform bereitgestellt werden, da eine Vereinheitlichung von Teilen der Infrastruktur in der Cloud von *MU* angestrebt wird.

Die Nachrichtenparameter (**H**) entsprechen der Mindestanforderung für etwaige Integrationsabläufe.

Zur Gewährleistung der Einsatzfähigkeit des Systems in diversen Bereichen des Unternehmens, sind multiple Protokolle oder Methoden zur Kommunikation (**I**) nötig.

4. Gewichtung der WANT Kriterien

Zur Ermittlung der Gewichtung werden nun paarweise Vergleiche der Kriterien durchgeführt. Dazu wird zunächst eine Bewertungsskala für die unterschiedlichen Varianten definiert.

Sei $x, y \in M$ wobei $M = \{A, B, E, F, G, J\}$ der Menge an *WANT*-Kriterien entspricht. Es gilt:

$$\text{x wichtiger als y?} \begin{cases} 0 & \text{falls } \textit{nein} \\ 1 & \text{falls } \textit{gleichwertig} \\ 2 & \text{falls } \textit{ja} \end{cases} \tag{3.1}$$

Der in Formel 3.1 angeführte Vergleich wird für alle möglichen Kombinationen der *WANT*-Kriterien durchgeführt. Die Ergebnisse der paarweisen Vergleiche sind in Tbl. 3.2 zusammengeführt.

	A	B	E	F	G	J	$\sum_r$	Faktor
A		2	0	0	2	0	4	14.8%
B	0		0	0	1	0	1	3.7%
E	2	2		1	1	1	6	22.2%
F	2	0	1		2	1	6	22.2%
G	0	1	1	0		0	2	7.4%
J	2	2	1	1	2		8	29.6%
						$\sum_c$	27	100%

Tabelle 3.2: Gewichtsverteilung der *WANT*-Kriterien, basierend auf paarweiser Vergleiche der Kriterien.

Zur Darstellung der Punktzahl eines Kriteriums wurden in Tbl. 3.2 zwei Notationen eingeführt. $\sum_r$ repräsentiert die Summe der paarweisen Vergleiche eines Kriteriums mit allen anderen Kriterien in der Menge M. $\sum_c$ entspricht der Summe über alle $\sum_r$. Mithilfe der Kenngrößen wird der Gewichtungsfaktor $Faktor_{x \in M} = \frac{\sum_r(x)}{\sum_c}$ gebildet. Dieser Faktor zeigt den prozentualen Anteil der Gewichtung eines Kriteriums.

Zur Differenzierung unterschiedlicher Ausprägungen der Kriterien wird eine Zielerfüllungsskala eingeführt. Es werden $0-5$ Punkte vergeben, welche den Erfüllungsgrad des Kriteriums widerspiegeln. Zwischen $0-1$ ist ein Kriterium schlecht, $2-3$ mittel und $4-5$ gut erfüllt.

	Skala		
	0-1	2-3	4-5
Kriterien	schlecht	mittel	gut
A	Alter <10 Jahre	10 <Alter <20 Jahre	Alter >20 Jahre
	Mk[1]: nano-micro*	Mk[1]: small-mid**	Mk[1]: big-mega***
B	Niche Player	Visionaire, Challenger	Leader
E	Jährlich	Quartalsweise	Monatlich
F	min. 1, max 1	min. 1	min. 2
G	minimale Doku	mittelmäßige Doku	ausführliche Doku
	Training: <8 h	Training: <8 h	Training: <8 h
	viel Pk[2]	wenig Pk[2]	wenig bis keine Pk[2]
J	at least once Delivery	at least once Delivery	at least once Delivery
		at most once Delivery	at most once Delivery
			exactly once Delivery

[1] Marktkapitalisierung, [2] Programmierkenntnisse
[*] 250 Millionen bis 2 Milliarden, [**] 2 bis 10 Milliarden, [***] über 10 bzw. über 200 Milliarden

Tabelle 3.3: Geltungsbereichsdefinition der *WANT*-Kriterien basierend auf der Zielerfüllungsskala.

Basierend auf der Zielerfüllungsskala, wird in Tbl. 3.3 eine Definition für die Geltungsbereiche der *HAVE*-Kriterien gegeben. Dargestellt werden die Kriterien *A*, *B*, *E*, *F*, *G* und *J* untergliedert nach den Kategorien der Bewertungsskala, sowie die Definitionen der einzelnen Parameter.

5. Lösungsalternativen

Die getroffenen Einschränkung der Lösungsalternativen aus der Entscheidungserklärung haben bestand. Es werden die Lösungen Apache Kafka, Microsoft Event Hub und SAP Event Mesh verglichen.

Die nachfolgende Prüfung der Kriterien wird basierend auf den zugehörigen Dokumentationen der Lösungsalternativen Event Mesh [34], Kafka [35] und Event Hub [36] durchgeführt. Zusätzliche Quellen werden gesondert ausgewiesen.

6. Prüfen der MUST-Kriterien

In diesem Abschnitt werden die Lösungsalternativen auf die *MUST*-Kriterien geprüft. Sofern ein System einem Kriterium nicht genügt, wird dieses System in der weiteren Analyse nicht berücksichtigt.

				Must-Kriterien erfüllt		
Kriterien	SAP	Kafka	Microsoft	SAP	Kafka	Microsoft
C	*API* für *P/S* Java, Nodejs	*API* für *P/S* Java	*API* für *P/S* Java	ja	ja	ja
D	Vst. gemanaged PaaS	Integrierbar: On-Prem in Cloud Plattform	Vst. gemanaged PaaS	ja	ja	ja
H	1 MegaByte	1 MegaByte	1 MegaByte	ja	ja	ja
I	HTTP, AMQP, AMQP	Kafka, Konnektor erforderlich	HTTP, AMQP, Kafka	ja	ja	ja

Tabelle 3.4: Prüfung der Lösungsalternativen auf die *MUST*-Kriterien.

In Tbl. 3.4 ist eine Übersicht der zugrundeliegenden Parameter jeder Lösungsvariante dargestellt. Basierend auf der dargestellten Untersuchung, genügen alle drei Lösungen den definierten *MUST*-Kriterien C, D, H, I.

7. Prüfen der WANT-Kriterien

In diesem Abschnitt werden die Lösungsalternativen auf die *WANT*-Kriterien geprüft.

		Lösungsalternativen					
		Event Mesh (SAP)		Kafka		Event Hub (Microsoft)	
Kriterium	Faktor (%)	Zielerfüllung	Nutzwert	Zielerfüllung	Nutzwert	Zielerfüllung	Nutzwert
A	14.8	4	0.592	0	0.0	5	0.74
B	3.7	5	0.185	4	0.148	4	0.148
E	22.2	5	1.11	1	0.222	4	0.888
F	22.2	5	1.11	5	1.11	5	1.11
G	7.4	3	0.222	2	0.148	5	0.37
J	29.6	3	0.88	5	1.48	1	0.396
		Summen	4.099		3.108		3.652

Tabelle 3.5: Prüfung der Lösungsalternativen auf die *WANT*-Kriterien.

In Tb. 3.5 ist die Auswertung der Lösungsalternativen der *WANT*-Kriterien dargestellt. Darüber hinaus sind die erreichten Werte der Lösungsalternativen bezüglich der Zielerfüllungsskala aufgezeigt. Der Nutzwert eines Kriteriums entspricht dem Gewichtungsfaktors multipliziert mit dem Zielerfüllungsgrad. Anhand der Summe der Nutzwerte wird der Gesamtnutzwert der Lösungsalternative bestimmt.

Wie aus Tbl 3.5 hervorgeht, erreicht die SAP Lösung Event Mesh mit 4.099 den höchsten Nutzwert.

8. Bewertungsdiskussion

Im Folgenden werden die Gründe für die dargestellte Bewertung detailliert erläutert:

A: Die SAP besteht seit 1972 und hat eine Marktkapitalisierung von etwa 150 Milliarden Euro [37]. Microsoft besteht seit 1972 mit einer Marktkapitalisierung von etwa 2000 Milliarden Dollar [38]. Kafka ist Open Source Framework und wird von diversen Unternehmen wie Amazon, Microsoft und der SAP genutzt.

Microsoft als auch SAP sind Mega- und Big-Cap Unternehmen. Dies mindert die Wahrscheinlichkeit, dass Software-Lösungen dieser Unternehmen keinen Support oder neue Updates mehr erhalten. Kafka hingegen ist ein Open-Source Framwork, dass zwar viel Anwendung findet, aber nicht direkt durch ein Unternehmen verkauft wird.

B: Nach Gartner [39] handelt es sich bei SAP, um einen *Leader* für *Platform-as-a-Service (PaaS)* Systeme. Microsoft wird mit einer geringeren Bewertung ebenfalls als *Leader* eingestuft. Unter der Bedingung, dass Kafka von Microsoft genutzt wird, liegt eine identische Bewertung vor. Unter Verwendung von IBM, wird Kafka als *Visionaire* eingestuft.

E: Die SAP hat seit 2019 durchschnittlich 11.5 Sicherheitsupdates pro Jahr durchgeführt [40]. Kafka in dem gleichen Zeitraum 2, jeweils im Juli 2019 und 2020 [41]. Microsoft führt seit 2019 durchschnittlich 11 Sicherheitsupdates pro Jahr durch [42].

F: Event Mesh bietet OAuth, SAML, Basic und für interne Applikationen kann ein sogenannter User-Provided Service zur Authentifizierung verwendet werden. Event Hub unterstützt OAuth, SAS und eine Freigabe von Ports für die direkte Kommunikation. Kafka unterstützt in Verbindung mit IBM Cloud: SASL, Basic, SCRAM, OAuth und GSSAPI [43].

G: Die Dokumentation von SAP ist in Teilen zu allgemein verfasst. Die Trainingsangebot für Event Mesh ist sowohl für Node.js als auch Java in unter 2 Stunden durchführbar. Es sind wenig bis keine eigenen Programmierkenntnisse nötig. Microsoft bietet eine detaillierte Dokumentation, insbesondere hinsichtlich der programmatischen Umsetzung von Applikationen. Das Trainings-Angebot für Event Hub kann in 2 bis 3 Stunden durchgeführt werden. Es sind wenig eigene Programmierkenntnisse nötig. Kafka kann beispielsweise über die IBM MQ oder in Verbindung mit Microsoft Event Hub getestet werden [44]. Alternativ wird ein eigenes System benötigt. Dieser Vorgang kann mehrere Stunden dauern und benötigt extern bezogene Hardware. Eigene Programmierkenntnisse sind Voraussetzung.

J: Event Mesh unterstützt *Quality of Service (QoS)* Konfigurationen *at most once* (0) und *at least once* (1). Microsoft Event Hub unterstützt nur (1). Apache Kafka unterstützt (0), (1) und *exactly once* (2).

9. Auswahl der Lösung

Die Bewertung der *EM*-Lösungen zeigt, dass Event Mesh die bessere Wahl für dieses Messaging-System ist. Insbesondere die Tatsache, dass Event Mesh über MQTT und *QoS* (0) verfügt, hat die Entscheidung beeinflusst. MQTT und *QoS* (0) könnten Anwendung in der hochfrequenten Sensordatenübertragung finden. Weiterhin werden bereits diverse Services und Applikationen der SAP in der Gesamtlandschaft eingesetzt, siehe hierzu Abschnitt 3.1. Durch die Nutzung von Event Mesh sind keine Kompatibilitätsprobleme zu erwarten. Der Nutzwert von insgesamt 4.1 gegenüber 3.1 bei Kafka und 3.7 bei Microsoft verdeutlicht dieses Verhalten.

Die SAP *EM*-Lösung Event Mesh wird im weiteren Verlauf dieser Arbeit verwendet.

3.2 DATENGRUNDLAGE UND PARTITIONIERUNG

In diesem Kapitel wird der Fragestellung nachgegangen, ob doppelte Einträge im Datenbestand detektierbare Muster aufweisen.

Zunächst wird in Abschnitt 3.2.1 die Datenmenge und -Struktur dargelegt. Daraufhin wird in Abschnitt 3.2.2 eine Merkmalsselektion durchgeführt. In Abschnitt 3.2.3 wird eine Datentransformation mithilfe ausgewählter Distanzmaße durchgeführt. Abschließend wird in Abschnitt 3.2.4 eine Hypothese aufgestellt, welche mithilfe einer Faktoranalyse (Abschnitt 3.2.4.2) und einem partitionierenden Clustering-Verfahren (Abschnitt 3.2.4.3) geprüft wird.

3.2.1 *Datenmenge und Struktur*

In der relevanten Datenbanktabelle *ASSET* aus dem *assetCentral*-Schema sind 360 000 Einträge vorhanden, die keine Unikate Seriennummer aufweisen. Der daraus erzeugte Datensatz beinhaltet diese Duplikate, 21 Merkmale und wird fort folgend als *Dupl* bezeichnet. Für die spätere Analyse wird ein Vergleichsdatensatz (*Base*) ohne Duplikate, identischen Merkmalen und 350 000 Einträgen generiert. Weiterhin befinden sich in den Datensätzen, Einträge aus dem Jahr 2000 bis heute. Die Selektion der entsprechenden Datensätze erfolgt mithilfe von SQL-Anfragen.

Für *Base* werden alle nicht leeren und einmal vorkommenden Seriennummern selektiert. Von dieser Menge werden zufällig 350 000 Einträge ausgewählt, siehe Lst. A.1.

Für *Dupl* werden nur diejenigen Seriennummern augewählt, die nicht leer sind und mehr als einmal vorkommen, siehe Lst. A.2.

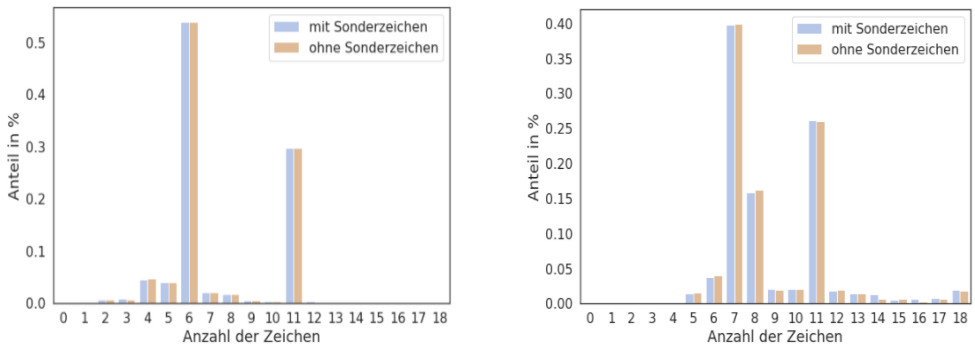
Beide Anfragen werden mithilfe von *JOIN* Klauseln um zusätzliche Merkmale diverser Tabellen erweitert.

3.2.2 *Merkmalsselektion*

In diesem Abschnitt wird eine Merkmalsselektion durchgeführt, um die spätere Analyse auf die relevanten Daten zu beschränken. Dabei wird zunächst die Seriennummer näher untersucht. Anschließend werden die vorhandenen Merkmale geprüft.

3.2.2.1 *Untersuchung - Seriennummer*

Zunächst wird geprüft, ob die vorliegenden Seriennummern dem *MU* Standard entsprechen. In dem Standard wird die Form der Seriennummer definiert. Zulässige Seriennummern haben eine Länge von 11 Zeichen und bestehen ausschließlich aus den Zeichen 0 bis 9 und A bis Z.



(a) Verteilung Seriennummerlänge, *Dupl*

(b) Verteilung Seriennummerlänge, *Base*

Abbildung 3.3: Gegenüberstellung der Seriennummerlänge mit und ohne Sonderzeichen für *Dupl* und *Base*.

In Abb. 3.3 wird ein Vergleich von *Dupl* und *Base* anhand der Seriennummernlänge unterteilt nach Seriennummern mit und ohne Sonderzeichen dargelegt. Visuell ist sowohl für *Dupl* als auch *Base* kein Unterschied zwischen Seriennummern mit oder ohne Sonderzeichen zu erkennen. Der Anteil der Seriennummern mit Länge 6 liegt bei etwa 50% bei *Dupl* (vgl. Abb. 3.3a) und der Länge 7 bei etwa 40% bei *Base* (vgl.Abb. 3.3b). Anhand des *MU* Standards müsste jedoch der größte Anteil der Daten eine Seriennummerlänge von 11 Zeichen aufweisen.

Um diesen Effekt weiter zu untersuchen, werden die Daten um Sonderzeichen bereinigt, in Großbuchstaben umgewandelt und anhand des Erzeugungsjahres in Zeitintervalle von 5 Jahren ausgehend von dem Jahr 2001 unterteilt.

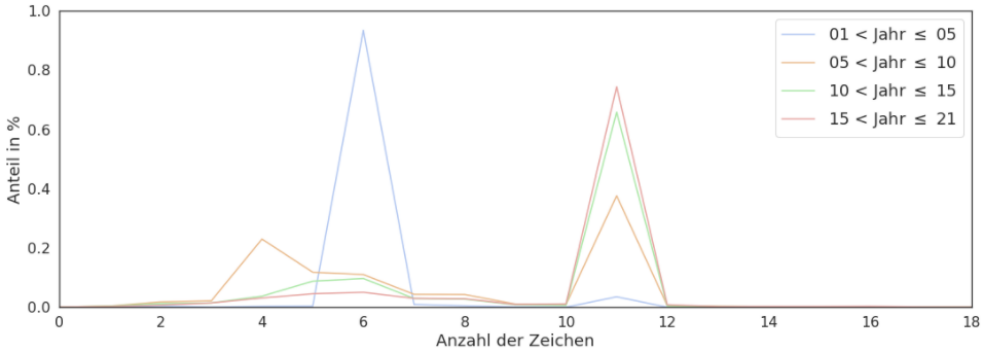


Abbildung 3.4: Seriennummerlänge unterteilt in 5 Jahresintervalle, *Dupl*

Die vorliegende Abb. 3.4 gibt Auskunft über die Entwicklung der Seriennummerlänge in vier Zeitintervallen. Der Anteil der Seriennummern mit 11 Zeichen hat zwischen 2005 und 2010 um etwa 40% und zwischen 2005 und 2021 um etwa 80% zugenommen. In den Intervallen nach 2005 nähert sich die Seriennummerlänge zunehmend dem *MU* Standard von 11 Zeichen an. Ein möglicher Grund für die dargestellte Entwicklung könnte sein, dass die Einführung des *MU* Standard nach 2005 stattfand.

Schlussfolgerung - Seriennummer

Die Ergebnisse der Untersuchung von den Sonderzeichen lassen den Schluss zu, dass die Unterschiede zwischen Seriennummerlängen mit und ohne Sonderzeichen vernachlässigbar sind. Des Weiteren konnte durch die Untersuchung der zeitlichen Entwicklung der Seriennummer, eine Veränderung der Längen identifiziert werden. Es ist anzunehmen, dass nach 2005 der *MU* Standard eingeführt wurde und dies zu der gezeigten Entwicklung führt.

Für die weitere Analyse werden aufgrund des Ziels dieser Arbeit, der Identifizierung von neuen Daten, alle Daten mit *Erzeugungsjahr* $\leq$ 2005 verworfen. Weiterhin werden Sonderzeichen entfernt und nur diejenigen Daten berücksichtigt, die nach dem *MU* Standard aus 11 Zeichen bestehen. Durch diese Einschränkungen wurde *Dupl* von 360 000 auf 96 690 Einträge reduziert.

3.2.2.2 *Untersuchung - Merkmale*

In diesem Abschnitt, werden die Merkmale auf leere Einträge (NULL) untersucht. Das Ziel dieser Untersuchung besteht in der Identifizierung von Merkmalen, welche wenig Einfluss auf die Erklärbarkeit der Duplikatsbildung haben. Dazu werden die Merkmale von *Dupl* und *Base* auf den Anteil an NULL Werten untersucht. Zusätzlich wird die Differenz des prozentualen Anteils an NULL zwischen den Datensätzen verglichen. Mithilfe dieser Differenz werden diejenigen Merkmale identifiziert, welche in *Dupl* im Verhältnis zu *Base* weniger NULL Werte beinhalten. Darauf basierend werden folgende Kriterien definiert:

$$\begin{aligned} x_i &\leq \tilde{Z} \\ z_i - x_i &\geq \mu - \sigma \end{aligned} \tag{3.2}$$

wobei x_i dem prozentuale Anteil von NULL eines Merkmal aus *Dupl* und z_i aus *Base* entspricht. Weiterhin ist $\tilde{Z}$ der Median von NULL in *Base*, μ der Mittelwert und σ die Standardabweichung der Differenz. Ein Merkmal wird verworfen sofern eines der beiden Kriterien nicht erfüllt ist.

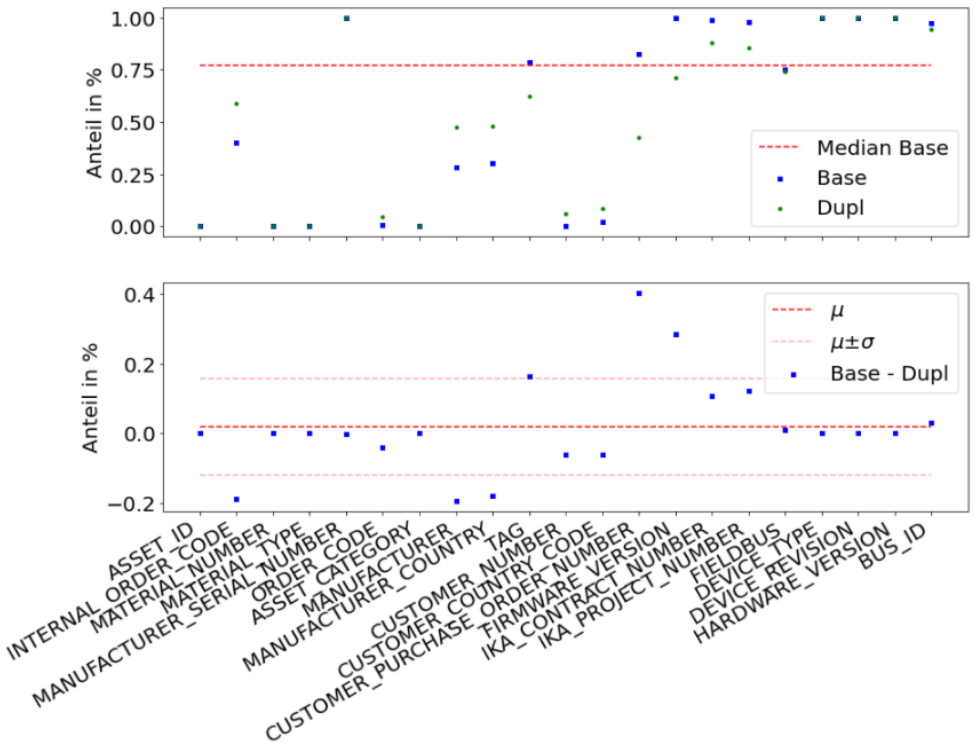


Abbildung 3.5: Vergleich zwischen *Base* und *Dupl* anhand der prozentualen Anteile der NULL-Werte je Merkmal.

Abb. 3.5 wird zur Überprüfung der in Formel 3.2 definierten Kriterien verwendet. Aus der oberen Abbildung geht hervor, dass 7 Merkmale zu über 80% aus leeren Einträgen bestehen. Diese Merkmale liegen über dem Grenzwert $\tilde{Z}$.

Der unteren Abbildung wird entnommen, dass insbesondere 2 Merkmale in *Dupl* im Vergleich zu *Base* weniger NULL-Werte beinhalten mit $z_i - x_i \geq \mu + \sigma$. Basierend auf dem zweiten Kriterium werden 3 weitere Merkmale identifiziert, welche unter dem Grenzwert $z_i - x_i \geq \mu - \sigma$ liegen. Darüber hinaus wird für die Merkmale *asset_id* (eindeutig für jedes Gerät) und *asset_category* ($\geq$ 99% identische Werte) kein Informationsgewinn angenommen.

Zusammenfassend sind nach der Selektion insbesondere die folgenden 9 Merkmale von Interesse: MATERIAL_NUMBER (MN), MATERIAL_TYPE (MT), ORDER_CODE (OC), CUSTOMER_NUMBER (CN), FIELDBUS (FB), FIRMWARE_VERSION (FV), CUSTOMER_COUNTRY_CODE (CCC), TAG (T) CUSTOMER_PURCHASE_ORDER_NUMBER (CPON).

3.2.3 *Transformation der Daten*

In diesem Abschnitt wird die angewandte Methode zur Transformation der Daten erläutert.

Zunächst werden die Datentypen der nach Abschnitt 3.2.2 verbliebenen Merkmale dargelegt.

Spaltennamen	MN	MT	OC	T	CN	CCC	CPON	FV	FB
Datentypen	Int	String	String	String	String	String	String	String	String

Tabelle 3.6: Auflistung der Merkmale und Datentypen

Tabelle 3.6 lässt sich entnehmen, dass von den verbliebenen 9 Merkmalen, 8 String und 1 Int Datentyp vorhanden sind. Dies ist insbesondere für die Datentransformation relevant, da textuelle Werte eine gesonderte Betrachtungsweise erfordern. Für die Transformation werden die in Kapitel 2.2.3 vorgestellten Distanzmaße verwendet. Der Ablauf ist wie folgt (vgl. Lst. A.4):

Zunächst werden die Daten zur Identifizierung des ersten Eintrags nach Seriennummer und dem Erzeugungs- und Modifizierdatum sortiert. Anschließend wird die Damerau-Levenshtein- und Hamming-Distanz zwischen dem ersten Eintrag und den Duplikaten bestimmt Dieser Prozess wird iterativ für alle Seriennummern wiederholt und die Ergebnisse gespeichert.

	MN	MT	OC	T	CN	CCC	CPON	FV	FB
1.	56004142	HAWA	DTRANSRW	dent.Nr. 4226	30050497	CH	Mail	NaN	NaN
	56004119	HAWA	DTRANSRW	dent.Nr. 4226	30050497	CH	NaN	NaN	NaN
$d_{x,y}$	2	0	0	0	0	0	4	0	0
$d^{rel}_{x,y}$	0.25	0	0	0	0	0	1.0	0	0
d_h	1	0	0	0	0	0	1	0	0

$d^{rel}_h = 0.22$

Tabelle 3.7: Beispielhafte Berechnung der Distanzmaße.

Das angeführte Beispiel der Transformation aus Tbl. 3.7 zeigt das Verhalten der Transformation. Mithilfe der relativen Damerau-Levenshtein-Distanz $d^{rel}_{x,y}$, werden Abweichungen der Merkmalsausprägungen quantifiziert. Die relative Hamming-Distanz d^{rel}_{h} spiegelt die Abweichung der Datensätze zueinander wider. Die beiden Transformationen wurden gewählt, um sowohl innere Abweichungen als auch eine Gesamtabweichung der Daten zu ermitteln.

Um das Verhältnis beider Distanzmaße pro Datensatz akkumuliert zu betrachten, wird eine neue Zielgröße eingeführt. Die Zielgröße bildet den Mittelwert der Damerau-Levenshtein-Distanz $d^{rel}_{x,y}$ pro Datensatz ab, es gilt

$$T = \overline{d^{rel}_{x,y}} = \frac{1}{n} \sum_{i=1}^{n} d^{rel}_{x,y}(i), \tag{3.3}$$

wobei n der Anzahl der Merkmale entspricht.

Basierend auf dem in Formel 3.3 erzeugten Mittelwert wird angenommen, dass die Hamming- und die Damerau-Levenshtein-Distanz akkumuliert abgebildet werden. Diese Annahme wird mithilfe einer einfachen Regression [10] und dem Korrelationskoeffizient nach Bravais-Pearson [9] geprüft.

Sei $x_1, ..., x_n$, $x_i \in d^{rel}_{h}$ und $y_1, ..., y_n$, $y_i \in T$. Es soll die Geradengleichung $\hat{y} = b_0 + b_1 x$ geschätzt werden, wobei b_0 der Intercept und b_1 die Steigung der Geraden ist.

Listing 3.1: Regressionsberechnung für den linearen Zusammenhang von der Hamming- und Damerau-Levenshtein-Distanz.

```
from scipy.stat import linregress

x, y = data.hamming_dist.values, data.lev_dist_row_mean.values
slope, intercept, r_value, p_value, std_err = linregress(x,y)
# slope: 0.91, intercept: -0.033, r_value: 0.961, p_value: 0.001
```

In Listing 3.1 werden die Ergebnisse einer Linearen Regression der beiden Distanzmaße aufgeführt. Dabei entspricht der Intercept $b_0 = -0.033$ und die Steigung $b_1 = 0.91$. Weiterhin zeigt $r = 0.961$ eine hohe Korrelation der beiden Werte. Aufgrund des stark linearen Verhaltens wird angenommen, dass die Akkumulation durch T die Hamming- und Damerau-Levenshtein-Distanz ausreichend repräsentiert.

3.2.4 *Modellauswahl*

In diesem Abschnitt wird zunächst, basierend auf Stichprobenuntersuchungen und interner Expertenmeinungen, eine Hypothese über die Daten formuliert. Anschließend wird eine Faktoranalyse durchgeführt, um Zusammenhänge zwischen den Daten deutlicher darzustellen. Abschließend wird ein partitionierendes Clustering (K-Means) durchgeführt. Es wird erwartet, eine Gruppe an Daten zu identifizieren, die sich eindeutig von anderen Gruppen unterscheidet und der Hypothese entspricht.

Die erarbeitete Methode wird in Form eines Modells in Kapitel 4.1 in einem Anwendungsfall verwendet. Der hier verwendete Datensatz (*Dist*) beinhaltet nach der Distanzberechnung 48 762 Einträge und die in Tbl. 3.6 aufgeführten Merkmale.

3.2.4.1 *Hypothese*

Anhand von Stichprobenuntersuchungen und Konsultation interner Experten, wurde ein möglicher Grenzwert angenommen. Dabei wurde untersucht, ob die Abweichung eines neuen Datensatzes auf Tipp- oder Systemfehler zurückzuführen sein könnte oder ob es sich um ein neues Gerät handelt und damit ein falscher bzw. nicht eindeutig zuzuordnender Eintrag vorliegt. Die Annahme umfasst zwei Teile.

Zunächst gilt es die systembedingten Zusammenhänge der Merkmale zu berücksichtigen.

MN - MT : MT repräsentiert unterschiedliche Materialtypen. Jedem Typ wird eine Gruppe an Nummern (MN) zugewiesen, um verschiedene Ausprägungen von Geräten oder Services eindeutig zu identifizieren.

CN - CCC - CPON : Jedem Kunden werden eine oder mehrere Länder (CCC) zugewiesen in denen dieser Kunde agiert. Weiterhin beinhaltet die CPON zur Identifizierung des Kunden, einen Teil der Kundennummer.

FV - FB : Der FB wird regelmäßig geupdated wodurch die FV gegebenenfalls eine neue Version erhält.

T ist ein zusätzliches Feld, dass einer Kommentarspalte entspricht. Folglich ist für T kein direkter Zusammenhang mit anderen Merkmalen vorgesehen. Die unterschiedlichen Ausprägungen der Geräte sind aufgrund einer systeminternen Abstraktion, teilweise in dem OC enthalten.

Weiterhin wird angenommen, dass Merkmale mit einer geringen Abweichung (maximal $d^{rel}_{x,y} \leq 0.3$) zu dem ursprünglichen Gerät einem Tippfehler geschuldet sein können. Tippfehler sind systematisch nicht berücksichtigt, da manuelle Einträge nur bedingt geprüft werden.

Entsprechend der obigen Zusammenhänge wird schließlich angenommen, dass ein automatisch zusammenführbares Duplikat vorliegt wenn

$$\mathbb{1}_{Duplikat} = \begin{cases} 1 & \text{wenn } T \leq 0.25 + \varepsilon \\ 0, & \text{wenn } T > 0.25 + \varepsilon \end{cases}, \tag{3.4}$$

wobei $\mathbb{1}_{Duplikat}$ eine Indikatorfunktion repräsentiert, die ein Duplikat annimmt für $T \leq 0.25 + \varepsilon$ und kein Duplikat in sonstigen Fällen. Der Fehler ε wird als Toleranz berücksichtigt und auf $\varepsilon \leq 0.5$ festgelegt.

Im Fall von $T \leq 0.25$, treten alle systembedingten Zusammenhänge auf (7 Merkmale) und die innere Abweichung entspricht dem maximal zulässigen Wert. Die Toleranz wurde eingeführt, ob im Vorfeld nicht absehbare Konstellation zu berücksichtigen. Folglich wird die maximal zulässige Abweichung T mit $T \leq 0.3$ festgelegt.

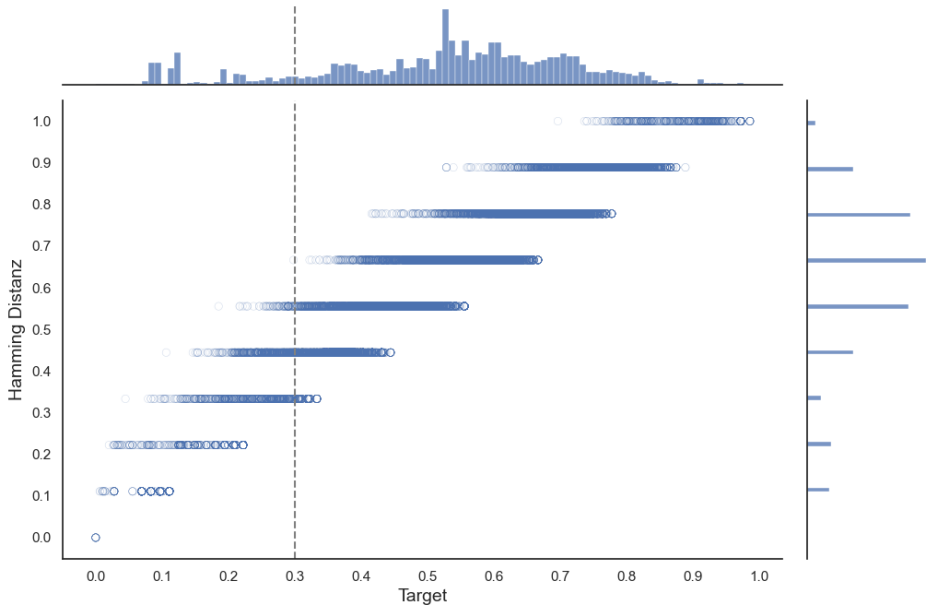


Abbildung 3.6: Abbildung des Zusammenhanges der Hamming-Distanz im Verhältnis zu der Zielgröße T. Der angenommene Grenzwert von $T \leq 0.3$ wird in Form einer vertikalen Linie dargestellt.

In Abbildung 3.6 ist der lineare Zusammenhang zwischen der Hamming-Distanz und dem Zielwert T sowie Histogramme der jeweiligen Verteilung dargestellt. Es fällt auf, dass vereinzelt eine Hamming-Distanz von $d_h = 0.66$ vorliegt. Weiterhin sind in dem oberen Histogramm, lokale Spitzen der Zielgröße T insbesondere im Bereich $T \leq 0.2$ zu erkennen. Erklären lässt sich $d_h = 0.66$ mit den programmatischen Zusammenhängen der Merkmale indem eine Änderung bei drei Kombinationen bis zu 7 Merkmale beeinflusst. In Bezug auf die Annahme der Tippfehler, könnte $T \leq 0.2$ ein Indiz für die angenommene Gruppe sein.

3.2.4.2 *Faktoranalyse*

In diesem Abschnitt wird die Faktoranalyse durchgeführt.

Zunächst wird mithilfe des Bartlett-Tests [45] und dem Kaiser-Meyer-Olkin-Kriterium (KMO) [46] geprüft, ob eine signifikante Interkorrelation der Merkmale vorliegt.

Listing 3.2: Bartlett-Test auf Spherizität zur Prüfung der Eignung einer Faktoranalyse

```
from factor_analyze import bartlett_sphericity, kmo
chi_square_value, p_value = bartlett_sphericity(X)
# 98375, 0.00
kmo = kmo(X)
# 0.64
```

In Listing 3.2 wird die Durchführung des Bartlett-Test und des KMO gezeigt. Mit $p = 0.00$ liegt ein signifikantes Ergebnis vor. Weiterhin liegt das Maß der partiellen Korrelationen einer Stichprobe nach dem KMO mit 0.64 über den Empfehlungen von > 0.5 [47] und > 0.6 [48]. Eine Faktoranalyse scheint geeignet.

Nun wird eine Faktoranalyse mit einer orthogonalen Rotation durchgeführt, um zunächst anhand der Eigenwerte und Kommunalitäten, eine Selektion der relevanten Merkmale durchzuführen.

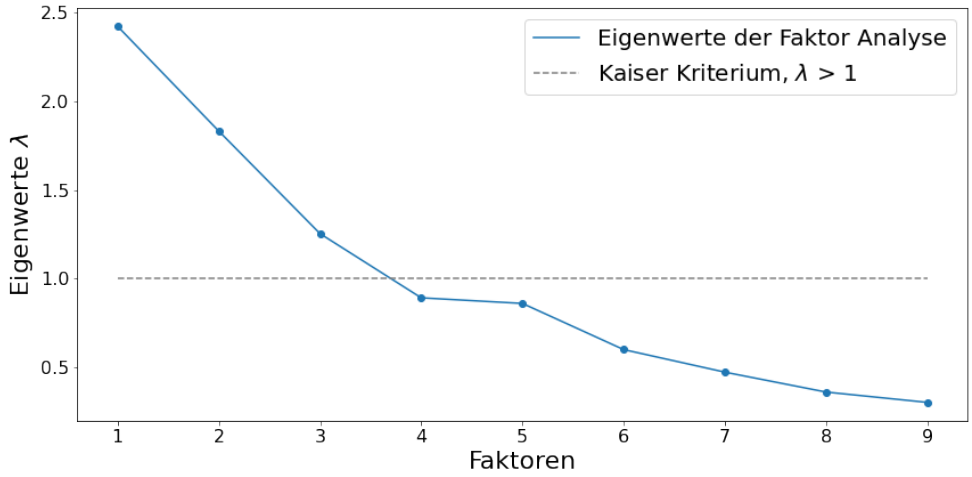


Abbildung 3.7: Screeplot der Eigenwerte λ der Faktoranalyse. Zusätzlich wird der Grenzwert von 1.0 nach dem Kaiser-Kriterium dargestellt.

In Abbildung 3.7 werden die Eigenwerte der Faktoren dargestellt. Nach dem Grenzwert des Guttman-Kaiser-Kriteriums [49] von $\lambda < 1.0$, sind 3 Faktoren geeignet.

Merkmal	MN	OC	T	CN	CCC	CPON	FV	FB	MT
Kommunalität	0.26	0.15	0.092	0.61	0.59	0.41	0.76	0.63	0.48

Tabelle 3.8: Kommunalitäten der Merkmale

In Tbl. 3.8 werden die Kommunalitäten h_i^2 der Merkmale dargestellt. Mit Kommunalitäten von $h_i^2 > 0.2$ wird angenommen, dass die Merkmale ausreichend gemeinsame Varianz erklären [50]. Folglich werden die Merkmale OC mit $h_{OC}^2 = 0.15$ und T mit $h_T^2 = 0.092$ verworfen.

Im Folgenden wird die Faktoranalyse mit den adjustierten Parametern erneut durchgeführt. Es werden $q = 3$ Faktoren, $p = 7$ Merkmale, eine nicht-orthogonale Rotation *promax* und die prinzipiellen Faktoren-Methode verwendet, die entsprechende Durchführung kann Lst. A.5 entnommen werden.

Merkmal	MN	CN	CCC	CPON	FV	FB	MT	Prop. Varianz (s_i^2)
FA_1	-0.39	-0.11	-0.21	**0.52**	**0.90**	**0.87**	0.37	0.31
FA_2	-0.17	**0.88**	**0.84**	**0.47**	-0.17	-0.19	-0.01	0.26
FA_3	**0.93**	-0.04	-0.17	0.20	-0.15	-0.14	**0.65**	0.19
Max. Ladung	FA3	FA2	FA2	FA1	FA1	FA1	FA3	$\sum s_i^2 = 0.77$

Tabelle 3.9: Faktorladungen der adjustierten Faktoranalyse.

In Tabelle 3.9 sind diejenigen Ladungen, die erklärte Varianz des Merkmals des entsprechenden Faktors, mit > 0.45 hervorgehoben. Nach Comrey [51] gilt, je größer die Ladung, desto mehr entspricht das Merkmal einem reinem Maß für den Faktor. Dabei wird eine Ladung > 0.45 als angemessen betrachtet mit 20% überlappender Varianz.

Es ist ersichtlich, dass FV und FB ein ausgezeichnetes Maß (> 0.71) für FA_1 entsprechen. CPON hingegen lädt angemessen (> 0.45) auf FA_1 mit 0.52 und FA_2 mit 0.47.

CN und CPON laden ausgezeichnet auf FA_2 und MN und MT laden respektive ausgezeichnet und sehr gut (> 0.63) auf FA_3.

FA_1 weist mit $s_i^2 = 0.31$ die größte proportionale Varianz des Modelles auf. Akummuliert erklären die Faktoren 77% der gesamten Varianz der Stichprobe.

Die erzeugten Ladungen bestätigen die in Abschnitt 3.2.4.1 aufgestellten systembedingten Zusammenhänge der Merkmale. Es liegen hohe Korrelationen zwischen FV und FB, CN und CCC, sowie MN und MT vor. Lediglich CPON weist vergleichbar große Zusammenhänge zu FA_1 und FA_2 auf.

Die aufgestellte Hypothese wird basierend auf den Ergebnissen der Faktoranalyse als angemessen angenommen.

3.2.4.3 *Partitionierendes Clustering*

In diesem Abschnitt wird das K-Means-Clustering-Verfahren durchgeführt. Die Durchführung erfolgt sowohl für den Basisdatensatz (*Dist*) als auch den mithilfe der Faktoranalyse transformierten Daten (*FA*).

Zunächst wird ein K-Means Clustering mit aufsteigender Cluster-Anzahl k durchgeführt. Anhand der Ellenbogen-Methode wird mithilfe der Summe der quadratischen Abweichung (SSE) und dem SSWC aus Kapitel 2.2.4.2 ein optimales k gewählt. Mithilfe der Ellenbogen-Methode wird diejenige Anzahl an Clustern identifiziert die angibt, dass durch hinzufügen eines weiteren Clusters kein zusätzlicher Informationsgewinn zu erwarten ist [52].

Für SSE gilt $SSE = \sum_i^k \sum_{p \in C_i} (p - c_i)^2$ wobei C_i den Daten innerhalb eines Clusters und c dem Zentroid entspricht [28]. Der verwendete Algorithmus kann Lst. A.6 entnommen werden.

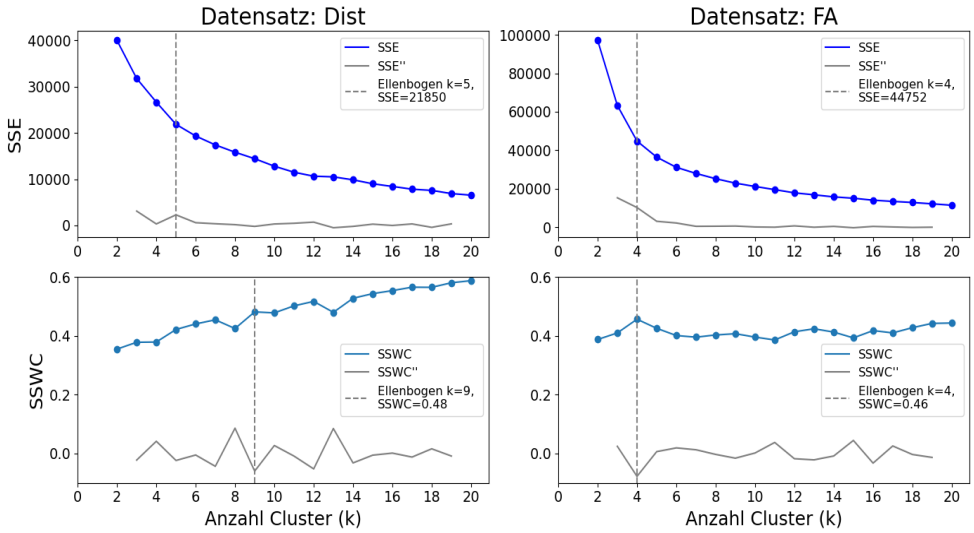


Abbildung 3.8: Ellenbogen Graph und SSWC des K-Means Clustering für aufsteigende k. Angewandt auf *Dist* und *FA*.

In Abb. 3.8 werden die Ellenbogen-Graphen für den *Dist*- und *FA*-Datensatz anhand des SSE und SSWC dargestellt. *Dist* weist unterschiedliche Ellenbogen für SSE und SSWC mit $k = 5$ und $k = 9$ auf. *FA* hingegen zeigt übereinstimmende Ellenbogen mit $k = 4$.

Aufgrund des unklaren Ergebnisses der Ellenbogen-Methode für *Dist* wird im Folgenden $k = 4$ gewählt und *FA* verwendet.

Nun wird die Methodik mit $k = 4$ Clustern angewandt und die Güte der Cluster mit SSWC untersucht. Die Inititalisierung der Zentroide erfolgt zufällig. Anschließend werden die Cluster den Daten zugeordnet und eine Silhouette-Analyse über den gesamten Datensatz *FA* durchgeführt.

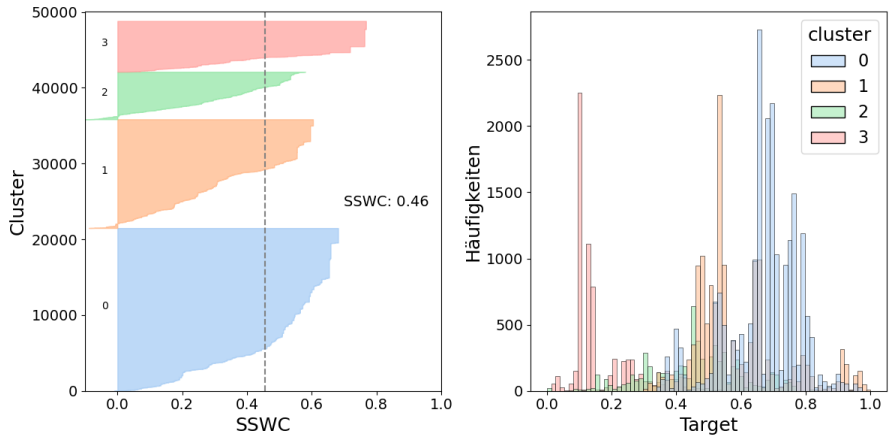


Abbildung 3.9: Silhouette Graph des K-Means Verfahrens mit $k = 4$ der FA-Daten unterteilt nach den zugehörigen Clustern.

Die vorliegende Abb. 3.9 gibt Auskunft über die Verteilung und Silhouette-Koeffizienten (SSWC) der Daten, gruppiert nach den jeweiligen Clustern. Cluster c_0 und c_2 weisen die beste Struktur der vorliegenden Cluster auf. Diese Beobachtung kann anhand des Histogramms in Abb. 3.9 nachvollzogen werden. Die Stichproben scheinen nah beieinander zu liegen. In Cluster c_1, c_3 sind Stichproben mit negativem SSWC zu erkennen. Dies suggeriert einen kleineren relativen Abstand der Stichproben zu c_0 oder c_2. Die Zuordnung dieser Stichproben könnte fehlerhaft sein.

Cluster	$\overline{SSWC}_{c_i}$	$\overline{T}$	N
0	0.59	0.17	6695
1	0.32	0.46	6255
2	0.51	0.67	21433
3	0.37	0.58	14379
$\overline{SSWC}$	0.46		

Tabelle 3.10: Ergebnisse des Clusteringverfahrens

In Tabelle 3.10 wird eine Zusammenfassung des $\overline{SSWC}$ und $\overline{T}$ der einzelnen Cluster aufgeführt. Cluster c_0 und c_2 zeigen eine vernünftige Struktur mit $0.51 \leq \overline{SSWC} \leq 0.71$. Cluster c_1 und c_3 zeigen eine schwache Struktur mit $0.26 \leq \overline{SSWC} \leq 0.50$ [53]. Es fällt auf, dass zwei übergeordnete Partitionen mit $\overline{T}_0 = 0.17$ und $\overline{T}_{1,2,3} \geq 0.32$ zu erkennen sind. Die Annahme liegt nahe, dass c_0 solche Stichproben repräsentiert, welche als Duplikat interpretiert werden können. Demnach würden c_1, c_2, c_3 diejenigen Stichproben zugeordnet, bei denen es sich um ein neues Gerät handeln könnte.

Abschließend werden die Ergebnisse der in Abschnitt 3.2.4.1 aufgestellten Hypothese gegenübergestellt. Basierend auf dem Grenzwert $T \leq 0.3$ entsprechen $n = 6772$ Daten der angenommenen Definition eines Duplikates.

	$\overline{T} \leq 0.3$	
Cluster	Anteil der Duplikate (%)	N
0	87.5	5929
1	12.5	843
2	0	0
3	0	0

Tabelle 3.11: Auflistung der prozentualen Anteile der Duplikate je Cluster

Aus Tbl. 3.10 geht hervor, dass Cluster c_0 zu 87.5% aus angenommenen Duplikaten besteht. Die übrigen 12.5% der Duplikate sind Cluster c_1 zugeordnet. In c_0 ist der durchschnittliche Zielwert $\overline{T}_0 = 0.17$ (vgl. Tbl 3.10) und liegt unterhalb des angenommenen Grenzwertes. $\overline{T}_0 = 0.17$ entspricht jedoch dem systembedingten Zusammenhang von 5 Merkmalen unter der maximalen zulässigen Abweichung $d^{rel}_{x,y} \leq 0.3$ mit $\frac{5*0.3}{9} = 0.1667$. Die maximale Abweichung der Merkmale könnte von 7 auf 5 reduziert werden.

Es ist anzunehmen dass die aufgestellte Hypothese der Duplikate basierend auf Tippfehlern und den systembedingten Zusammenhängen der Merkmale angemessen ist.

3.2.4.4 *Zusammenfassung*

Zum Abschluss dieses Kapitels werden die Ergebnisse zusammengefasst und anhand der aufgestellten Hypothesen dargelegt.

Nach Durchführung einer Merkmalsselektion in Abschnitt 3.2.2 konnte der Merkmalsraum von 21 auf 9 Merkmale reduziert werden. Die Untersuchung der Seriennummerlänge ergab eine Diskrepanz der Daten insbesondere vor 2005. Beruhend auf dem Ziel, der Einordnung neuer Daten, wurde die Stichprobengröße von 361.000 auf etwa 100.000 eingegrenzt.

Anschließend wurden Hypothesen zu den technischen Zusammenhängen der Merkmale aufgestellt. Angenommen wurden systembedingte Zusammenhänge von insgesamt 3 Kombinationen an Merkmalen sowie ein möglicher Grenzwert anhand von Stichprobenuntersuchungen.

Die aufgestellten Hypothesen wurden mithilfe einer Faktoranalyse in Abschnitt 3.2.4.2 und dem K-Means Clustering-Verfahren in Abschnitt 3.2.4.3 geprüft. Hierzu wurden die Ergebnisse der relativen Damerau-Levenshtein-Distanz mit 48762 Einträgen verwendet.

Die systembedingten Zusammenhänge der Merkmale konnten mithilfe der Faktoranalyse bestätigt werden. Der gewählte Grenzwert könnte anhand der Ergebnisse des Clustering-Verfahrens kleiner gewählt werden.

3.3 KONZEPTIONIERUNG EINER ARCHITEKTUR

In diesem Abschnitt wird die in Kapitel 3.1 vorgestellte Gesamtarchitektur aus Abbildung 3.2 angepasst. Es wird die in Kapitel 3.1.2 ausgewählte *EM*-Lösung verwendet. Zunächst werden Ziele für die Architektur definiert.

Ziel dieser Architektur ist es, Daten mithilfe eines *P/S*-Mechanismus asynchron und in Echtzeit zu übertragen.

Die verwendete Software und Methoden sollen in der bestehenden Systemlandschaft integrier- und erweiterbar sein.

Um eine möglichst hohe Kompatibilität innerhalb der Architektur zu erreichen, sollen SAP eigene Bibliotheken und Softwarelösungen verwendet werden.

Die Architektur soll sowohl Batch- als auch Einzelnachrichten verarbeiten und potenziellen *Subscribern* zur Verfügung stellen.

Schließlich soll die Möglichkeit bestehen, vor oder nach Versenden der Nachrichten, die Daten mit ausgewählten Programmiersprachen verarbeiten zu können.

3.3.1 *Aufbau der Zielarchitektur*

In diesem Abschnitt wird die erarbeitete Architektur schrittweise vorgestellt.

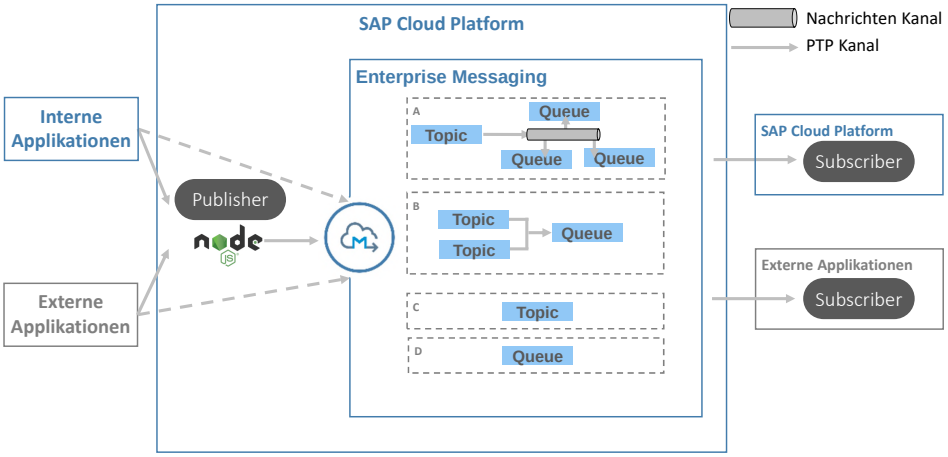


Abbildung 3.10: Allgemeines Lösungskonzept für eine lose gekoppelt und Echtzeit kommunikationsfähige Publish/Subscribe Architektur.

In Abb. 3.10 ist das erarbeitete Lösungskonzept dargestellt. Die linke Seite zeigt interne- und externe Applikationen mit einer *PTP*-Verbindung zu dem Node.js Publisher, welcher auf der *BTP* bereitgestellt wird. Das *EM*-System mittig in Abb. 3.10, wird von dem Publisher mit Nachrichten versorgt. Auch eine direkte Kommunikation der Applikationen mit dem *EM*-System wird unterstützt. Nachfolgend werden die einzelnen Komponenten näher erläutert.

Publisher

Eine Node.js Applikation wird vorgeschlagen, da diese direkt in der *BTP* bereitgestellt werden kann. Um eine möglichst unkomplizierte Kommunikation mit dieser Applikation zu ermöglichen, wird ein Webserver mit gängigen *API* Methoden eingesetzt.

Für jede eingehende Nachricht wird ein eigener Prozess initiiert und dadurch eine parallele Verarbeitung ermöglicht. Nach [54] liegt die Auslastung des System bei 70 gleichzeitigen Anfragen pro Sekunde bei etwa 50%. Demzufolge scheint eine parallele Verarbeitung von Anfragen ausreichend performant durchführbar.

Node.js wird verwendet, da die SAP Bibliotheken wie *sap/xb-msg-env*, *sap/xb-msg-amqp-v100* und *sap/xb-msg-mqtt-v311* explizit für *P/S* Systeme bereitstellt [55]. Es werden multiple Kommunikationsprotokolle wie *Hypertext Transfer Protocol (HTTP)*, *Message Queuing Telemetry Transport (MQTT)* und *Advanced Message Queueing Protocol (AMQP)* unterstützt.

HTTP ist ein Basisprotokoll, welches für die Kommunikation zwischen Webservices verwendet wird. Die Kommunikation zwischen Client und Server steht im Vordergrund. Es wird synchrone Kommunikation verwendet [56].

AMQP ist ein applikationsorientiertes Protokoll. Es ist leichtgewichtig und nutzt asynchrone Kommunikation [57]. Wie auch *AMQP* ist *MQTT* ein leichtgewichtiges Kommunikationsprotokoll, welches auf Publish/Subscribe Methoden basiert [58]. Unterschieden wird zwischen *AMQP* und *MQTT* anhand der unterstützten Nachrichtengröße. *MQTT* unterstützt Nachrichten mit $\leq$ 10 Byte. *MQTT* ist insbesondere für die Kommunikation zwischen *IoT*-Geräten geeignet.

Die Kommunikation mit dem Webserver erfordert für interne wie auch externe Applikationen eine Bearer Authentifizierung. Dazu wird eine *GET* Anfrage an den Webserver gesendet und nach erfolgreicher Authentifizierung, ein Bearer Token zurückgesendet. Anschließend wird dieses Token für die Kommunikation verwendet. Eine beispielhafte Umsetzung ist in Lst. A.7 vorzufinden.

Für die Publizierung von Nachrichten wird nach erfolgreicher Authentifizierung eine Verbindung mit dem *EM*-System hergestellt. Je nach Konfiguration der entsprechenden Queue oder Topic kann No-, Basic- oder Bearer-Authentifizierung verwendet werden. Innerhalb der *BTP* kann alternativ ein sogenannter User-provided Service verwendet werden, welcher der Node.js Applikation die Rechte zur Kommunikation mit dem *EM*-System bereitstellt. Anschließend wird ein Buffer mit der Nachricht beschrieben und entsprechend der verwendeten Bibliothek mit *HTTP*, *MQTT* oder *AMQP* versendet.

Im Folgenden wird die Umsetzung eines Nodejs Publisher anhand eines *AMQP* Publisher erläutert. Es werden Teile des Programmcode aufgeführt, der gesamte Programmcode ist in Lst. A.9 vorzufinden.

Listing 3.3: Nodejs Webserver mit express

```
1 const express = require("express"); // App Bibliothek
2 const app = express(); // Starte app
3 const PORT = process.env.PORT || 8080;
4 app.post('/', (req, res) => {  ...  } // App Methode
5 app.listen(PORT, function () {  ...  } // App starten
```

Wie in Lst. 3.3 dargestellt wird eine Applikation *app* in Zeile 2 initialisiert. Zeile 4 zeigt eine Methode der *app* um *POST* Anfragen verarbeiten zu können. In Zeile 5 wird die *app* gestartet und wartet auf Anfragen. Die Applikation ist nun erreichbar unter der im *manifest.yml* festgelegten Route

```
- route: producer-thesis-single-long-term-amqp-host.<host>.com
```

siehe hierzu Lst. A.8 für die komplette Definition. Anschließend wird die *EM* Umgebung initiiert.

Listing 3.4: Nodejs Messaging Umgebung

```
1 const AMQP = require('@sap/xb-msg-amqp-v100');
2 const service = 'emLennartTest'; // Messaging Service
3 function send(tasks, client, assetId, timestamp, run_id) {
4       const stream = client.ostream(id); // Ausgehender Stream
5       if (!stream.write(message)) // Senden des Inhalts
6 }
7 const client = new AMQP.Client(env.msgClientOptions(service, [], [
    'myOutB']));
8 client.connect()
```

In Lst. 3.4 sind Auszüge des *Messaging* Codes dargestellt. Zeile 1 initiiert die Umgebung. In Zeile 2 wird der Name des *EM*-Systems definiert. Anhand des Servicenamen kann mithilfe eines User-provided Service,

```
services:
  - xsuaaThesisProducer
```

eine Verbindung zwischen dem Publisher und dem *EM*-System ohne weitere Authentifizierungsschritte aufgebaut werden. In Zeile 3 wird eine Funktion definiert, die nach eintreffen einer *POST* Anfrage an *app.post*, die Inhalte der Anfrage und einen aktuellen Zeitstempel publiziert. In Zeile 7-8 wird das Zielsystem anhand des *service* identifiziert, initialisiert und eine Verbindung mit dem *EM*-System hergestellt. Die Differenzierung zwischen den Protokollen wird anhand des verwendeten *Client* durchgeführt. In dem aufgeführten Auszug wird die *AMQP* spezifische Bibliothek verwendet.

3.3.1.1 *EM-System*

Das *EM*-System beinhaltet wie in Abb. 3.10 dargestellt, vier Kategorien zur Nachrichtenverarbeitung [34].

A steht für die Variante von einem Topic, welchem drei Queues subskribiert sind. Zu der Topic publizierte Nachrichten werden über einen Message Bus an für alle subskribierten Queues vervielfältigt. Diese Kategorie wird genutzt, sofern multiple Applikationen, Nachrichten einer Topic konsumieren.

B illustriert eine Variante in der eine Queue multiple Topics subskribiert. Dadurch können Daten unterschiedlicher Herkunft akkumuliert einer Queue zur Verfügung gestellt werden. Diese Kategorie wird genutzt, sofern eine Applikation Nachrichten multipler Topics konsumieren soll.

C zeigt eine einfache Topic. Diese Kategorie wird intern behandelt wie Kategorie **A**. Demzufolge können multiple Applikationen diese Topic subskribieren und neu publizierte Nachrichten empfangen. Eine Topic speichert die Daten nicht. Die subskribierten Applikationen müssen folglich zu dem Zeitpunkt des Nachrichtenempfangs aktiv sein. Andernfalls wird die Nachricht ohne Zustellung gelöscht.

D zeigt eine einfache Queue. Diese Kategorie wird vorzugsweise bei einer dedizierten eins zu eins Nachrichtenübermittlungen verwendet. Eine Queue stellt sicher, dass die Nachricht zugestellt wird, indem die Nachrichten in der Queue vorgehalten werden bis diese konsumiert sind. Jede Queue kann von einer internen oder externen Applikation subskribiert werden, es wird eine *PTP* Verbindung verwendet.

Nach Erzeugung einer Service-Instanz des *EM*-Systems, wird ein Service-Key generiert.

Listing 3.5: Darstellung eines Service Key einer *EM*-Instanz

```
1  {
2     "xsappname": "default-aa4134f5-65ec-<Broker>",
3     "management": [
4           {"oa2": {
5              "clientid": "<Appname + Zusatz>",
6              "clientsecret": "<Zufaellig>",
7              "tokenendpoint": "https://<host>.authentication.com/
                  oauth/token",
8              "granttype": "client_credentials"},
9           "uri": "https://<host>-backend.com"}],
10    "messaging": [
11       {
12       "oa2": {"clientid": "<Name>", "clientsecret":"<Passwort>"},
13       "protocol": ["amqp10ws"],
14       "uri": "wss://<host>.cfapps.com/protocols/amqp10ws"}, ... ]
15 }
```

Wie in Lst. 3.5 zu sehen ist, beinhaltet der Service-Key die Zugangsdaten für den in Abschnitt 3.3.1 vorgestellten User-provided Service in Zeile 5 bis 9. Die aufgeführten Zugangsdaten stehen auch unabhängig von dem Service für eine *oa*2 gesteuerte Bearer-Authentifizierung zur Verfügung. Weiterhin sind die Endpunkte für die unterschiedlichen Protokolle sowie dedizierte Zugangsdaten vorhanden. Exemplarisch wird der Aufbau für AMQP in Zeile 16 bis 18 gegeben.

Darüber hinaus werden einer Service-Instanz Ressourcen in Form von *Resource Unit (RU)* zugewiesen [34]. Eine *RU* kann parallel bis zu 3 Endpunkte für Producer und Subscriber zur Verfügung stellen. Es besteht die Möglichkeit einer Service-Instanz multiple *RU*s zuzuweisen, um diese Kapazitäten zu erhöhen.

Um eine möglichst eindeutige Zuordnung der Queues und Topics zu einzelnen Service-Instanzen herzustellen, wird für die Namensgebung ein Muster (Namespace) vorausgesetzt. Der Namespace wird der Benennung der Queue oder Topic vorangestellt und besteht aus einer Kombination des Namen der Cloud-Instanz sowie einer bei der Erstellung der Service-Instanz festgelegten Bezeichnung. Dieses Verfahren wird verwendet, um mögliche Namenskonflikte vorzubeugen.

3.3.1.2 *Subscriber*

Das Vorgehen um Nachrichten einer *EM*-Instanz zu konsumieren, entspricht dem in Abschnitt 3.3.1 geschilderten Vorgehen des Publisher.

Zunächst muss eine Authentifizierung mit der gewünschten *EM*-Instanz durchgeführt werden. Dazu können die gleichen Verfahren wie bei einem Publisher verwendet werden. Abhängig von den Konfigurationen der Queues oder Topics werden No-, Basic- oder eine über *oa*2 gesteuerte Bearer-Authentifizierungen unterstützt. Sofern der Subscriber innerhalb der *BTP* bereitgestellt wird, kann auf einen User-provided Service zurückgegriffen werden.

Bei einer Subskription können Pull oder Push Technologien verwendet werden. Die Möglichen Varianten werden in dem *EM*-System definiert. Dadurch kann die Verarbeitung der Nachrichten je Queue oder Topic individuell konfiguriert werden.

Das *EM*-System Event Mesh beinhaltet für Subscriber einen Timeout. Sofern eine Applikation mehr als 30 Minuten inaktiv ist, wird die aktive Verbindung getrennt. Dies könnte bei der Implementierung folgendermaßen berücksichtigt werden:

Listing 3.6: Wiederaufbau nach Verbindungsabbruch

```
1 // Initialisiere Client
2 const client = new AMQP.Client(env.msgClientOptions(service, [], [
    'myOutB']));
3
4 // Client Optionen
5 client.
6    .on('disconnected', (hadError) => {
7       setTimeout(()=> client.connect(), reconnect_retry_ms);
8    })
9 });
```

In Lst. 3.6 wird bei Verbindungsabbruch, nach *reconnect_retry_ms* Millisekunden, erneut versucht eine Verbindung aufzubauen.

Auf diese Weise kann sichergestellt werden, dass die Applikation stets mit dem *EM*-System verbunden ist. Der gesamte Programmcode für den *MQTT* Subscriber ist in Lst.A.10 vorzufinden.

Es gilt zu beachten, dass *HTTP* subscriberseitig nicht für Publish/Subscribe konzipiert ist. Unter Verwendung eines Push-Mechanismus wie *WSS* und *QoS* (1), sollte ein Webserver verwendet werden, welcher Callbacks unterstützt. Mittels Callbacks kann parallel zu dem Empfang und der Verarbeitung einer Nachricht, eine Empfangsbestätigung gesendet werden. Dadurch wird asynchrone Kommunikation ermöglicht. Die Durchführung in Kapitel 4.1 erfolgt unter Verwendung eines OpenAPI Webserver.

3.3.2 *Zusammenfassung der Architektur*

In diesem Abschnitt wird die vorgeschlagene Architektur anhand der zu Beginn definierten Ziele erläutert.

Die Möglichkeit der direkten Kommunikation sowie Webserver basierte Publish- und Subscriber führen zu einer Entkopplung der Applikationen in der Architektur. Es werden keine direkten Abhängigkeiten unterschiedlicher Applikationen zueinander vorausgesetzt. Die Echtzeitfähigkeit und Skalierbarkeit des Systems wird in Kapitel 4 anhand dreier Testreihen mit unterschiedlichen Subskriptionsvarianten geprüft.

Alle Komponenten der aufgezeigten Architektur sind SAP eigene oder von SAP bereitgestellte Softwarelösungen. Infolgedessen sind keine Kompatibilitätsprobleme zu erwarten. Das *EM*-System ist durch Erhöhung der *RU* skalierbar und kann basierend auf den *RU* Parametern, Nachrichten variabel vielen Applikationen zur Verfügung stellen. Die in Abschnitt 3.3.1 referenzierten Bibliotheken der Java-Script Applikation sind SAP eigene Entwicklungen. Dadurch können Bausteine wie der User-provided Service problemlos in die Applikation integriert und gewartet werden.

Wie in Kapitel 4 gezeigt wird, sind Latenzen im Millisekundenbereich möglich. Unter der Echtzeit Voraussetzung und unter Berücksichtigung der maximalen Nachrichtengröße von 1 MegaByte, scheint eine batchweise Verarbeitung nicht angebracht. Sollte eine Batch-Verarbeitung dennoch mit dem Messaging-System durchgeführt werden, kann ein Aggregator Pattern verwendet werden. Dadurch wird die Akkumulation der Nachrichten von Applikationsseite geregelt [2]. Folglich wird für den Anwendungsfall in Kapitel 4 eine Verarbeitung von Einzelnachrichten betrachtet.

Durch die Entkopplung der Applikationen fungiert die Architektur als Mittelsmann multipler Applikationen. Es sind keine Einschränkungen für die Datenverarbeitung vor oder nach versenden der Nachricht vorhanden.

Einschränkungen sind bei der batchweisen Verarbeitung von Nachrichten zu erkennen. Zwar ermöglicht die Verwendung eines Aggregator Pattern die batchweise Verarbeitung, jedoch ist es fraglich ob diese Umsetzung für große Datenmengen geeignet ist. Des Weiteren sollte unter Vewendung von *HTTP* mit Push-Verfahren, die Kompatibilität der Subscriber berücksichtigt werden. Zusammenfassend lässt sich sagen, dass die vorgestellte Architektur allen Zielen gerecht wird. In Kapitel 4 wird die vorgestellte Architektur mit anwendungsfall spezifischen Ausführungen definiert und evaluiert.

4 UMSETZUNG UND EVALUIERUNG

In diesem Kapitel wird die Umsetzung eines Anwendungsfalles und eine Evaluierung anhand dreier Testreihen präsentiert.

4.1 ANWENDUNGSFALL UND UMSETZUNG

In diesem Abschnitt wird ein Anwedungsfall für die in Kapitel 3.3 erarbeitete Architektur geschildert. Ziel dieser Anwendung ist es, neue Geräteinformationen zur Überprüfung ihrer Validität in Echtzeit zu liefern. Es sollen Duplikate detektiert und eingeordnet werden, sodass spätere Prozesse zur Überarbeitung des Datenbestandes möglich sind. Als Echtzeit wird in diesem Fall, eine Übertragungsgeschwindigkeit von Sekunden bis Minuten definiert.

Zunächst werden mithilfe einer geeigneten Methode, Änderungen an dem aktuellen Datenbestand der *ASSET* Tabelle identifiziert. Diese Änderungen werden mithilfe einer Applikation entgegengenommen und publiziert. Hierzu wird ein Publisher aus Abschnitt 3.3.1 verwendet.

Anschließend werden die publizierten Nachrichten auf eine der in Abschnitt 3.3.1.1 geschilderten Varianten, verarbeitet und einem Subscriber zur Verfügung gestellt. Nach Nachrichteneingang wird anhand der in Kapitel 3.2.4 erarbeiteten Methodik geprüft, welchem Cluster der neue Eintrag zugeordnet wird und ob ein Duplikat angenommen werden kann. Die Zuordnung wird in einer Zieltabelle persistiert und steht für weitere Verarbeitungsprozesse zur Verfügung.

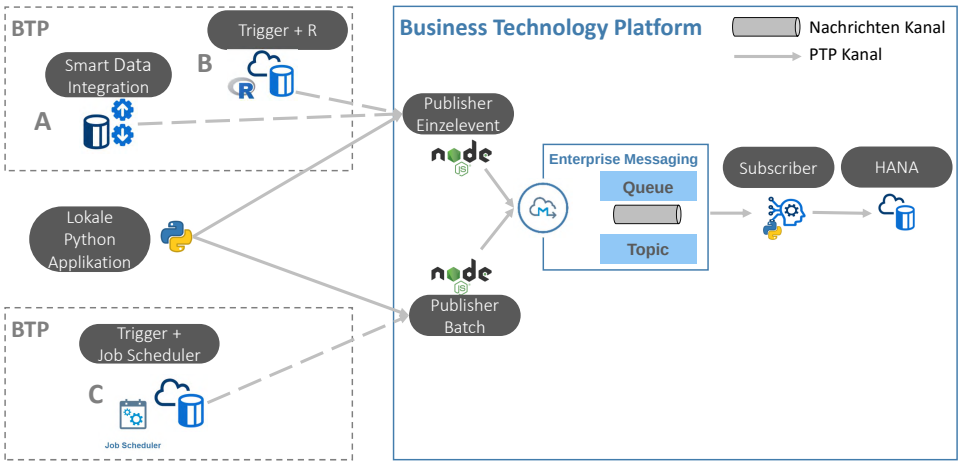


Abbildung 4.1: Basisaufbau der experimentellen Zielarchitektur.

Abb. 4.1 stellt den experimentellen Aufbau des in Abschnitt 4.1 geschilderten Anwendungsfalles dar. Der Aufbau basiert auf der in Abschnitt 3.3.1 vorgestellten Architektur. Auf der linken Seite von Abbildung 4.1 sind mögliche Varianten zur Publikation von Datenbestandsänderungen dargestellt.

Variante **A** symbolisiert einen Eingriff in den Integrationsvorgang. Nach der Projektion der Daten wird mithilfe eines R-Scripts, ein *POST* Aufruf zu dem Publisher durchgeführt. Auf diese Weise können die Daten während der Persistierung, geprüft und verteilt werden. In Abb. 4.2 wird ein solcher Vorgang anhand eines Flowgraph aufgezeigt [59].

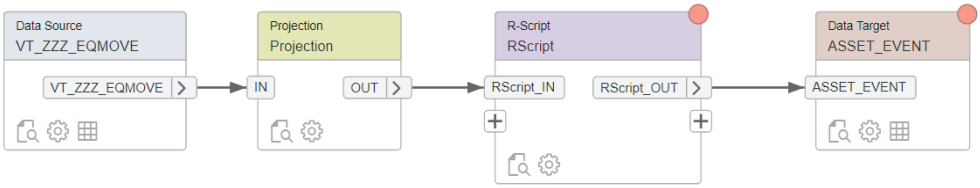


Abbildung 4.2: Flussdiagramm zur Integration einer Remote Datenquelle.

Variante **B** beinhaltet einen Datenbank-Trigger und eine R-Prozedur. Der Trigger wird aufgerufen, sofern eine Datenbankoperation wie *UPDATE* oder *INSERT* durchgeführt wurde. Anschließend kann mithilfe eines R-Scripts eine *POST* Anfrage mit den Änderungen der Daten an den Publisher gesendet werden. Dieses Verfahren wird nach dem Persistieren der Daten in der Datenbank durchgeführt. In Lst. 4.1 ist ein solches Verfahren dargestellt.

Listing 4.1: Datenbanktrigger zur Detektion von Änderungen im Datenbestand und eine R Prozedur zur Veröffentlichung dieser Änderungen

```
CREATE TRIGGER TEST_TRIGGER
AFTER INSERT OR UPDATE ON "ASSET_CENTRAL"."ASSET" REFERENCING
    NEW ROW mynewrow
FOR EACH ROW
LANGUAGE RLANG AS
BEGIN
      library(httr)
      login <- list(email = "login", password = "password",
       submit = mynewrow)
      res <- POST("<Nodejs Host>", body = login, encode = "form",
          verbose())
END;
```

In Variante **C** wird zur Detektion von Änderungen ein Trigger verwendet. Diese Änderungen werden in einer dedizierten Datenbanktabelle gespeichert. Anschließend werden die Daten in einem definierten Intervall mithilfe eines Job-Scheduler Service von einem Publisher gelesen und verarbeitet [60].

Der mittlere Teil von Abb. 4.1 umfasst das Publizieren der Nachrichten. Es wird ein Publisher verwendet, welcher in der *BTP* bereitgestellt wird. Die Funktionsweise des Publisher entspricht der Schilderung aus Abschnitt 3.3.1. Nach Empfang der Nachricht, wird die Nachricht über eine *WSS* an einen Subscriber in der *DI* gesendet.

Abbildung 4.3: Prozessablauf Subscriber in der DI.

In Abb. 4.3 wird der subscriberseitige Ablauf des Datenempfang, der Verarbeitung und Persistierung der Ergebnisse aufgezeigt. Eine in der Umsetzung verwendete Pipeline inklusive der unterschiedlichen Subscriber-Varianten in der *DI* ist in Abb. B.1 dargestellt.

Zunächst wird eine Subskription zu einer Topic oder Queue mithilfe von *WSS* durchgeführt. Die Nachrichten des *EM*-Systems werden von dem Subscriber entgegengenommen. Dazu wird nach Abschnitt 3.3.1.2 ein Nodejs Operator für *AMQP* und *MQTT* oder ein OpenAPI Webserver für *HTTP* verwendet. OpenAPI unterstützt *WSS* und ist damit geeignet für eine asynchrone Kommunikation [30].

Anschließend wird der Inhalt der Nachricht gelesen, die Uhrzeit des Empfangs gespeichert und zusätzliche Informationen zu dem Gerät abgefragt. Dafür wird abhängig von der Testreihe, eine dedizierte Zeitabfrage durchgeführt oder die Zeitsynchronisierung der *BTP* verwendet. In der Beschreibung der Testreihen wird die verwendete Methode geschildert.

Daraufhin werden die Informationen gebündelt, Modelle gelesen und eine Transformation der Daten basierend auf den Distanzmaßen durchgeführt. Basierend auf den Methoden aus Abschnitt 3.2.4 wird geprüft, ob ein automatisch zusammenführbares Duplikat vorliegt.

Zuletzt werden die Ergebnisse in einer dedizierten Zieltabelle persistiert. Aufgrund der Beschaffenheit der vorhandenen Systemlandschaft, wird auf eine Durchführung der Testreihen mit den vorherig beschriebenen Varianten verzichtet. Das versenden einzelner Nachrichten wird durch eine lokale Python Applikation simuliert. Dabei wird zunächst die nötige Testgröße N und der Sendeintervall in Sekunden anhand von *Nachrichten pro Minute (Npm)* und der Testlaufzeit in Minuten berechnet mit

$$N = Npm * Zeitdauer, \qquad Sendeintervall = \frac{Npm}{60}.$$

Anschließend wird die Zielapplikation definiert und die Nachrichten iterativ versendet. Um die definierte *Npm* Anzahl sicherzustellen, wird Multhreading und eine Differenzberechnung der Sende- und Empfangszeit verwendet, siehe Lst. A.11.

4.2 DURCHFÜHRUNG DER TESTREIHEN

In diesem Abschnitt werden die Testreihen zur Prüfung der Echtzeitfähigkeit und Skalierbarkeit des Systems geprüft. Die Prüfung erfolgt mit unterschiedlichen Subskriptionsvarianten. Angelehnt wird die Durchführung der Testreihen an das *one factor at a time* Experimentelle Design [61]. Bei dieser Methode werden Faktoren einzeln variiert, um den Einfluss dieser Faktoren auf die Zielgröße zu bemessen. Die übrigen Faktoren bleiben konstant.

Komponenten und Zielgröße

Zur Durchführung der Testreihen werden zunächst die variierenden Faktoren und die Zielgröße definiert. Eine Zusammenstellung dieser Parameter ist in Tbl. 4.1 dargestellt.

Faktoren	Einstellungen	Zielgröße
Protokolle	HTTP, AMQP, MQTT	Latenz
Subskriptionsvariante	WSS: Topic / Queue	
Nachrichten pro Minute	35 bis 280	
Anzahl Konsumenten	1 bis 3	

Tabelle 4.1: Zusammenstellung der experimentellen Komponenten. Dargestellt werden die Faktoren, zugehörige Ausprägungen und die Zielgröße.

Event Mesh unterstützt *HTTP*, *AMQP* und *MQTT*. Es werden alle Protokolle untersucht, um mögliche Unterschiede in der Performanz zu identifizieren.

Weiterhin werden angelehnt an Abschnitt 3.3.1.1 unterschiedliche Subskriptionsvarianten geprüft. Für jede Variante wird eine *WSS* verwendet.

Um den Parameter *Npm* abzuschätzen, wurden die Modifikationen pro Minute im produktiven System ermittelt, siehe Lst. A.3.

Min	0.25 Quantil	Median	0.75 Quantil	Max
2	11	23	34	136

Tabelle 4.2: Fünf-Punkte-Zusammenfassung des Npm Parameters.

Basierend auf den in Tbl. 4.2 dargelegten Ergebnissen, entspricht der Median im produktiven System 23 Datenbankoperationen wie *INSERT* oder *UPDATE* pro Minute. Die maximale Anzahl an Änderungen liegt bei 136 *Npm*. Aufgrund dieser Beobachtung, wird folgendes Intervall definiert $I_{Npm} = [35, 70, 105, 140, 175, 210, 245, 280]$. Dabei wird das System ausgehend von dem 75% Quantil auf bis zu 200% der Spitzenbelastung geprüft.

Der letzte zu untersuchende Faktor ist die Skalierbarkeit anhand der parallelen Subscriber. Dabei gilt es zu beachten, dass in der vorliegenden Service Konfiguration, eine *RU* verwendet wird. Es stehen parallel drei Endpunkte, Publisher und Subscriber für die Event Mesh Instanz zur Verfügung. Entsprechend wird die Analyse anhand des Intervalls $I_{Subs} = [1, 2, 3]$ parallele Subscriber durchgeführt.

Die Zielgröße ist die Zeitdifferenz zwischen Senden und Empfangen der Nachricht in Millisekunden. Die Zeiten werden von dem Publisher und Subscriber dokumentiert. Diese Zielgröße wird als Latenz bezeichnet und ist definiert als $Latenz \mathrel{\widehat{=}} \Delta T = T_{sub} - T_{pub}$.

Bei allen durchgeführten Testreihen wird eine identische Nachrichtenstruktur in Folgender Form verwendet:

```
{"assetId":<Int32>, "timestamp":<String>, "run_id":<String>}
```

Die Größe des Nachrichteninhalts von Publisher zu *EM*-System entspricht etwa 110 Byte. Sowohl der Publisher als auch die Subscriber werden in der *BTP* bereitgestellt. Die *BTP* wird auf *Amazon Web Services* gehostet.

Hypothesen

Angelehnt an die verfügbaren Kategorien zur Nachrichtenverarbeitung aus Abschnitt 3.3.1.1, werden im Folgenden Hypothesen definiert und anhand dreier Testreihen überprüft. Die Hypothesen werden so gewählt, dass H_1 dem erwarteten Ergebnis entspricht. Es wird eine Irrtumswahrscheinlichkeit von $\alpha = 5\%$ festgelegt.

	Null Hypothese (Ho)	Alternativ Hypothese (H1)
I	Die Wahl des Protokolls hat keinen Einfluss auf die Latenz	Die Wahl des Protokolls hat Einfluss auf die Latenz
II	Eine Topic hat die gleichen Latenzzeiten wie eine Queue.	Eine Topic Subsrkiption hat kleinere Latenzzeiten als eine Queue.
III	Steigende Nachrichten pro Minute haben keinen Einfluss auf die Latenz.	Die Latenz steigt bei zunehmenden Nachrichten pro Minute.
IV	Die Latenz wird durch eine steigende Anzahl an Subscribern nicht beeinflusst.	Die Latenz des Systems nimmt mit einer steigenden Anzahl an Subscribern zu.

Tabelle 4.3: Auflistung der experimentellen Null- und Alternativhypothesen.

Definition der Testreihen

Anhand der Hypothesen aus Tbl. 4.3, werden drei Testreihen definiert.

TESTREIHE 1 (KURZZEIT-TEST): Iterative Steigerung der *Nachrichten pro Minute (Npm)* ausgehend von 35 bis 280 *Npm*, bei identischer Nachrichtengröße. Durchführung der Testreihe für *HTTP*, *AMQP* und *MQTT* bei identischer Testreihenlänge von 5 Minuten. Es wird eine Topic Subskription verwendet und eine Zeitstempelabfrage durchgeführt.

TESTREIHE 2 (LANGZEIT-TEST): Iterative Steigerung der *Npm* ausgehend von 35 bis 280 *Npm*, bei identischer Nachrichtengröße. Durchführung der Testreihe für *AMQP* und *MQTT* bei identischer Testreihenlänge von 30 Minuten. Es wird eine Topic Subskription für *MQTT* und eine Queue Subskription für *AMQP* verwendet und die Zeitsynchronisierung von Amazon Web Services verwendet.

TESTREIHE 3 (SUBSCRIBER-VARIANTEN-TEST): Feste *Npm* bei Spitzenlast (140 *Npm*) und identischer Nachrichtengröße. Durchführung der Testreihe für *AMQP* und *MQTT* bei identischer Testreihenlänge von 30 Minuten. Es wird eine Topic Subscription für *MQTT* und eine Queue Subskription für *AMQP* verwendet. Es wird zwischen 1 und 3 Subscribern variiert und die Zeitsynchronisierung von Amazon Web Services verwendet.

4.2.1 *Testreihe 1: Kurzzeit-Test*

In diesem Abschnitt wird Testreihe 1 - Kurzzeit-Test durchgeführt und die Ergebnisse präsentiert.

4.2.1.1 *Aufbau der Testreihe*

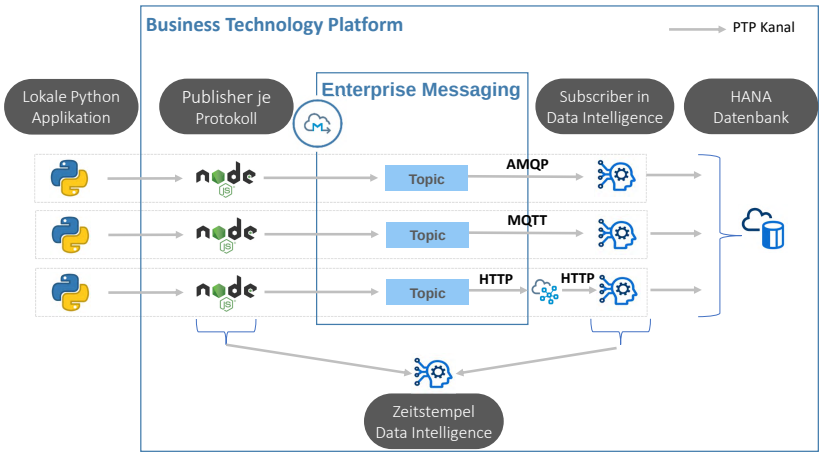


Abbildung 4.4: Experimenteller Aufbau der Testreihe 1: Kurzzeit-Test. Es wird eine dedizierte Zeitstempelabfrage durchgeführt.

In Abb. 4.4 ist der experimentelle Aufbau von Testreihe 1 dargestellt. Jedes Protokoll wird über eine dedizierte, lokale Applikation unabhängig mit Daten versorgt.
Weiterhin wird für jede Variante ein Publisher in der *BTP*, sowie ein Topic Subskription nach Kategorie **C** aus 3.10 in dem *EM*-System verwendet. Der *HTTP* Subscriber benötigt durch Cross-Origin Resource Sharing Restriktionen zwischen dem *EM*-System und der *DI* eine Middleware [56]. Verwendet wird hierfür eine Weiterleitung über einen SAP Service, *Cloud Platform Integration (CPI)*. Dabei wird der Anfrage ein weiterer Header *X-Requested-With* hinzugefügt.

Die Zeitstempelabfrage wird vor dem Publizieren und nach Erhalt der Nachricht durchgeführt.

Der Programmcode der Publisher ist in Lst. A.9 und der Subscriber in Lst. A.10 aufgeführt. Je nach Protokoll wird die entsprechende Messaging Umgebung verwendet. Für *HTTP* wird ein OpenAPI Server verwendet. Die lokale Python Applikation entspricht der Schilderung aus Abschnitt 4.1.

4.2.1.2 *Geprüfte Hypothesen*

Geprüft wird die Hypothese **I** aus Tbl. 4.3. Es werden die Test-Statistiken von Levene, Shapiro-Wilk und Mann-Whintey-U verwendet. Die Prüfung erfolgt mit einem Signifikanzniveau von $\alpha = 0.05$. Die Testreihen wurden unabhängig voneinander durchgeführt und die Latenz ist metrisch skaliert. Latenzen können sortiert und Abstände zueinander berechnet werden.

4.2.1.3 *Ergebnisdarstellung der Testreihe*

In diesem Abschnitt werden die Ergebnisse des Experiments Testreihe 1: Kurzzeit-Test dargelegt. Eingangs wird die Eignung des nicht parametrischen Mann-Whintey-U Tests überprüft. Diese Überprüfung erfolgt mithilfe von graphischen Hilfsmitteln und den Test-Statistiken von Levene und Shapiro-Wilk. Vor Untersuchung der Hypothesen, wird in Tbl. 4.4, die deskriptive Statistik dieser Testreihe gegeben.

	$\bar{L}$	$\tilde{L}$	s	IQR	g	N
AMQP	16002	254.69	35064	3083	2.63	27378
HTTP	16755	473.16	36341	3006	2.64	27362
MQTT	15510	254.54	34125	2322	2.65	27330

Tabelle 4.4: Deskriptive Statistik der Latenz (L) anhand der Komponenten von Hypothese **I**.

*Hypothese **I***

Zunächst wird geprüft, ob bei den Verteilungen der Latenz eine **Normalverteilung** angenommen werden kann.

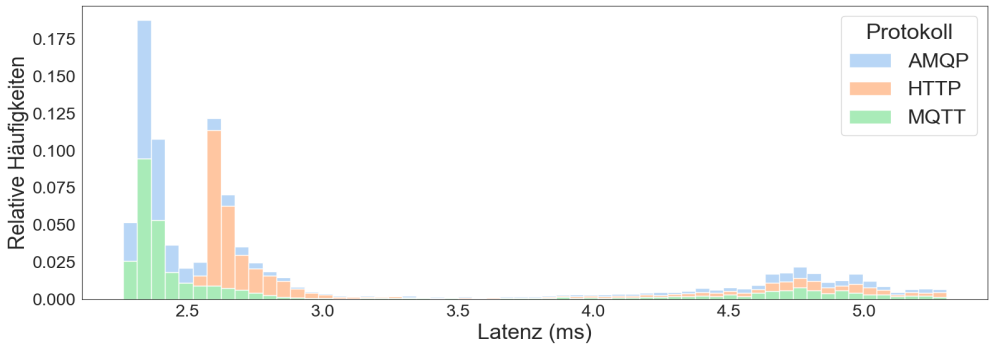


Abbildung 4.5: Histogramm der Latenz bei unterschiedlichen Protokollen. Logarithmisch skaliert mit log^{10}.

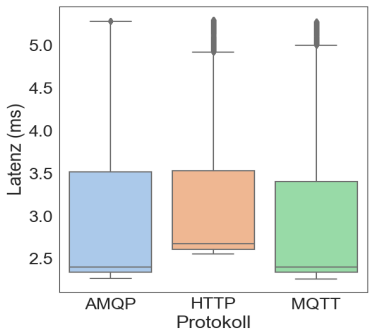
In Abb. 4.5 sind die Verteilungen Latenz unterteilt nach den Protokollen in Form eines Histogramm dargestellt. Die Verteilungen sind rechtsschief ($g > 2.6$) und nicht symmetrisch, vgl. Tbl. 4.4.

Protokoll	W	p
AMQP	0.525	0.000
MQTT	0.524	0.000
HTTP	0.523	0.000

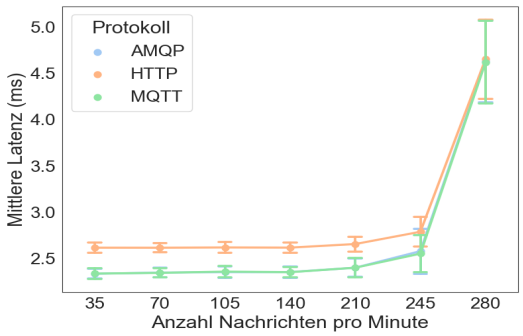
Tabelle 4.5: Darstellung der Testergebnisse des Shapiro-Wilk Tests.

Nach visueller Begutachtung scheint kein Protokoll normalverteilt zu sein. Gestützt wird diese Annahme durch die Ergebnisse des in Tbl. 4.5 dargelegten *Shapiro*-Wilk Tests. Alle drei Protokolle weisen ein stark signifikantes Ergebnis ($p < 0.001$) auf.

Nun wird geprüft, ob zwischen den Latenzverteilungen der Protokolle Varianzgleichheit vorliegt. Es wird die **Homoskedastizität** mithilfe des Levene Tests geprüft.



(a) Boxplot der Latenz pro Protokoll



(b) Entwicklung der Latenz und der Standardabweichung

Abbildung 4.6: Gesamtdarstellung und Entwicklung der Latenz je Protokoll. Logarithmisch skaliert, mit Basis 10 (log_{10}).

Die vorliegende Abb. 4.6 gibt Auskunft über die Verteilung und Entwicklung der Latenz (L) anhand der Protokolle und der *Npm*. Im Vergleich zu $\tilde{L}_{AMQP} = 254$ und $\tilde{L}_{MQTT} = 254$ ist der Median bei *HTTP* mit $\tilde{L}_{HTTP} = 473$ deutlich höher, vgl. Abb. 4.6 und Tbl. 4.4. Nach Abb. 4.6b liegt eine ähnliche Entwicklung der Standardabweichung für alle Protokolle vorzuliegen. Auffällig ist der Unterschied der IQR der Protokolle, vgl. Abb. 4.6a.

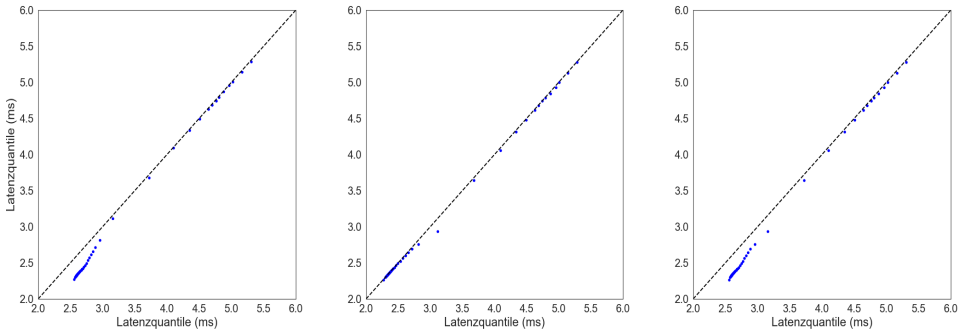
Durch den Versatz von $\tilde{L}_{HTTP} = 473$ und die Unterschiede der IQR zwischen $IQR_{MQTT} = 2322$ und $IQR_{AMQP} = 3083$, scheint keine Gleichheit der Varianzen vorzuliegen.

Variable	W	p
Protokoll	6.172	0.0028

Tabelle 4.6: Darstellung der Testergebnisse des Levene Tests für Hypothese **I**.

Tbl. 4.6 lässt sich entnehmen, dass nach Levene ein signifikantes Ergebnis ($p < 0.05$) vorliegt und die Null Hypothese (H_0) verworfen wird. Nach Berücksichtigung der visuellen Inspektion und des Levene Tests, wird keine Homoskedastizität angenommen.

Zuletzt wird geprüft, ob es sich bei den Varianten um die **gleiche Verteilung** handelt.



(a) HTTP (X), AMQP (Y) (b) AMQP (X), *MQTT*(Y) (c) *HTTP* (X), *MQTT*(Y)

Abbildung 4.7: Quantil-Quantil Plot der Protokoll Kombinationen.
Logarithmisch skaliert, mit Basis (log_{10}).

In Abb. 4.7 sind Quantil-Quantil (QQ)-Plots der Protokoll Kombinationen dargestellt. Es lässt sich an Abb. 4.7a und 4.7c deutlich der Versatz des Median $\tilde{L}_{HTTP}$ bei *HTTP* erkennen. Die Quantile zeigen geringe Abweichungen zu der Winkelhalbierenden und implizieren die gleiche Verteilung für alle Kombinationen. Es wird angenommen, dass gleiche Verteilungen vorliegen.

Basierend auf den dargelegten Ergebnissen wird für Hypothese **I** angenommen, dass keine Normalverteilungen vorliegen, keine Varianzhomogenität gilt und die Latenzverteilungen gleich sind. Der Mann-Whintey-U Test scheint geeignet.

Abschließend werden die Ergebnisse des beidseitigen Mann-Whitney-U Tests erläutert (Signifikanzniveau $\alpha = 0.05$). Unter H_1 gilt: Die Latenzverteilung der Protokolle ist stochastisch gleichwertig.

Protokolle		Teststatistik				
X	Y	U_X	U_Y	z	p	r^2
HTTP	*AMQP*	541	207	-90.4	0.00	0.15
AMQP	*MQTT*	376	371	-1.22	0.22	0.00
HTTP	*MQTT*	545	202	-92.8	0.00	0.16

Tabelle 4.7: Testergebnisse des Mann-Whitney-U Tests für Hypothese **I**. U_X und U_Y skaliert, mit 10^6.

In Tbl. 4.7 werden die Ergebnisse des Mann-Whitney-U Tests zur Bestimmung der stochastischen Gleichwertigkeit zwischen den Protokollen bezüglich der Latenz (L) aufgeführt. Weiterhin ist der Determinationskoeffizient $r^2 = \frac{z^2}{N}$ für die Effektstärke gegeben. Basierend auf den Ergebnissen unterscheidet sich L nach Mann-Whitney für die Kombinationen *AMQP*/*HTTP* und *HTTP*/*MQTT* stark signifikant voneinander ($p < 0.001$), $U_X = 541$, $U_Y = 207$ $z = -90.4$ und $U_X = 545$, $U_Y = 202$ und $z = -92.8$. Beide Kombinationen weisen nach Cohen einen starken Effekt auf, $0.13 < r^2 < 0.26$ [62]. Die Kombination *AMQP*/*MQTT* zeigt kein signifikantes Ergebnis, $U_X = 376$, $U_Y = 371$, $z = -1.22$ und $p = 0.22$.

Die dargelegten Ergebnisse werden durch die Beobachtung des Median gestützt, siehe Tbl. 4.4. Der Versatz von $\tilde{L}_{HTTP}$ zu $\tilde{L}_{AMQP}$ und $\tilde{L}_{MQTT}$ liegt bei etwa 220 ms. Womit die Latenz unter *HTTP* fast doppelt so hoch ist wie unter *AMQP* und *MQTT*.

Die Null-Hypothese (H_0) von Hypothese **I** kann folglich für die Kombinationen *HTTP*/*AMQP* und *HTTP*/*MQTT* verworfen werden. Es liegt ein signifikanter Unterschied zwischen *HTTP* und *AMQP* sowie *HTTP* und *MQTT* vor. Es kann kein signifikanter Einfluss auf die Latenz zwischen den Protokollen *AMQP* und *MQTT* identifiziert werden.

4.2.2 *Testreihe 2: Langzeit-Test*

In diesem Abschnitt wird Testreihe 2-Langzeit-Test durchgeführt und die Ergebnisse dargelegt.

4.2.2.1 *Aufbau der Testreihe*

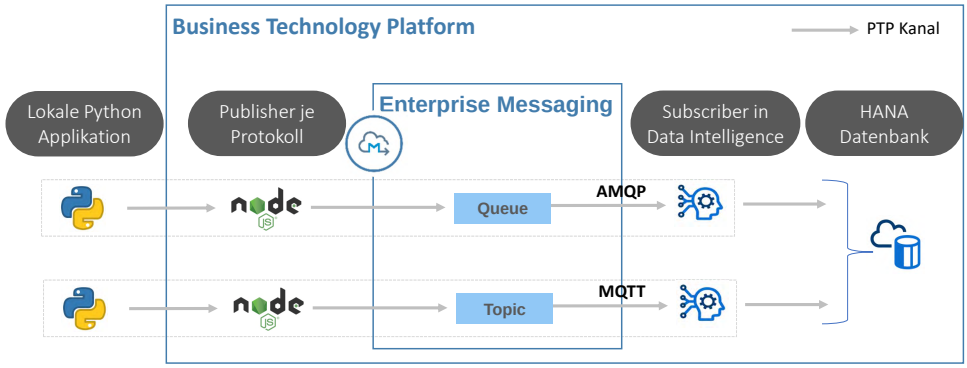


Abbildung 4.8: Experimenteller Aufbau der Testreihe 2: Langzeit-Test. Es wird die Zeitsynchronisierung der BTP verwendet.

In Abb. 4.8 ist der Experimentelle Aufbau von Testreihe 2 dargelegt. Für *AMQP* wird eine *Queue*- (Kategorie **D**) und für *MQTT* eine *Topic*-Subskription (Kategorie **C**) verwendet, vgl. Abschnitt 3.10. Alle Applikationen werden in der *BTP* bereitgestellt und von *Amazon Web Services* gehostet. Es wird die Zeitsynchronisierung von *Amazon Web Services* verwendet. *AMQP* und *MQTT* kann angelehnt an die Ergebnisse aus Abschnitt 4.2.1 (Testreihe 1) für den Vergleich verwendet werden.

In diesem Testlauf werden die entsprechenden Messaging Umgebungen aus Lst. A.9 und A.10 verwendet. Es wird der gleiche Mechanismus für die lokale Python Applikation genutzt wie in Testreihe 1 (siehe 4.2.1).

4.2.2.2 *Geprüfte Hypothese*

Geprüft wird Hypothese **II** und **III** aus Tbl. 4.3. Es werden die Test-Statistiken von Levene, Shapiro-Wilk und Mann-Whintey-U verwendet. Die Prüfung erfolgt mit einem Signifikanzniveau von $\alpha = 0.05$. Die Testreihen wurden unabhängig voneinander durchgeführt und die Latenz ist metrisch skaliert.

4.2.2.3 *Ergebnisdarstellung*

In diesem Abschnitt werden die Ergebnisse des Experiments Testreihe 2: Langzeit-Test dargelegt. Die Aufbereitung der Ergebnisse erfolgt pro untersuchter Hypothese.

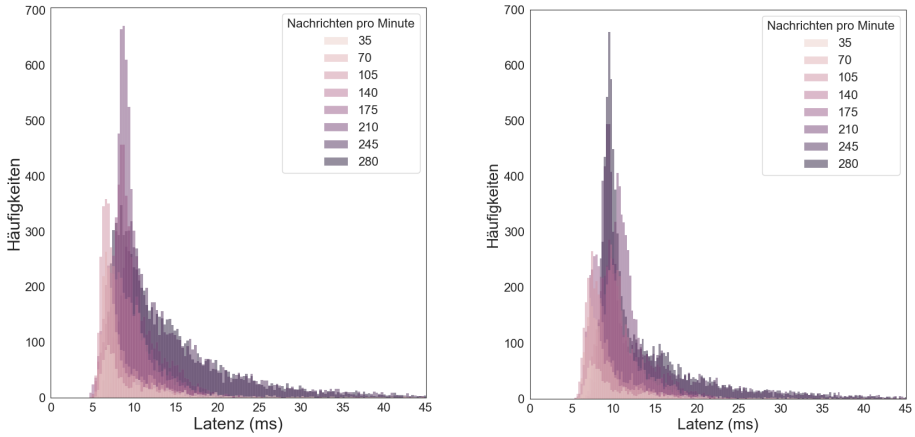
Dabei wird zunächst die Eignung des nicht parametrischen Mann-Whitney-U Tests geprüft. Diese Überprüfung erfolgt mithilfe von graphischen Hilfsmitteln und den Test-Statistiken von Levene und Shapiro-Wilk. Vor der Untersuchung der Hypothesen, wird in Tbl. 4.8 eine Übersicht über die deskriptive Statistik dieser Testreihe gegeben.

	Latenz (L_Q), Queue						Latenz (L_T), Topic					
Npm	$\bar{L}_Q$	$\tilde{L}_Q$	s_Q	IQR_Q	g_Q	N_Q	$\bar{L}_T$	$\tilde{L}_T$	s_T	IQR_T	g_T	N_T
Gesamt	12.4	9.3	10.7	5.2	11.2	37944	12.8	10.2	11.2	4.2	13.9	37795
35	12.5	8.6	13.2	6.5	7.9	1050	12.5	8.9	11.7	6.1	9.1	1050
70	8.5	7.3	8.7	2.2	24.5	2100	9.2	8.1	4.7	2.4	12.4	2100
105	8.1	7.1	3.5	2.2	14.5	3150	9.2	8.2	6.0	2.6	30.8	3150
140	9.9	8.9	7.4	3.5	20.3	4199	11.2	10.3	6.6	2.8	26.0	4200
175	9.8	8.9	6.8	2.5	24.1	5250	10.3	9.6	5.5	3.2	22.6	5250
210	10.5	9.3	5.4	2.5	25.3	6299	11.5	10.9	3.5	2.8	17.8	6300
245	16.7	12.9	13.7	8.6	9.1	7350	16.0	11.1	16.2	7.5	9.5	7350
280	15.2	11.7	13.2	7.9	8.1	8392	15.3	10.9	14.9	6.7	11.7	8395

Tabelle 4.8: Übersichtstabelle Deskriptive Statistik, Testreihe 2: Langzeit-Test.

*Hypothese **II***

Zunächst wird geprüft, ob bei den Latenzverteilungen eine **Normalverteilung** angenommen werden kann.



(a) Verteilung der Queue Subskription

(b) Verteilung der Topic Subskription

Abbildung 4.9: Histogramme der Latenz bei unterschiedlichen Subskriptionsvarianten unterteilt nach Nachrichten pro Minute.

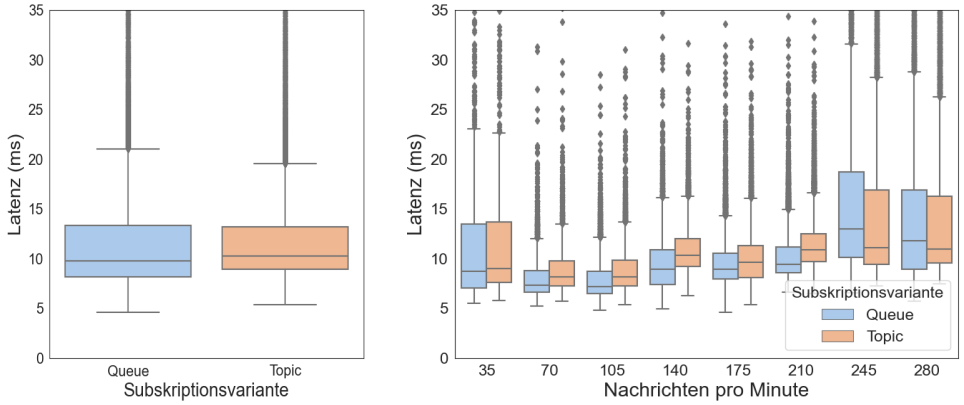
In Abb. 4.9 sind die Latenzverteilungen nach *Npm* unterteilt dargestellt. Aus Abb. 4.9a und 4.9b geht hervor, dass beide Subskriptionsvarianten rechtsschief ($g_Q = 11$ und $g_T = 14$) und asymmetrisch sind, vgl. Tbl. 4.8.

		Nachrichten pro Minute								
Variante	Statistik	Gesamt	35	70	105	140	175	210	245	280
Queue	W	0.41	0.42	0.15	0.50	0.26	0.25	0.28	0.49	0.48
	p	0.00	0.00	0.00	0.00	0.00	0.0	0.00	0.00	0.00
Topic	W	0.34	0.42	0.400	0.26	0.25	0.35	0.59	0.40	0.37
	p	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Tabelle 4.9: Darstellung der Testergebnisse des Shapiro-Wilk Tests für Hypothese **II**.

Anhand der Testergebnisse nach Shapiro-Wilk in Tbl. 4.9, liegen für alle *Npm* signifikante Ergebnisse vor ($p < 0.05$). Demzufolge und unter Berücksichtigung der visuellen Inspektion, wird keine Normalverteilung angenommen.

Nun wird geprüft, ob die Verteilungen der Latenz (L) zwischen den Subspriktionsvarianten und *Npm* die gleiche Varianz aufweisen. Es wird die **Homoskedastizität** mithilfe des Levene-Tests geprüft.



(a) Boxplot der Latenz pro Subskriptionsvariante

(b) Entwicklung der Latenz nach Nachrichten pro Minute

Abbildung 4.10: Verteilung und Entwicklung der Latenz bei unterschiedlichen unterschiedlichen Subskriptionsvarianten.

In Abb. 4.10 wird mithilfe von Boxplots die Gesamtverteilungen der Subskriptionsvarianten gegenübergestellt. Weiterhin werden die Varianten nach *Npm* unterteilt und vergleichbar dargestellt.

Nach visueller Inspektion von Abb. 4.10a, weisen beide Varianten einen vergleichbaren IQR auf. Bei $IQR_Q = 5.2$ und $IQR_T = 4.2$ handelt es sich, um eine Abweichung von 1 ms. Die Entwicklung des IQR ist nach Abb. 4.10b ähnlich für beide Subskriptionsvarianten. Schwankungen der Standardabweichung s_T und s_Q zwischen den *Npm* Gruppen sind bei beiden Varianten ausgeprägt in den *Npm* Randgruppen $35, 245, 280$, vgl. Tbl. 4.8. Visuell ist kein Unterschied der Varianzen erkennbar.

		Nachrichten pro Minute								
Variante	Statistik	Gesamt	35	70	105	140	175	210	245	280
Queue / Topic	W	22.00	1.48	0.04	5.41	6.14	0.99	0.21	0.14	4.84
	p	0.00	0.22	0.83	0.02	0.013	0.32	0.65	0.71	0.03

Tabelle 4.10: Darstellung der Testergebnisse des Levene Tests für Hypothese **II**.

Tabelle 4.10 lässt sich entnehmen, dass keine Gleichheit der Varianzen für alle Gruppen angenommen werden kann ($p < 0.05$). Für 35, 70, 175, 210, 245 *Npm* liegt kein signifikantes Ergebnis vor. Die Testergebnisse der gesamten Verteilungen sind stark signifikant mit $p < 0.001$. Es wird Varianzhomogenität angenommen.

Zuletzt wird geprüft, ob es sich bei den Varianten um die **gleiche Verteilung** handelt.

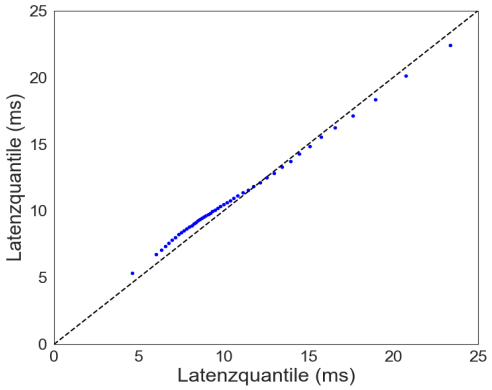


Abbildung 4.11: QQ-Plot der Subskriptionsvarianten.

In Abb. 4.11 wird ein Quantil-Quantil-Plot der Subskriptionsvarianten Queue und Topic dargestellt. Aufgrund der geringen Abweichungen der Quantile von der Winkelhalbierenden, scheint es sich bei beiden Varianten um die gleiche Verteilung zu handel. Diese Annahme wird gestützt aufgrund der ähnlichen Mediane und Standardabweichungen $\tilde{L}_Q = 9.73$, $s_Q = 10.66$ und $\tilde{L}_T = 10.28$, $s_T = 11.25$ aus Tbl. 4.8.

Basierend auf den dargelegten Ergebnissen wird für Hypothese **II** angenommen, die Verteilungen sind nicht Normalverteilt, es gilt Varianzhomogenität und beide Varianten haben die gleiche Verteilung. Der Mann-Whitney-U Test scheint geeignet.

Abschließend werden die Ergebnisse des einseitigen Mann-Whitney-U Test betrachtet (Signifikanzniveau $\alpha = 0.025$). Unter H_1 gilt, die Latenzverteilung einer Topic ist stochastisch kleiner als bei einer Queue. Um eine möglichst Aussagekräftige Vergleichbarkeit zu erhalten, wird der Mann-Whitney-U Test pro *Npm* Gruppe durchgeführt.

Npm		Teststatistik				
X (Queue)	Y (Topic)	U_X	U_Y	z	p	r^2
Gesamt	Gesamt	63702	79706	-26.6	1.0	0.009
35	35	57	69	-3.8	0.994	0.007
70	70	156	284	-16.2	1.0	0.063
105	105	338	653	-21.8	1.0	0.076
140	140	575	1188	-27.6	1.0	0.091
175	175	1196	1559	-11.7	1.0	0.013
210	210	1296	2671	-33.7	1.0	0.09
245	245	3071	2331	-14.4	0.0	0.014
280	280	3374	3671	-4.7	1.0	0.001

Tabelle 4.11: Testergebnisse des Mann-Whitney-U Tests für Hypothese **II**. U_X und U_Y skaliert, mit 10^4.

Tbl. 4.11 zeigt die Ergebnisse des Mann-Whitney-U Tests zur Prüfung der Annahme, ob die Latenzverteilung einer Topic- stochastisch kleiner als einer Queue-Subskription ist. Unter Betrachtung des gesamten Datensatzes, ist die Latenzverteilung einer Queue stochastisch kleiner oder gleichwertig wie bei einer Topic, $U_x = 637$, $U_y = 797$, $z = -26.6$, $p > 0.025$ und $r^2 = 0.009$. Nur bei 245 *Npm* liegt ein stark signifikantes Ergebnis vor ($p < 0.001$). Für die übrigen *Npm* Gruppen gilt entgegen der Erwartung, dass die Latenzverteilung einer Queue kleiner oder gleichwertig mit einer Topic ist ($p > 0.05$).

Die aufgeführten Ergebnisse können anhand der Mediane L_Q, L_T zusätzlich gezeigt werden. Es gilt $\tilde{L}_Q < \tilde{L}_T$ für jede *Npm* Gruppe, außer 245 und 280. Dies impliziert, dass eine Queue sogar kleinere Latenzzeiten als eine Topic aufweist.

Die Null-Hypothese (H_0) von **II** kann nicht verworfen werden. Latenzzeiten einer Queue- sind stochastisch mindestens gleichwertig mit einer Topic-Subskription. Durch die Mediane wird von kleineren Latenzzeiten für eine Queue ausgegangen.

*Hypothese **III***

Auf eine Prüfung der **Normalverteilung** wird verzichtet. Die Untersuchung aus Tbl. 4.9 hat weiterhin bestand.

Nun wird geprüft, ob die Verteilung der Latenz (L) zwischen den *Nachrichten pro Minute (Npm)*-Gruppen die **gleiche Varianz** aufweisen. Die Untersuchung wird für beide Subskriptionsvarianten mithilfe des Levene Tests durchgeführt.

		Npm-Kombinationen						
Variante	Statistik	35/70	70/105	105/140	140/175	175/210	210/245	245/280
Queue	W	87.4	2.63	35.66	7.64	3.31	715.30	3.1
	p	0.00	0.11	0.00	0.006	0.07	0.00	0.08
Topic	W	106.89	0.09	1.45	0.56	13.15	533	13.8
	p	0.00	0.76	0.23	0.46	0.00	0.00	0.00

Tabelle 4.12: Testergebnisse des Levene Tests für Hypothese **III**.

In Tbl. 4.12 werden die Ergebnisse des Levene Tests auf Gleichheit der Varianzen dargelegt. Basierend auf den Ergebnissen liegt bei beiden Subskriptionsvarianten ein signifikantes Ergebnis ($p < 0.05$) für 35/70 und 210/245 vor. Weiterhin sind die Gruppen 105/140 und 140/175 der Queue signifikant mit $p < 0.05$. Die Topic Subskription ist signifikant für 175/210 und 245/280. Die Verteilung für die einzelnen *Npm* Gruppen in Abb. 4.10b ist uneinheitlich in den Randgruppen 35, 245, 280. Änderungen in den restlichen Gruppen sind relativ konstant. Es wird eine Gleichheit der Varianzen angenommen.

Zuletzt wird geprüft, ob **gleiche Verteilungen** zwischen den Gruppen vorliegen. Hierbei wird aufgrund der Ergebnisse aus Abb. 4.11, auf eine Darstellung beider Subskriptionsvarianten verzichtet. Es wird angenommen, dass die Verteilung beider Varianten gleich ist.

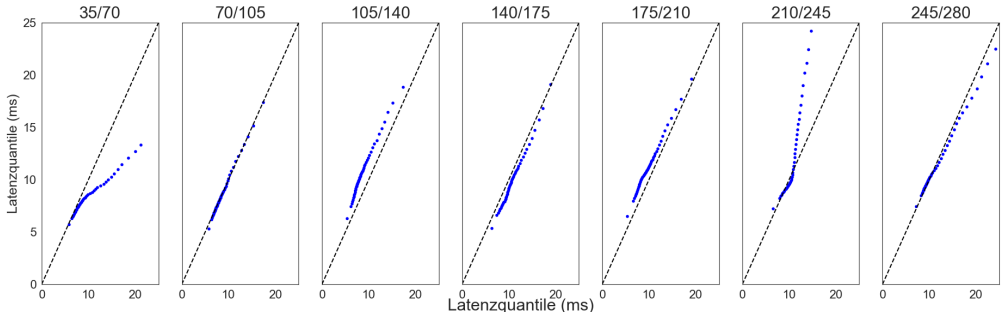


Abbildung 4.12: Quantil-Quantil Plot der Topic. Die Unterteilung der Graphen erfolgt nach den in Tabelle 4.12 dargelegten Gruppen.

In Abb. 4.12 sind die QQ-Plots der einzelnen *Npm* Gruppen dargestellt. Die Graphen implizieren eine gleiche Verteilung der Gruppen für die Kombinationen 70/105, 105/140, 140/175, 175/210 und 245/280. Ausgeprägte Abweichungen von der Winkelhalbierenden liegen bei den Gruppen 35/70 und 210/245 vor. Es wird die gleiche Verteilung innerhalb der Gruppen angenommen.

Basierend auf den dargelegten Ergebnissen wird für Hypothese **III** angenommen, dass die Verteilungen nicht normalverteilt sind, Varianzhomogenität gilt und die Gruppen die gleiche Verteilung haben. Der Mann-Whitney-U Test scheint geeignet.

Abschließend werden die Ergebnisse des einseitigen Mann-Whitney-U Test betrachtet (Signifikanzniveau $\alpha = 0.025$). Unter H_1 gilt, die Latenzverteilung aufsteigender *Npm* Gruppen sind stochastisch größer für schrittweise Gruppen.

		Npm Kombinationen (X / Y)						
		35/70	70/105	105/140	140/175	175/210	210/245	245/280
Queue	U_X	167	349	414	1068	1306	1085	3568
	U_Y	85	312	908	1135	2000	3544	2599
	z	-15.40	-3.40	-27.44	-2.56	-19.46	-53.57	-17.05
	p	1.00	0.99	0.00	0.005	0.00	0.00	1.00
	r^2	0.07	0.002	0.10	0.00	0.03	0.21	0.02
Topic	U_X	136	332	319	1349	1080	2055	3032
	U_Y	84	329	1003	855	2227	2575	3138
	z	-10.78	-0.28	-37.94	-18.75	-32.14	-11.34	-1.87
	p	1.00	0.61	0.00	1.00	0.00	0.00	0.03
	r^2	0.04	0.00	0.20	0.04	0.09	0.01	0.00

Tabelle 4.13: Testergebnisse des Mann-Whitney-U Tests für Hypothese **III**. U_X und U_Y skaliert, mit 10^4.

Tbl. 4.13 zeigt die Ergebnisse des Mann-Whitney-U Tests zur Überprüfung steigender Latenzzeiten für aufsteigende *Npm*. Basierend auf den Ergebnissen liegt keine signifikante Steigerung der Latenz (L) in den *Npm* Randgruppen 35/70, 70/105 und 245/280 für die *Queue* vor ($p > 0.025$). Gruppe 35/70 weist eine schwache Effektstärke mit $r^2 > 0.07$ auf [62]. Es gilt zu beachten, dass die Gruppe 35/70 die geringste Stichprobengröße mit $N = 1050$ aufweist, siehe Tbl. 4.8. Die übrigen Gruppen der Queue zeigen ein signifikantes Ergebnis für 140/175 mit $p < 0.025$ und stark signifikante Ergebnisse für 105/140, 175/210 und 210/245 ($p < 0.001$).

Anhand der deskriptiven Statisk kann dieses Ergebnis nachvollzogen werden, vgl. Tbl. 4.8.

Der Median der Queue $\tilde{L}_Q$ zeigt in der *Npm* Gruppe 35/70 einen Versatz des von $\tilde{L}_{35/70} = -4$ ms. Bei 70/105 zeigt sowohl der QQ-Plot als auch der Median, dass es sich um die gleiche Verteilung handelt. Ähnliches Verhalten liegt für 245/280 vor. Bei den restlichen Gruppen steigt der Median im Bereich von $\tilde{L} \leq 2$ ms. stetig an.

Die Latenz bei einer Topic weist ähnliches Verhalten auf. Es liegt ein Abfall des Median für die Gruppen 35/70, 140/175 und 245/280 vor. Sonstige Änderungen sind weitestgehend konstant mit Anstiegen der Latenz im Median von $\tilde{L}_T \leq 1$ ms.

Basierend auf diesen Beobachtungen, kann kein eindeutiges Ergebnis für Hypothese **III** abgeleitet werden. Steigende *Npm* scheinen einen Einfluss auf die Performanz des Systems zu haben, dieser setzt sich allerdings nicht ausnahmslos fort. Auch Abb. 4.10b und Tbl. 4.8 implizieren dieses Verhalten mit steigendem IQR, Median und Standardabweichung außerhalb der Randgruppen.

4.2.3 *Testreihe 3: Subscriber-Varianten-Test*

In diesem Abschnitt wird Testreihe 3: Subscriber-Varianten-Test durchgeführt und die Ergebnisse dargelegt.

4.2.3.1 *Aufbau der Testreihe*

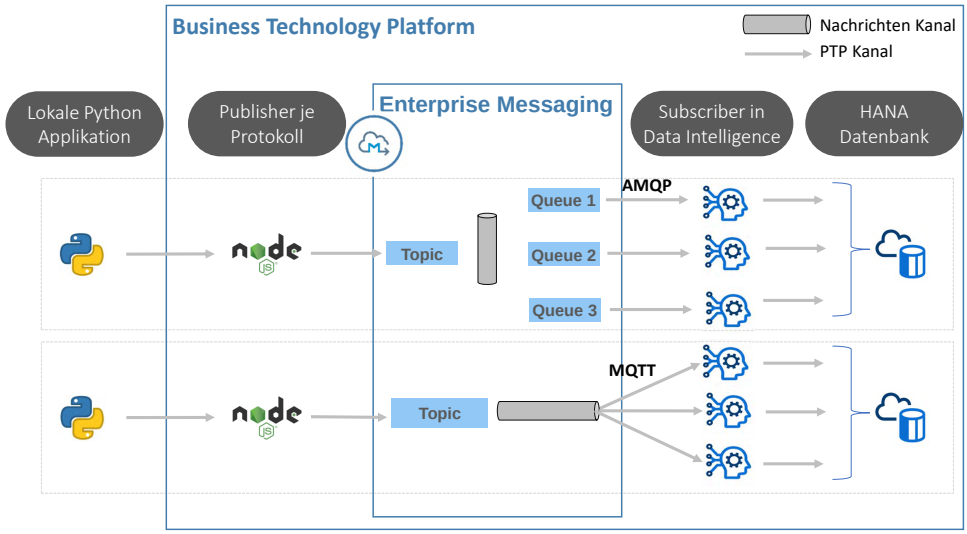


Abbildung 4.13: Experimenteller Aufbau der Testreihe 3: Subscriber-Varianten-Test. Es wird die Zeitsynchronisierung der BTP verwendet.

Die vorliegende Abb. 4.8 zeigt den experimentellen Aufbau von Testreihe 3.

Es wird eine Topic- und eine Queue-Subskription nach Kategorie **B**, **A** aus Abschnitt 3.10 verwendet. Die Anzahl der Subcriber wird von 1-3 variiert basierend auf den *RU* Parametern aus Abschnitt 3.3.1.1. Alle Applikationen werden in der *BTP* bereitgestellt und von *Amazon Web Services* gehostet. Es wird die Zeitsynchronisierung von *Amazon Web Services* verwendet.

In diesem Testlauf werden drei Subscriber nach Lst. A.10 verwendet. Der restliche Aufbau entspricht Testreihe 2 aus Abschnitt 4.2.2.

4.2.3.2 *Geprüfte Hypothese*

Geprüft wird Hypothese **IV** aus Tbl. 4.3. Es werden die Test-Statistiken von Levene, Shapiro-Wilk und Mann-Whintey-U verwendet. Die Prüfung erfolgt mit dem Signifikanzniveau $\alpha = 0.05$. Die Testreihen wurden unabhängig voneinander durchgeführt und die Latenz ist metrisch skaliert.

4.2.3.3 *Ergebnisdarstellung*

In diesem Abschnitt werden die Ergebnisse des Experiments Testreihe 3-Subscriber-Varianten Test dargelegt. Zunächst wird die Eignung des nicht parametrischen Mann-Whitney-U Tests geprüft. Diese Überprüfung erfolgt mithilfe von graphischen Hilfsmitteln und den Test-Statistiken von Levene und Shapiro-Wilk. Vor der Untersuchung der Hypothesen, wird in Tbl. 4.14 eine Übersicht über die deskriptive Statistik dieser Testreihe gegeben.

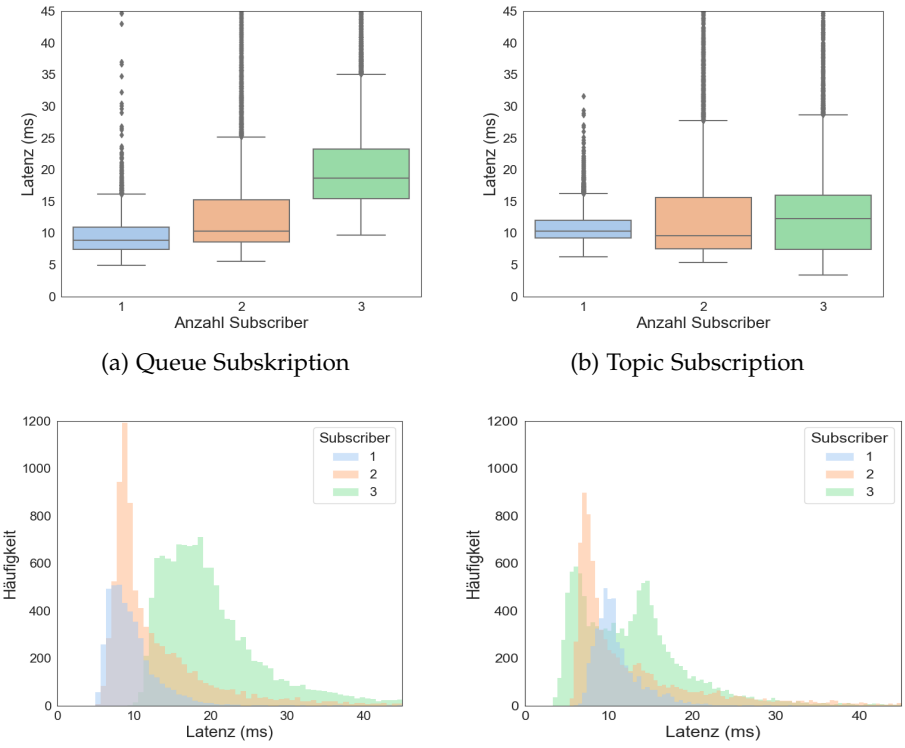
Variante	Anzahl Subscriber	Latenz (L) $\bar{L}$	$\tilde{L}$	s	IQR	g	N
Queue	1	9.91	8.91	7.48	3.50	20.31	4199
	2	14.28	10.31	28.47	6.63	54.40	8400
	3	21.41	18.71	13.04	7.85	8.81	12597
Topic	1	11.23	10.33	6.62	2.81	26.08	4200
	2	15.49	9.59	38.59	8.09	31.63	8400
	3	13.76	12.31	15.13	8.47	21.87	12582

Tabelle 4.14: Deskriptive Statistik, Testreihe 3: Subscriber-Varianten Test.

*Hypothese **IV***

Auf eine Prüfung der **Normalverteilung** wird verzichtet. Die Untersuchung aus Tbl. 4.9 hat weiterhin bestand.

Nun wird geprüft, ob die Latenzverteilungen zwischen der Anzahl an Subscribern die **gleiche Varianz** aufweisen. Die Untersuchung wird für beide Subskriptionsvarianten mithilfe des Levene Tests durchgeführt.



(a) Queue Subskription (b) Topic Subscription

(c) Queue Subskription (d) Topic Subscription

Abbildung 4.14: Gesamtdarstellung der Latenz anhand der Anzahl an Subscribern unterteilt nach den Subskriptionsvarianten.

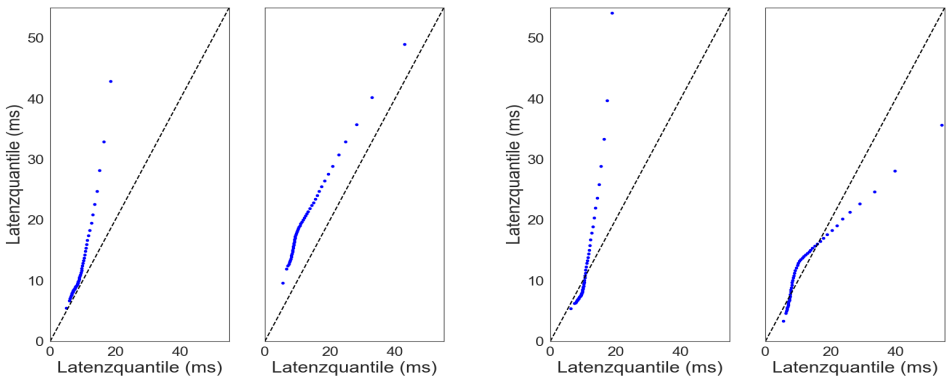
Abb. 4.14 liefert Informationen zu der Latenzverteilung, bei steigender Anzahl Subscriber unterteilt nach der Anzahl. Für die Queue steigt der Inter-Quartilsabstand IQR_q mit steigenden Subskriptionen, siehe Abb. 4.14a. Gleiches Verhalten zeigt die Topic, siehe Abb. 4.14b. Der Anstieg des IQR scheint ähnlich bei der Queue zwischen 2 und 3 Subscribern. Nach den Histogrammen aus Abb. 4.14c und 4.14d zu urteilen, nimmt die Breite der Verteilung sowie die Anzahl an Ausreißern für steigende Subscriber zu. Nach visueller Inspektion kann keine eindeutige Aussage über die Varianzen getroffen werden.

		Kombinationen	
Variante	Statistik	1 / 2	2 / 3
Queue	W	52.76	1.37
	p	0.00	0.24
Topic	W	93.39	29.30
	p	0.00	0.00

Tabelle 4.15: Darstellung der Testergebnisse des Levene Tests für Hypothese **IV**.

Basierend auf den Ergebnissen aus Tbl. 4.15, liegt für die Queue bei 2/3 Subscribern kein signifikantes Ergebnis vor ($p > 0.05$). Alle anderen Varianten sind stark signifikant mit $p < 0.001$. Dieses Ergebnis wird gestützt bei näherer Betrachtung der Stichproben-Standardabweichung s^2 aus Tbl. 4.14. s^2 schwankt bei beiden Varianten stark. Es kann keine Varianzhomogenität angenommen werden.

Zuletzt wird geprüft, ob es sich bei den Varianten um die **gleiche Verteilung** handelt.



(a) Queue Subskription, 1/2 Subscriber, links 2/3 Subscriber, rechts

(b) Topic Subskription, 1/2 Subscriber, links 2/3 Subscriber, rechts

Abbildung 4.15: Quantil-Quantil Plot der Subskriptionsvarianten unterteilt nach Anzahl an Subscribern.

In Abb. 4.15 werden die QQ-Plots der Varianten 1/2 und 2/3 Subscriber dargestellt. Abb. 4.15a zeigt die Varianten für eine Queue und Abb. 4.15b für eine Topic Subskription. Variante 1/2 beider Subskriptionsvarianten zeigen eine starke linksschiefe. Dieses Verhalten ist auch anhand der Schiefe g_Q und g_T ersichtlich, siehe Tbl. 4.14. Die Bereiche der höchsten Dichte liegen bei beiden Varianten bei etwa $\tilde{L} \leq 12$ Millisekunden. Bei 2 Subscribern könnte eine Verschiebung mit ähnlicher Form vorliegen, siehe Abb. 4.14c und 4.14d.

Variante 2/3 der Queue Subskription zeigt einen Versatz der Quantile, auch ersichtlich anhand des Median $\tilde{L}_Q$ aus Tbl. 4.14. Neben dem Versatz verläuft der QQ-Plot nahezu parallel zur Winkelhalbierenden. Variante 2/3 der Topic Subskription weist auf eine Bi-Modale Verteilung von 3 Subscriber hin, siehe Abb. 4.15b und 4.14d. Es kann keine gleiche Verteilung angenommen.

Basierend auf den dargelegten Ergebnissen wird für Hypothese **IV** angenommen, dass die Verteilungen nicht normalverteilt sind, keine Varianzhomogenität gilt und beide Varianten nicht die gleiche Verteilung haben. Der Mann-Whitney-U Test wird verwendet. Die aufgeführten Abweichungen werden bei der Interpretation berücksichtigt.

Abschließend werden die Ergebnisse des einseitigen Mann-Whitney-U Test betrachtet (Signifikanzniveau $\alpha = 0.025$). Unter H_1 gilt, die Latenzverteilung ist für weniger Subscriber stochastisch kleiner.

Subscriber (n)		Teststatistik, Queue					Teststatistik, Topic				
X	Y	U_X	U_Y	z	p	r^2	U_X	U_Y	z	p	r^2
1	2	1162	2364	-31.24	0.0	0.08	1927	1600	-8.48	1.0	0.01
2	3	1837	8743	-80.25	0.0	0.31	5056	5512	-5.30	0.0	0.01

Tabelle 4.16: Testergebnisse des Mann-Whitney-U Tests für Hypothese **IV**. U_X und U_Y skaliert, mit 10^6. n entspricht der Anzahl an Subscribern.

Tbl. 4.16 zeigt die Ergebnisse des Mann-Whitney-U Tests, zur Bestimmung eines signifikanten Anstieges der Latenz bei steigender Anzahl an Subscribern. Ein stark signifikanter Unterschied der Latenz liegt für die Queue von 1/2 Subscriber vor $U_X = 1162$, $U_Y = 2364$, $z = -31.24$, $p < 0.001$ und $r^2 = 0.08$. sowie bei 2/3 Subscribern mit $U_X = 1837$, $U_Y = 8743$, $z = -80.25$, $p < 0.001$ und $r^2 = 0.31$. Die Effektstärke liegt respektive im schwachen und starken Bereich [62].

Gruppe 1/2 der Topic zeigt keinen signifikanten Unterschied $U_X = 1927$, $U_Y = 1600$, $z = -8.5$, $p > 0.025$. Hierbei gilt es zu beachten, dass zwar der Median bei einem kleiner als bei zwei Subscribern ist ($\tilde{L}_1 = 10.33 > \tilde{L}_2 = 9.59$), aber die Standardabweichung s von 6.7 auf 38.6 springt, vgl. Tbl. 4.14.

Dies könnte auf die unterschiedlichen Stichprobengrößen N der Subscriber zurückzuführen sein, vgl. Tbl. 4.14. Ein signifikanter Unterschied konnte zwischen $2/3$ Subscribern gezeigt werden mit $U_X = 5056$, $U_Y = 5512$, $z = -5.30$ und $p < 0.025$. Die Effektstärke liegt unterhalb des schwachen Bereich $r^2 < 0.07$.

Basierend auf den Ergebnissen, konnte ein statistisch signifikanter Unterschied der Latenz bei steigender Anzahl Subscriber für die Queue gezeigt werden. Die Null-Hypothese (H_0) von Hypothese **IV** kann für eine Queue verworfen werden. Bei einer Topic liegen keine eindeutigen Ergebnisse vor, ein Einfluss auf die Latenz scheint vorhanden.

Weiterhin wird geprüft, ob die Ergebnisse aus Hypothese **II** auch bei einer steigenden Anzahl an Subscribern verworfen werden kann. Unter H_1 gilt, die Latenz für aufsteigende Subscriber einer Topic ist stochastisch kleiner als bei einer Queue (Signifikanzniveau $\alpha = 0.025$).

Subscriber (n)		Queue (X), Topic (Y)				
X	Y	U_X	U_Y	z	p	r^2
1	1	5	11	-27.59	1.0	0.09
2	2	39	30	-14.49	0.00	0.01
3	3	127	30	-84.33	0.00	0.28

Tabelle 4.17: Testergebnisse des Mann-Whitney-U Tests für Hypothese **IV** basierend auf den Ergebnissen aus Hypothese **II**.
U_X und U_Y skaliert, mit 10^6. n entspricht der Anzahl an Subscribern.

Wie aus Tbl. 4.17 hervorgeht, sind die Kombinationen $2/2$ und $3/3$ mit $U_X = 39$, $U_Y = 30$, $z = -14.49$, $p < 0.001$ und $r^2 = 0.01$ und $U_X = 127$, $U_Y = 30$, $z = -84.33$, $p < 0.001$ und $r^2 = 0.28$ stark signifikant. Kombination $1/1$ zeigt kein signifikantes Ergebnis ($p > 0.05$).

Dieses Verhalten wird auch anhand der Boxplots aus Abb. 4.14a und 4.14b ersichtlich. Es ist eine Steigerung der Latenzverteilung bei der Queue zu erkennen. Die Latenzverteilung der Topic zeigt nur leichte Änderungen des Median.

Für Hypothese **II** wird für steigende Subscriber gefolgert: Die Latenz einer Topic für eine steigende Anzahl an Subscribern ist stochastisch kleiner als bei einer Queue.

4.3 VERGLEICH UND DISKUSSION DER ERGEBNISSE

In diesem Abschnitt werden die Ergebnisse der Testreihen anhand der in Abschnitt 4.2 aufgestellten Hypothesen verglichen. Weiterhin wird eine Empfehlung der getesteten Subskriptionsmethoden und Protokolle gegeben.

In Testreihe 1 wurde nach Hypothese **I** untersucht, ob Unterschiede in der Latenz des Systems auf die Verwendung unterschiedlicher Protokolle zurückzuführen sind. Statistisch signifikante Unterschiede liegen zwischen *HTTP*, *AMQP* und *HTTP*, *MQTT* vor. Die Latenz ist unter Verwendung von *HTTP* signifikant größer als bei *AMQP* und *MQTT*. Basierend auf dem Testaufbau kann angenommen werden, dass die nötige Nachrichtenweiterleitung über die *CPI* diesen Unterschied erklärt. *MQTT* und *AMQP* zeigten keinen signifikanten Unterschied.

Alle drei Protokolle sind echtzeitfähig, die Latenzzeiten liegen zwischen 200 und 500 Millisekunden. Die dedizierte Zeitstempelabfrage führt ab etwa 245 *Nachrichten pro Minute (Npm)* zu einer Überlastung des Systems. Wie in Testreihe 2 gezeigt wurde, sind Latenzzeiten von wenigen ms möglich. Die Diskrepanz der Testreihen ist auf die Zeitstempelabfrage zurückzuführen.

In Testreihe 2 wurde der Unterschied zwischen einer Topic- und Queue-Subskription untersucht. Angenommen wurden kleinere Latenzzeiten bei einer Topic-Subskription nach Hypothese **II**. Die Durchführung erfolgte jeweils mit einem Subscriber. Da kein signifikanter Unterschied zwischen *MQTT* und *AMQP* nach Hypothese **I** vorliegt, wurde in Testreihe 2 *MQTT* für die Topic- und *AMQP* für die Queue-Subskription verwendet. Dies ist ebenfalls darauf zurückzuführen, dass das *EM*-System der SAP aktuell keine Queue-Subskription für *MQTT* unterstützt.

Die aufgestellte Hypothese wurde ausgehend von dem 75% Quantil der Nachrichten des produktiven Systems bis zu 200% der Spitzenbelastung durchgeführt. In 7 von 8 Gruppen zeigte, entgegen der Erwartung, eine Queue stochastisch gleichwertig oder kleinere Latenzzeiten als ein Topic. Es wird angenommen, dass eine Queue-Subskription kleinere Latenzzeiten als eine Topic-Subskription für einen Subscriber hat.

Weiterhin wurde nach Hypothese **III** erwartet, dass mit steigender Anzahl an *Npm* die Latenz des Systems signifikant zunimmt. Bei 4 von 7 Gruppen konnte die Erwartung der Latenzsteigerung im nahezu linearen Bereich von ≤ 2 *ms* gezeigt werden. Ab 245 *Npm* zeigt das System höhere Variiationen der Latenzzeiten. Unter Berücksichtigung der deskriptiven Statistik wurde für steigende *Npm* ein Einfluss auf die Latenzzeiten angenommen. Die Hypothese konnte nicht eindeutig verworfen werden.

Bei beiden Varianten lag die Latenz im Bereich von 8 bis 13 Millisekunden. Folglich sind die Latenzzeiten nach Definition echtzeitfähig und die Nachrichtenmenge skalierbar. Die Wahl der Subskription sollte anhand von Kompatibilitätskriterien erfolgen.

Mit Testreihe 3 wurde eine Untersuchung der Latenzzeiten bei steigender Anzahl an Subscribern angestrebt. Nach Hypothese **IV**, wurde eine Zunahme der Latenzzeit erwartet. Die Testreihe wurde sowohl für Topic- als auch Queue-Subskriptionen durchgeführt. Weiterhin wurde die Testreihe gemäß einer *RU* auf maximal drei parallel aktive Subscriber beschränkt.

Die Queue-Variante zeigte signifikante Ergebnisse bei jeder Steigerung der aktiven Subscriber. Die Topic-Variante hingegen, zeigte erst bei 3 Subscribern einen signifikanten Unterschied. Weiterhin wurde gezeigt, dass bei einer steigenden Anzahl an Subscribern, die Queue- entgegen Hypothese **II** statistisch kleinere Latenzzeiten aufweist als die Topic-Subskription. Dies könnte auf den technologischen Unterschied zwischen einer Topic- und einer Queue-Subskription zurückzuführen sein. Wie in Abschnitt 3.3.1.1 beschrieben, hält eine Queue-Subskription die Daten solange vor, bis diese konsumiert wurden. Bei hoher Last könnte dieser Unterschied dazu führen, dass die Queue multiple Versuche benötigt, um die Nachricht zuzustellen und damit zu einer höheren Latenz führen.

Die Latenzzeiten lagen in Testreihe 3 für drei Subscriber unter 20 Millisekunden. Demnach ist das System skalierbar und auch unter Spitzenbelastung (140 *Npm*) für multiple Applikationen echtzeitfähig.

Eine allgemeine Empfehlung kann anhand der Anforderungen nicht ausgesprochen werden.

Anhand der Echtzeitanforderung der Nachrichtenübertragung in Sekunden bis Minuten, sind alle Varianten und Protokolle geeignet. Die Latenz liegt selbst bei 200% der Spitzenbelastungen noch im zweistelligen Millisekunden Bereich. Bei Kompatibilitätseinschränkungen des Publisher oder Subscriber hinsichtlich der Protokolle, ist *HTTP* mit dem verfügbaren Service über eine *WSS* geeignet. Die Verwendung eines oder multipler dedizierter Publisher, welche eine einfache Schnittstelle für multiple Applikationen bereitstellen, ist zu empfehlen. Bei der Wahl des Protokolles, sollte die Kompatibilität des Zielsystems im Vordergrund stehen. Hinsichtlich der Subskriptionsvarianten, sollte je nach *QoS* Anforderung entschieden werden. Sofern keine Notwendigkeit besteht jede Nachricht zu erhalten, bietet sich eine Topic-Subskription an. Wenn die Zielapplikation nicht über die Möglichkeit verfügt Daten vorzuhalten, sollte eine Queue-Subskription verwendet werden. Dabei sollte die Einschränkung des Systems für *MQTT* für eine Topic-Subskription mit multiplen Queues berücksichtigt werden. In dieser Variation ist *AMQP* vorzuziehen.

Die Konfiguration des Systems sollte basierend auf den genannten Abwägungen gewählt werden.

5 FAZIT UND AUSBLICK

5.1 FAZIT

Ziel dieser Arbeit war die Konzeption, Umsetzung und Prüfung einer echtzeitfähigen, eventgetrieben und nachrichtenbarsierten Integrationsarchitektur, sowie die Erarbeitung einer Methode zur Einordnung von neuen Gerätestammdaten. Es sollte evaluiert werden, ob ein Duplikat vorliegt, welches durch eine automatische Zusammenführung bereinigt werden kann. Zu diesem Zweck wurden drei Schwerpunkte untersucht:

1. Entwicklung einer in die bestehende Infrastruktur integrierbaren, eventgetrieben und nachrichtenbasierten Integrationsarchitektur.
2. Analyse und Klassifizierung von Duplikaten anhand ausgewählter statistischer Methoden.
3. Umsetzung einer Architektur anhand eines Anwendungsfalles und die Prüfung der Echtzeitfähigkeit und Skalierbarkeit unter verschiedenen Last- und Verarbeitungsszenarien.

Zur Entwicklung einer Integrationsarchitektur wurde zunächst ein Teil der bestehenden Infrastruktur von *MU* untersucht. Anschließend wurden drei *EM*-Systeme mithilfe einer Nutzwertanalyse verglichen und bewertet. Die *EM*-Lösung Event Mesh erzielte den größten Nutzwert und wurde als Basis für die Integrationsarchitektur verwendet. Anhand der definierten Ziele, der Integrier- und Nutzbarkeit sowie der Funktionalität, wurde eine Integrationsarchitektur entworfen und diskutiert.

Bei der Untersuchung der Gerätestammdaten konnte festgestellt werden, dass 12 der 21 betrachteten Merkmale wenig bis keinen Informationsgehalt besitzen. Weiterhin zeigten Datensätze vor 2005 eine von dem *MU* Standard abweichende Struktur, welcher möglicherweise auf einen Wechsel des Standards zurückzuführen ist. Infolgedessen wurden Datensätze vor 2005 verworfen. Nach der Datenbereinigung wurde anhand von Stichprobenuntersuchungen und unter Konsultation interner Experten, eine Annahme für Duplikate formuliert. Die formulierte Hypothese wurde mithilfe einer Faktoranalyse und einem K-Means Clustering geprüft. Aus der Untersuchung ging hervor, dass die Annahmen über die Zusammenhänge der Merkmale zutreffend sind. Der angenommene Grenzwert für ein Duplikat konnte mit einer eindeutigen Zuordnung von 87.5% der Duplikate zu einem Cluster, ausreichend bestätigt werden. Somit konnten 12.16% aller relevanten Duplikate als automatisch zusammenführbar eingeordnet werden.

Abschließend wurde eine Zielarchitektur und drei Methoden zur Publizierung von Events vorgestellt. Anhand eines Anwendungsfalles, wurde die Echtzeitfähigkeit und Skalierbarkeit unter verschiedenen Last- und Verarbeitungsszenarien geprüft. Neben der Performanz konnte durch definierte Hypothesen, Aufschluss über die Unterschiede von vier möglichen Subskriptionsvarianten gegeben werden.

Die Analyse der Testreihen hat ergeben, dass alle möglichen Kommunikationsprotokolle der definierten Echtzeitfähigkeit von Sekunden bis Minuten gerecht werden. Die Latenzen liegen bei 200% der Spitzenbelastung unter 15 Millisekunden, sofern auf eine dedizierte Zeitabfrage verzichtet wird. Unter der maximal zulässigen Anzahl an Subscribern einer *RU*, konnten bei Spitzenbelastung Latenzzeiten von unter 20 Millisekunden nachgewiesen werden.

Weiterhin konnte gezeigt werden, dass entgegen der Annahme, eine Topic-Subskription keine signifikant kleinere Latenzzeit als eine Queue-Subskription aufweist. Der Einfluss steigender Nachrichten pro Minute auf die Latenz konnte nur teilweise durch die nicht eindeutigen Ergebnisse erklärt werden. Letztlich wurde gezeigt, dass eine steigende Anzahl an parallel subskribierten Applikationen einen siginifikanten Einfluss auf die Latenz bei Queue-Subskriptionen hat. Die Topic-Subskriptionen hingegen zeigten kein eindeutiges Ergebnis.

Somit wurde eine eventgetrieben und nachrichtenbasierte Integrationsarchitektur erarbeitet, die skalierbar und in allen Kategorien echtzeitfähig ist. Diese weist zudem eine hohe Kompatibilität für die bestehende Infrastruktur auf und biete Schnittstellen für externe Systeme und Plattform eigener Services, welche beispielsweise zur Authentifizierung genutzt werden können. Zusätzlich wurde eine Methode zur Einordnung von Duplikaten entwickelt und bereits 5929 Einträge identifiziert, welche durch eine automatische Zusammenführung bereinigt werden können. Die Ergebnisse werden persistiert und stehen zur Initialisierung weiterer Prozesse zur Bereinigung der Daten zur Verfügung. Dies ermöglicht eine nahtlose Einbindung in bestehende Integrationsprozesse und eine Minimierung falscher Einträge für den zukünftigen Datenbestand.

5.2 AUSBLICK

Bei der Durchführung konnten drei Bereiche identifiziert werden, die einer weiteren Untersuchung bedürfen und in zukünftigen Szenarien gewinnbringend für *MU* wären. Im Folgenden werden diesbezüglich Vorschläge zur weiteren Untersuchung beschrieben.

In der Analyse und Auswahl einer geeigneten *EM*-Lösung wurde auf die Kompatibilitätsvorteile von Event Mesh in der Infrastruktur von *MU* hingewiesen. Zukünftig wäre eine Untersuchung der Varianten und Methoden zur Publizierung von businessgetriebenen Events empfehlenswert. Die im Fazit ausgesprochene Eignung der Architektur in bestehende Integrationsprozesse kann auch für weitere Bereiche des Unternehmens nützlich sein.

In diesem Kontext kann eine Methode zur Provisionierung unterschiedlicher Anwendungsfälle untersucht werden. Diese Methode soll die Kapazitätseinschränkung einer *RU* berücksichtigen und die Event Mesh Instanzen entsprechend dimensionieren.

Des Weiteren zeigt die Untersuchung der Gerätestammdaten, Potenzial für eine detailliertere Betrachtungsweise dieser. Anhand der aufgestellten Hypothese der Duplikatszuordnung wurden neben der Zielgruppe zusätzliche Gruppen identifiziert, welche weiterer Untersuchung bedürfen. Auch die Untersuchung der programmatischen Zusammenhänge von Merkmalen könnte anhand der Gerätespezifikation erweitert werden. Hierfür könnte bei Sensoren auf Messtechnik spezifische Konfigurationen zurückgegriffen werden. Diese Fragen müssten zukünftig weiter untersucht werden, um den Anteil der automatisch zusammenführbaren Geräte zu erhöhen.

Bei der Durchführung der Testreihen wurde anhand unterschiedlicher Lastszenarien die Eignung des Systems für hochfrequenten Datentransfer gezeigt. Die Prüfung des Systemverhaltens bei geplanten Wartungsarbeiten oder ungeplantem Systemausfall bedarf weiterer Betrachtung. Bezüglich des Systemausfalls könnten Testreihen zur Ermittlung des potenziellen Datenverlustes durchgeführt werden. Sollte es zu Datenverlust kommen, bedürfe es der Entwicklung und Umsetzung von Mechanismen zur ausfallgeschützten Datensicherung.

Teil II

APPENDIX

A

PROGRAMMCODE

In diesem Teil des Anhangs werden zunächst die verwendeten SQL Abfragen gelistet.

Listing A.1: SQL-Abfrage der *Base* Daten

```
SELECT A.*, AE.cntEvents, AE_SDOR.cntEventsSDOR, MAT."MATERIAL_TYPE",
    MATTEXT."TEXT" FROM
(
SELECT * FROM
(  SELECT SUM(1) OVER(PARTITION BY "SERIAL_NUMBER") AS cnt, A.*
       FROM "ASSET_CENTRAL"."ASSET" a WHERE "SERIAL_NUMBER" IS NOT NULL
      AND "SERIAL_NUMBER" != ''"
      LEFT JOIN
      (
       SELECT SUM(1) OVER(PARTITION BY "ASSET_ID") AS cntEvents, A.*
       FROM "ASSET_CENTRAL"."ASSET_EVENT" a
       WHERE "ASSET_ID" IS NOT NULL
      ) AS AE
      ON A."ASSET_ID" = AE."ASSET_ID"
      LEFT JOIN
      (
       SELECT SUM(1) OVER(PARTITION BY "ASSET_ID") AS cntEventsSDOR, A.*
       FROM "ASSET_CENTRAL"."ASSET_EVENT" a
       WHERE "ASSET_ID" IS NOT NULL
       AND "EVENT_TYPE" = 'SDOR'
      ) AS AE_SDOR
      ON A."ASSET_ID" = AE_SDOR."ASSET_ID"
      LEFT JOIN "ASSET_CENTRAL"."MATERIAL_TEXT" AS MATTEXT
      ON A."MATERIAL_NUMBER" = MATTEXT."MATERIAL_NUMBER"
      LEFT JOIN "ASSET_CENTRAL"."MATERIAL" AS MAT
      ON A."MATERIAL_NUMBER" = MAT."MATERIAL_NUMBER"
      WHERE A.cnt > 1 AND MATTEXT."LANGUAGE" = 'EN'
      )
WHERE cnt = 1 ORDER BY "SERIAL_NUMBER" AS T
)
ORDER BY RAND() LIMIT 350000
```

Listing A.2: SQL-Abfrage der *Dupl* Daten

```
SELECT DISTINCT A.*, AE.cntEvents, AE_SDOR.cntEventsSDOR, MAT."MATERIAL_
    TYPE", MATTEXT."TEXT" FROM
(
 SELECT SUM(1) OVER(PARTITION BY "SERIAL_NUMBER") AS cnt, A.*
 FROM "ASSET_CENTRAL"."ASSET" a
 WHERE "SERIAL_NUMBER" IS NOT NULL
 AND "SERIAL_NUMBER" != ''
 AND "DELETION_FLAG" != TRUE
) AS A
LEFT JOIN
(
 SELECT SUM(1) OVER(PARTITION BY "ASSET_ID") AS cntEvents, A.*
 FROM "ASSET_CENTRAL"."ASSET_EVENT" a
 WHERE "ASSET_ID" IS NOT NULL
) AS AE
ON A."ASSET_ID" = AE."ASSET_ID"
LEFT JOIN
(
 SELECT SUM(1) OVER(PARTITION BY "ASSET_ID") AS cntEventsSDOR, A.*
 FROM "ASSET_CENTRAL"."ASSET_EVENT" a
 WHERE "ASSET_ID" IS NOT NULL
 AND "EVENT_TYPE" = 'SDOR'
) AS AE_SDOR
ON A."ASSET_ID" = AE_SDOR."ASSET_ID"
LEFT JOIN "ASSET_CENTRAL"."MATERIAL_TEXT" AS MATTEXT
ON A."MATERIAL_NUMBER" = MATTEXT."MATERIAL_NUMBER"
LEFT JOIN "ASSET_CENTRAL"."MATERIAL" AS MAT
ON A."MATERIAL_NUMBER" = MAT."MATERIAL_NUMBER"
WHERE A.cnt > 1 AND MATTEXT."LANGUAGE" = 'EN'
ORDER BY A."SERIAL_NUMBER"
```

Listing A.3: SQL-Abfrage zur Evaluation der Nachrichten pro Minute Parameters

```

FROM hour("MODIFIED_DATE_TIME") as hour_created,
COUNT(*) AS per_hour, COUNT(*) / 60 AS per_minute
FROM "ASSET_CENTRAL"."ASSET"
WHERE ("MODIFIED_DATE_TIME" IS NOT NULL AND "MODIFIED_DATE_TIME" >
    '2021-07-18' AND "MODIFIED_DATE_TIME" < '2021-07-26' AND "CREATE_
    DATE" < '2021-07-18')
OR ("MODIFIED_DATE_TIME" IS NULL AND "CREATE_DATE" > '2021-07-26' AND "
    CREATE_DATE" < '2021-07-26')
GROUP BY hour("MODIFIED_DATE_TIME")
ORDER BY hour("MODIFIED_DATE_TIME") ASC;
```

Im Folgenden werden Ausschnitte des Programmcode der Datenselektion, -transformation und -partitionierung dargelegt.

Listing A.4: Transformation der Daten mithilfe der Damerau-Levenshtein und Hamming Distanz.[1]

```
import jellyfish

# speicher intialen index
base_df['initial_ind'] = base_df.index.values
# Select first entry per group
tmp = base_df.sort_values(['CREATE_DATE', 'MODIFIED_DATE'], ascending=
    True).groupby('SERIAL_NUMBER').nth(0)
tmp['FIRST_ENTRY'] = 1 # add first entry flag
# join on initial df
param_df_first_entry = base_df.join(tmp.reset_index().set_index('initial
    _ind')['FIRST_ENTRY'], how='left', on='initial_ind')
param_df_first_entry = param_df_first_entry.drop(columns=['initial_ind
    '])
param_df_first_entry['FIRST_ENTRY'].fillna(0, inplace=True)
sorted_df = param_df_first_entry.sort_values(['SERIAL_NUMBER', 'FIRST_
    ENTRY'], ascending=False)
lst_df = []
lst_cols_to_diff = ['MN', 'OC', 'T', 'CN', 'CCC', 'CPON', 'FV', 'FB', '
    MT']

for sn in sorted_df.SERIAL_NUMBER.unique():
   sub = sorted_df.loc[sorted_df.SERIAL_NUMBER == sn].fillna('').copy()
   # Calc Levenshtein und speicher sub werte
   for col in lst_cols_to_diff:
      str_lengths = sub[col].apply(str).str.len().values
      sub[col + '_len'] = str_lengths
      sub[col + '_len_base'] = [str_lengths[0]] * len(str_lengths)
      sub[col + '_lev_dist'] = -1
      sub[col + '_lev_dist'].iloc[1:] = [jellyfish.damerau_levenshtein_
          distance(sub[col].apply(str).iloc[0], sub[col].apply(str).iloc
          [k+1]) for k in range(len(sub[col].apply(str).iloc[1:]))]
      sub[col + '_lev_dist_ratio'] = -1
      if sum(str_lengths) != 0:
         sub[col + '_lev_dist_ratio'].iloc[1:] = sub[col + '_lev_dist'].
             iloc[1:].div(max(sub[[col + '_len', col + '_len_base']].
             iloc[1:].max(axis=0)))
   # Calculate Hamming
   sub['hamming_dist'] = -1
   sub['hamming_dist'].iloc[1:] = [distance.hamming(sub[lst_cols_to_diff
       ].iloc[k+1], sub[lst_cols_to_diff].iloc[0]) for k in range(len(
       sub[lst_cols_to_diff].iloc[1:]))]
   lst_df.append(sub)

result_diff = pd.concat(lst_df)
```

1 jellyfish Package, 0.8.8

Listing A.5: Durchführung der Faktoranalyse des optimierten Aufbaus[2]

```
from factor_analyzer import FactorAnalyzer
n_fa = 3

X = result_diff.copy()
fa_sub = FactorAnalyzer(rotation='promax', n_factors=n_fa, method='
    principal')
fa_sub.fit(X)

# Varianz der Faktoren
pd.DataFrame(fa_sub.get_factor_variance(), columns = ['FA{}'.format(i)
    for i in range(1, n_fa+1)],
          index = ['SS_Ladungen', 'Proportionale_Varianz', 'Kumulative_
              Varianz'])

# Generieren Ladugnsmatrix
fa_ladungsmatrix = pd.DataFrame(fa_sub.loadings_, columns=['FA{}'.format
    (i) for i in range(1, n_fa+1)],
                    index = sub_cols)
# Print mit maximaler Ladungs
fa_ladungsmatrix['max Ladung'] = fa_ladungsmatrix.idxmax(axis=1)
```

Listing A.6: Durchführung der ersten Iteration des K-Means Clustering[3]

```
from sklearn.metrics import silhouette_samples, silhouette_score
from sklearn.cluster import KMeans

def run_clustering(fa_data, base_data, sub_cols,
                   init='random', rows=fa_data.shape[0]):
   range_n_clusters = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,
       16, 17, 18, 19, 20]
   lst_dfs = [fa_data, base_data[sub_cols]]
   lst_agg_dict = []
   for n_clusters in range_n_clusters:
      for i, val in enumerate(['FA', 'BASE']):
         X = lst_dfs[i]
         clusterer = KMeans(n_clusters=n_clusters, random_state=10, init
             =init)
         cluster_labels = clusterer.fit_predict(X)
         sil_score, sum_sqaured_dist = silhouette_score(X, cluster_
             labels), clusterer.inertia_
         # Speichern der Daten
         lst_agg_dict.append({'n_cluster':n_clusters, 'data':val, 'sil_
             score':sil_score, 'ssd':sum_sqaured_dist})
  return pd.DataFrame(lst_agg_dict).sort_values('data')
```

2 factor_analyzer Package, 0.3.1
1 factor_analyzer Package, 0.3.1
2 sklearn Package, 0.24.0

In diesem Teil des Anhangs folt der Programmcode der Konzeption. Dabei wird ein Beispiel für eine *GET* Anfrage sowie der Nodejs Publisher, das zugehörige yml und der Subscriber aufgeführt.

Listing A.7: GET Anfrage für ein Bearer Token. Die Authentifizierung erfolgt mit einer Basic Authentification.

```
import requests

url = "https://<host>.authentication.eu10.hana.ondemand.com/oauth/token?
    grant_type=client_credentials&response_type=token"
payload={}
headers = { 'Authorization': 'Basic <verschluesselte Name und Passwort'}

response = requests.request("GET", url, headers=headers, data=payload)
# "token":"xxx..."
```

Listing A.8: manifest.yml des *Publisher*

```
---
applications:
- name: producerThesisSingleLongTermAMQP
  domain: https://enterprise-messaging-pubsub.<host>.com/
  routes:
  - route: producer-thesis-single-long-term-amqp-host.<host>.com
  path: .
  memory: 128M
  services:
    - xsuaaThesisProducer
    - emLennartTest
  env:
     SAP_JWT_TRUST_ACL: '[{"clientid":"*","identityzone":"*"}]'
     SAP_XBEM_BINDINGS: |
       {
         "outputs": {
           "myOutAMQP" : {
             "service": "emLennartTest",
             "address": "topic:ati/emLennartTest/pubsubls/asset",
             "reliable": false },
           "myOutMQTT" : {
             "service": "emLennartTest",
             "address": "topic:ati/emLennartTest/pubsubls/asset",
             "reliable": false},
           "myOutHTTP" : {
             "service": "emLennartTest",
             "address": "topic:ati/emLennartTest/pubsubls/asset",
             "reliable": false}}}
```

Listing A.9: Nodejs Publisher JavaScript Code

```
'use strict';

// App Setup
const express = require("express");
const bodyParser = require("body-parser");

// Messaging Setup
const AMQP = require('@sap/xb-msg-amqp-v100');
//const MQTT = require('@sap/xb-msg-mqtt-v311');
//const msg = require('@sap/xb-msg');
//const env = require('@sap/xb-msg-env');
const xsenv = require('@sap/xsenv');
xsenv.loadEnv();
// EM Instanz
const service = 'emLennartTest';
//-----------------------------------------
// Start app
const app = express();
const PORT = process.env.PORT || 8080;
app.use(bodyParser.json());

const taskList = {
   myOutAMQP : { topic: 'ati/emLennartTest/pubsubls/asset'},
   //myOutMQTT : { topic: 'ati/emLennartTest/pubsubls/asset'},
   //myOutHTTP: { topic: 'ati/emLennartTest/pubsubls/asset'}
};

// Variablen und Const
var _client = undefined;
const reconnect_retry_ms = 500;
//-----------------------------------------------------------
// Methods functions
//-----------------------------------------------------------
app.post('/', (req, res) => {
 const reqInBody = req.body;
 // Sanity check
 if(reqInBody.constructor === Object && Object.keys(reqInBody).length
     === 0)
 {
  console.log('Object missing');
  res.status(204).send(outobject).end();
 };

if(reqInBody.constructor === Object && Object.keys(reqInBody).length !=
     0)
 {
  var infoTimestamp = new Date().getTime();

  // Weiterleiten der Nachricht
```

```
    send(taskList, _client, reqInBody.assetId, infoTimestamp, reqInBody.
        run_id);

    // Reponse an Sender
    res.status(200).send(JSON.stringify(
     {
     'assetId':reqInBody.assetId,
     'currTimestamp':infoTimestamp.curr_time,
     'run_id':reqInBody.run_id
     })
    ).end();
    };
});

//-------------------------------------------------------------
// Messaging functions
//-------------------------------------------------------------
function returnClient(){
 const client = new AMQP.Client(env.msgClientOptions(service, [], ['
     myOutAMQP']));
 //const client = new msg.Client(env.msgClientOptions(service, [], ['
     myOutHTTP']));
 //const client = new MQTT.Client(env.msgClientOptions(service, [], ['
     myOutMQTT']));
 return client;
}

function send(tasks, client, assetId, timestamp, run_id)
{
   Object.getOwnPropertyNames(tasks).forEach((id) =>
   {
      const task = tasks[id];
      const stream = client.ostream(id);

      const message =
      {
         payload: Buffer.from(JSON.stringify({'assetId':assetId, '
             timestamp': timestamp, 'run_id':run_id}))
      };
      console.log('Published : ' + JSON.stringify({'assetId':assetId, '
          timestamp':timestamp, 'run_id':run_id}) + ' to Topic: ' + id);
      // Publish to stream
      if (!stream.write(message))
      {
       console.log('wait');
      }
   });
}
app.listen(PORT, function () {
   console.log('App listening at port://' + PORT);
   _client = returnClient();
```

```
   // Handle client
   _client
97 .on('connected', () => {
      console.log('connected');
   })
   .on('drain', () => {
      console.log('continue');
102 })
   .on('error', (error) => {
      console.log(error);
   })
   .on('disconnected', (hadError) => {
107    setTimeout(()=> client.connect(), reconnect_retry_ms);
   });
   _client.connect();
});
```

Im Folgenden wird der Nodejs Subscriber aufgeführt.

Listing A.10: Nodejs Subscriber JavaScript Code

```
'use strict';

//-------------------------------------------------------------
4 // Setup Event Mesh Instanz
//-------------------------------------------------------------
const msg = require('@sap/xb-msg');
const msgenv = require('@sap/xb-msg-env');
const MQTT = require('@sap/xb-msg-mqtt-v311');
9 // Variablen und Kosntanten
const inputX = process.env.XBEM_INPUT_X;
const reconnect_retry_ms = process.env.RECONNECT_RETRY_MS;
var topicName = 'ati/emLennartTest/pubsubls/assetSingleMqtt'
//-------------------------------------------------------------
14 // Start messaging client
//-------------------------------------------------------------
const options = {
   oa2: {
      endpoint: 'https://<host>.com/oauth/token',
19    client: 'sb-default-<Client Identifier>',
      secret: <Passwort>',
   },
   wss: {
      host: 'em-<host>.com',
24    port: 443,
      path: '/protocols/mqtt311ws'
   },
   qos: 1
};
29

// MQTT Client
const client = new MQTT.Client(options);
```

```
// Methode
function receive(topic, payload, qos, duplicate) {
   console.log('Topic: ' + topic);
   console.log('Payload: ' + payload);
   console.log('Qos: ' + qos);
   console.log('duplicate: ' + duplicate);
}
client
.on('connected', () => {
 console.log('connected to Event Mesh');
 client.subscribe(topicName, options.qos, receive, () => console.log('
     subscribed'));
})
.on('message', function (topic, message) {
 context = topic + " : " + message.toString();
 var date = new Date();
 console.log(date.getTime() + " - " + context)
})
.on('error', (err) => {
 console.log('error occurred ' + err);
})
.on('disconnected', (hadError) => {
 console.log( 'trying to reconnect in ' + reconnect_retry_ms + ' ms');
 setTimeout(()=> client.connect(), reconnect_retry_ms);
});
client.connect();
```

Listing A.11: Lokale Python Routine der Testreihen.

```
def threaded_process(items_chunk, bearer_token, run_id, _url=<Zielapp
    URL>):
      response_lst = []
      for i, asset in enumerate(items_chunk):
            try:
                  start_time = datetime.now(timezone.utc)
                  # Send request
                  resp = send_request_single_long_term(bearer_token,
                      asset, run_id=run_id, _url=_url)
                  end_time = datetime.now(timezone.utc)
                  delta = (end_time - start_time).microseconds / 1000
                  # Calculate time to wait until new post
                  t_to_wait = (1000 - delta) / 1000
                  time.sleep(t_to_wait)
            except Exception as e:
                  print('error with item: {}, err: {}'.format(asset, e))
```

B

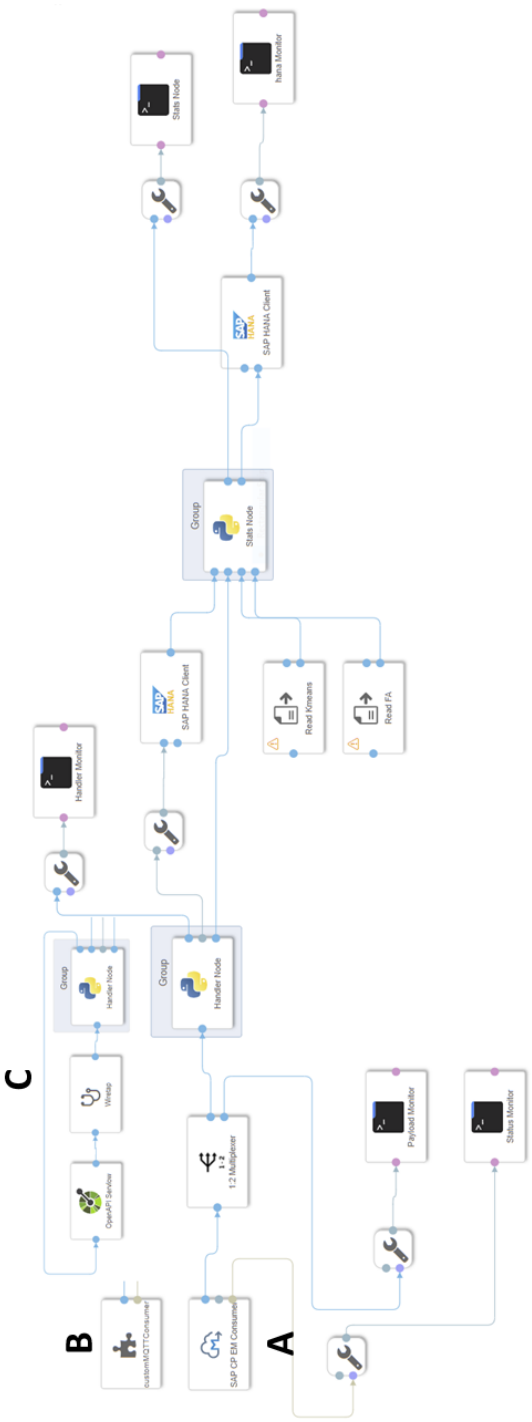


Abbildung B.1: Pipeline in der *Data Intelligence* zum Empfang, der Verarbeitung und persistierung der Nachrichten.
Subscriber A entspricht einem OpenApi Server, welcher WSS kompatibel ist.
Subscriber B entspricht einem erzeugten Nodejs Operator für MQTT.
Subscriber C entspricht einem Nodejs Operator für AMQP.

[1] M. Fowler, *Patterns of Enterprise Application Architecture*. USA: Addison-Wesley Longman Publishing Co., Inc., 2002, ISBN: 0321127420.

[2] G. Hohpe und B. Woolf, *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. USA: Addison-Wesley Longman Publishing Co., Inc., 2003, ISBN: 0321200683.

[3] A. Böhm und C.-C. Kanne, *Messaging Rules as a Programming Model for Enterprise Application Integration*, English, 2009. Adresse: https://madoc.bib.uni-mannheim.de/2372/.

[4] D. S. Linthicum, *Enterprise Application Integration*. GBR: Addison-Wesley Longman Ltd., 2000, ISBN: 0201615835.

[5] M. Rahman, *Basic Graph Theory*. BGD: Springer International Publishing AG, Jan. 2017, ISBN: 978-3-319-49474-6.

[6] J. P. Morgenthal und B. LaForge, *Enterprise Application Integration with XML and Java*. USA: Prentice Hall PTR, 2001, ISBN: 0130851353.

[7] K. Yusuf, *Enterprise Messaging Using JMS and IBM WebSphere*. USA: Prentice Hall PTR, 2004, ISBN: 0131468634.

[8] H. S. Pethuru Raj Anupama Raman, *Architectural Patterns*. Packt Publishing, 2017, ISBN: 9781787287495.

[9] T. K. Andreas Handl, *Einführung in die Statistik, Theorie und Praxis mit R*. GER: Springer Spektrum, Berlin, Heidelberg, 2018, ISBN: 978-3-662-56439-4. DOI: https://doi.org/10.1007/978-3-662-56440-0.

[10] M. T. Hans Friedrich Eckey Reinhold Kosfeld, *Deskriptive Statistik, Grundlagen – Methoden – Beispiele*. GER: Gabler, 2008, ISBN: 978-3-8349-0859-9. DOI: https://doi.org/10.1007/978-3-8349-8779-2.

[11] T. Cleff, *Angewandte Induktive Statistik und Statistische Testverfahren*. GER: Gabler, 2019, ISBN: 978-3-8349-0753-0. DOI: 10.1007/978-3-8349-6973-6.

[12] I. Frost, *Statistische Testverfahren, Signifikanz und p-Werte*. GER: VS Verlag für Sozialwissenschaften, 2017, ISBN: 978-3-658-16257-3. DOI: 10.1007/978-3-658-16258-0.

[13] S. S. Shapiro und M. B. Wilk, "An Analysis of Variance Test for Normality (Complete Samples)", *Biometrika*, Jg. 52, Nr. 3/4, S. 591–611, 1965, ISSN: 00063444. Adresse: http://www.jstor.org/stable/2333709.

[14] H. Levene, *Robust tests for equality of variances*. Palo Alto, CA: In: Contributions to Probability und Statistics: Essays in Honor of Harold Hotelling, I. Olkin, Stanford University Press, 1960, S. 278–292.

[15] M. B. Brown und A. B. Forsythe, "Robust Tests for the Equality of Variances", *Journal of the American Statistical Association*, Jg. 69, Nr. 346, S. 364–367, 1974. DOI: 10.1080/01621459.1974.10482955. eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1974.10482955. Adresse: https://www.tandfonline.com/doi/abs/10.1080/01621459.1974.10482955.

[16] N. Nachar, "The Mann-Whitney U: A Test for Assessing Whether Two Independent Samples Come from the Same Distribution", *Tutorials in Quantitative Methods for Psychology*, Jg. 4, März 2008. DOI: 10.20982/tqmp.04.1.p013.

[17] M. P. Fay und M. A. Proschan, "Wilcoxon-Mann-Whitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules", eng, *Statistics surveys*, Jg. 4, S. 1–39, 2010, 20414472[pmid], ISSN: 1935-7516. DOI: https://doi.org/10.1214/09-SS051.

[18] C. Duller, *Einführung in die nichtparametrische Statistik mit SAS, R und SPSS: Ein anwendungsorientiertes Lehr- und Arbeitsbuch*. Jan. 2018, ISBN: 978-3-662-57677-9. DOI: 10.1007/978-3-662-57678-6.

[19] S. Theodoridis und K. Koutroumbas, Hrsg., *Copyright*, Fourth Edition. Boston: Academic Press, 2009, ISBN: 978-1-59749-272-0. DOI: https://doi.org/10.1016/B978-1-59749-272-0.50001-3.

[20] N. Tokareva, *Distances Between Bent Functions*. Dez. 2015, S. 89–96, ISBN: 9780128023181. DOI: 10.1016/B978-0-12-802318-1.00011-X.

[21] C. Zhao und S. Sahni, "String correction using the Damerau-Levenshtein distance", *BMC Bioinformatics*, Jg. 20, Nr. 11, S. 277, 2019, ISSN: 1471-2105. DOI: 10.1186/s12859-019-2819-0. Adresse: https://doi.org/10.1186/s12859-019-2819-0.

[22] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006, ISBN: 0387310738.

[23] R. Cudeck, "10 - Exploratory Factor Analysis", in *Handbook of Applied Multivariate Statistics and Mathematical Modeling*, H. E. Tinsley und S. D. Brown, Hrsg., San Diego: Academic Press, 2000, S. 265–296, ISBN: 978-0-12-691360-6. DOI: https://doi.org/10.1016/B978-012691360-6/50011-2. Adresse: https://www.sciencedirect.com/science/article/pii/B9780126913606500112.

[24] L. Hatcher, *A Step-by-Step Approach to Using the SAS System for Factor Analysis and Structural Equation Modeling*, 1st. SAS Publishing, 1994, ISBN: 1555446434.

[25] M. von der Hude, *Predictive Analytics und Data Mining, Eine Einführung mit R*. 2020, S. 58–65, ISBN: 978-3-658-30152-1. DOI: 10.1007/978-3-658-30153-8.

[26] J. Wu, "Cluster Analysis and K-means Clustering: An Introduction", in Juli 2012, S. 1–16, ISBN: 978-3-642-29806-6. DOI: 10.1007/978-3-642-29807-3_1.

[27] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis", *Journal of Computational and Applied Mathematics*, Jg. 20, S. 53–65, 1987, ISSN: 0377-0427. DOI: https://doi.org/10.1016/0377-0427(87)90125-7. Adresse: https://www.sciencedirect.com/science/article/pii/0377042787901257.

[28] R. Scitovskim, K. Sabom, F. Martínez-Álvarezm und S. Ungar, "Cluster Analysis and Applications", in Cham, Switzerland: Springer Nature Switzerland AG, Juli 2012, ISBN: 978-3-030-74551-6. DOI: https://doi.org/10.1007/978-3-030-74552-3.

[29] SAP, *Business Technology Plattform Documentation*, [Online; Zugriff 16.05.2021], 2021. Adresse: https://help.sap.com/doc/bd6250c40c9c4c5391e3009a6f26dc3b/Cloud/en-US/SAP_Cloud_Platform.pdf.

[30] SAP, *Data Intelligence Documentation*, [Online; Zugriff 16.05.2021], 2021. Adresse: https://help.sap.com/doc/4004515a68e6441bb8d067c6ba8c9dba/3.1.latest/en-US/onprem_loio4004515a68e6441bb8d067c6ba8c9dba.pdf.

[31] H. Garcia-Molina und K. Salem, "Main memory database systems: an overview", *IEEE Transactions on Knowledge and Data Engineering*, Jg. 4, Nr. 6, S. 509–516, 1992. DOI: 10.1109/69.180602.

[32] C. Zangemeister, *Nutzwertanalyse in der Systemtechnik: eine Methodik zur multidimensionalen Bewertung und Auswahl von Projektalternativen*. GER, Berlin: Zangemeister & Partner, 1970, Bd. 05, ISBN: 978-3-923264-00-1.

[33] C. H. Kepner und B. B. Tregoe, "The New Rational Manager: An Updated Edition for a New World", *Princeton, N. J. (P. O. Box*, Jg. 704, 1997.

[34] SAP, *Event Mesh Documentation*, [Online; Zugriff 16.05.2021], 2021. Adresse: https://help.sap.com/doc/95ffc07cb5064bc5aaedf3b3172c28b8/Cloud/en-US/enterprise_messaging_en_US.pdf.

[35] Kafka, *Kafka Documentation*, [Online; Zugriff 16.05.2021], 2021. Adresse: https://kafka.apache.org/081/documentation.html#semantics.

[36] Microsoft, *Microsoft Azure Event Hub Documentation*, [Online; Zugriff 16.05.2021], 2021. Adresse: https://docs.microsoft.com/de-de/azure/event-hubs/.

[37] SAP, *SAP Invesort Relations*, [Online; 2. Quartal 2021]. Adresse: https://news.sap.com/2021/07/sap-announces-second-quarter-2021-results/.

[38] Microsoft, *Microsoft Investor Relations*, [Online; 2. Quartal 2021], 2021. Adresse: https://www.microsoft.com/en-us/investor.

[39] E. Thoo, M. Pezzini, K. Guttridge, B. Bhullar und S. P. andAbhishek Singh, "Magic Quadrant for Enterprise Integration Platform as a Service", Gartner, Inc., Discussion paper, 2021.

[40] SAP, *SAP Patch Day*, [Online; Zugriff 13.06.2021], 2021. Adresse: https://wiki.scn.sap.com/wiki/display/PSR/SAP+Security+Patch+Day.

[41] Kafka, *Kafka Patch Day*, [Online; Zugriff 13.06.2021], 2021. Adresse: https://kafka.apache.org/cve-list.

[42] Microsoft, *Microsoft Security Updates*, [Online; Zugriff 13.06.2021], 2021. Adresse: https://azure.microsoft.com/de-de/updates/?updateType=security&category=security&Page=1.

[43] IBM, *IBM Kafka Authentification*, [Online; Zugriff 13.06.2021], 2021. Adresse: https://developer.ibm.com/components/kafka/tutorials/kafka-authn-authz/.

[44] IBM, *IBM Kafka Training*, Online; Zugriff 13.06.2021, 2020. Adresse: https://developer.ibm.com/tutorials/deploying-and-using-a-basic-kafka-instance/.

[45] M. S. Bartlett, "Properties of Sufficiency and Statistical Tests", *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, Vol. 160, No*, Jg. 160, Nr. 901, S. 268–282, 1937.

[46] H. F. Kaiser, "An index of factorial simplicity", *Psychometrika*, Jg. 39, Nr. 1, S. 31–36, 1974, ISSN: 1860-0980. DOI: 10.1007/BF02291575. Adresse: https://doi.org/10.1007/BF02291575.

[47] T. Cleff, *Deskriptive Statistik und Explorative Datenanalyse: Eine computergestützte Einführung mit Excel, SPSS und STATA*, 3. Aufl. GER, Wiesbaden: Gabler Verlag, 2015, ISBN: 978-3-8349-4748-2.

[48] W. Möhring und D. Schlütz, *Handbuch standardisierte Erhebungsverfahren in der Kommunikationswissenschaft*. GER, Wiesbaden: Springer VS, 2013.

[49] L. Guttman, "Some necessary conditions for common-factor analysis", *Psychometrika*, Jg. 19, Nr. 2, S. 149–161, 1954, ISSN: 1860-0980. DOI: 10.1007/BF02289162. Adresse: https://doi.org/10.1007/BF02289162.

[50] D. Child, *Essentials of Factor Analysis*, 3. Aufl. Bloomsbury Publishing PLC, 2006, ISBN: 0826480004.

[51] A. L. Comrey und H. B. Lee, *A First Course in Factor Analysis (2nd ed.)* 2. Aufl. Hillsdale, NJ: Psychology Press, 1992.

[52] D. Marutho, S. Hendra Handaka, E. Wijaya und Muljono, "The Determination of Cluster Number at k-Mean Using Elbow Method and Purity Evaluation on Headline News", in *2018 International Seminar on Application for Technology of Information and Communication*, 2018, S. 533–538. DOI: 10.1109/ISEMANTIC.2018.8549751.

[53] L. Kaufman und P. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. Sep. 2009, ISBN: 9780470317488.

[54] M. Obert, *SAP Blog Post*, [Online; Zugriff 16.07.2021], 2018. Adresse: https://blogs.sap.com/2018/07/27/comparing-different-application-runtimes-on-sap-cloud-platform-cloud-foundry/.

[55] SAP, *SAP JavaScript Bibliothek*, [Online; Zugriff 16.07.2021], 2020. Adresse: https://www.npmjs.com/package/@sap/xb-msg.

[56] J. R. Fielding, *Hyptertext Transfer Protocol: Semantics and Content*, [Online; Zugriff 16.07.2021], 2014. Adresse: https://datatracker.ietf.org/doc/html/rfc7231#section-6.5.3.

[57] I. J. I. t. OASIS, *Information technology — Advanced Message Queuing Protocol (AMQP) v1.0*, [Online; Zugriff 16.07.2021], 2014. Adresse: https://www.iso.org/standard/64955.html.

[58] OASIS, *MQTT Version 3.1.1*, [Online; Zugriff 16.07.2021], 2014. Adresse: https://mqtt.org/mqtt-specification/.

[59] SAP, *Hana Administration Guide 2.0 SPS 01*, [Online; Zugriff 16.07.2021]. Adresse: https://help.sap.com/viewer/6b94445c94ae495c83a19646e7c3fd56/2.0.01/en-US/14d1158cefb74a58986be7de3f1a368b.html.

[60] SAP, *Job Scheduler Service, XS Advanced*, [Online; Zugriff 16.07.2021]. Adresse: https://help.sap.com/viewer/4505d0bdaf4948449b7f7379d24d0f0d/2.0.01/en-US/b2aff171211c4a4dbcbb55a7ebf98470.html.

[61] C.-F. Wu und M. Hamada, *Experiments: planning, analysis, and parameter design optimization*, 2. Aufl. USA, New York: Wiley, 2000, ISBN: 9780471699460.

[62] J. Cohen, *Statistical power analysis for the behavioral sciences*, 2. Aufl. Hillsdale, N.J. : L. Erlbaum Associates, 1988, ISBN: 9780805802832 080580283.