Institut für Parallele und Verteilte Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit

Entwicklung einer Datenbereitstellungsplattform für datenintensive IoT-Anwendungen

Heike Düsseldorf

Studiengang:	Technische Kybernetik
Prüfer/in:	PD Dr. Holger Schwarz
Betreuer/in:	Dr. Christoph Stach Rebecca Kay Eichler, M.Sc. Corinna Giebler, M.Sc.
Beginn am:	01. Juli 2020
Beendet am:	01. Dezember 2020

Kurzfassung

Internet of Things (IoT)-fähige Geräte halten in vielen Bereichen des alltäglichen Lebens Einzug. Dadurch steigt die Anzahl an Sensoren, die regelmäßig Daten bereitstellen können. Die große heterogene Datenmenge, die durch die verschiedenen Sensoren generiert wird, stellt IoT-Anwendungen jedoch vor eine Herausforderung bei der Datenverwaltung und -verarbeitung. Auch stellt sie eine Bedrohung der Privatsphäre dar. Nicht alle Daten dürfen allen Anwendungen bereitgestellt werden und durch die Anreicherung und Verknüpfung von Daten können neue Informationen generiert werden.

Frühere Arbeiten vereinen die Herausforderungen der Verwaltung von Big Data und den Schutz der Privatsphäre bei der Datenbereitstellung nicht. In dieser Arbeit wird daher RETORT, eine Dateibereitstellungsplattform für datenintensive Internet of Things-Anwendungen, vorgestellt und evaluiert. RETORT löst die Herausforderungen der Verwaltung von Big Data sowie den Schutz der Privatsphäre durch eine Datenreduktion.

Die Architektur gewährt externen Anwendungen Zugriff auf ausgewählte, vorverarbeitete Teilmengen der Daten aus unterschiedlichen, heterogenen Datenquellen, die in einem Data Lake verwaltet werden. Diese Daten können bei der Verarbeitung vor der Bereitstellung integriert und semantisch angereichert und verknüpft werden. Dieser Ansatz soll es externen Anwendungen ermöglichen, neue Funktionen basierend auf den bereitgestellten Daten und Informationen anzubieten, während die Privatsphäre der betroffenen Personen geschützt bleibt.

Abstract

Internet of Things (IoT)-enabled devices are finding their way into many areas of everyday life. This increases the number of different sensors that can regularly provide data. However, the large heterogeneous amount of data generated by the various sensors presents IoT applications with a challenge in terms of data management and processing. It also poses a threat to privacy. Not all data may be available to all applications, and new information can be generated by enriching and linking data.

Previous work does not combine the challenges of managing big data and protecting privacy when providing data. This thesis therefore presents and evaluates RETORT, a data provisioning platform for data-intensive Internet of Things applications. RETORT solves the challenges of managing big data and protecting privacy through data reduction.

The architecture grants external applications data access to selected, preprocessed subsets of the data from different, heterogeneous data sources that are managed in a data lake. This data can be integrated and semantically enriched and linked during processing prior to the data provision. With this approach external applications can have a wider range of functions based on the data and information provided, while protecting the privacy of the data subjects.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Listings

Akronyme

ACID Atomicity, Consistency, Isolation, Durability

API Application Programming Interface

BASE Basically Available, Soft State, Eventually Consistent

BDSG a.F. Bundesdatenschutzgesetz alte Fassung

BMI Body Mass Index

DBMS Database Management System

DIKW data-information-knowledge-wisdom

DSGVO Datenschutz-Grundverordnung

eGa elektronische Gesundheitsakte

FIFO First In – First Out

FOAF Friend of a Friend

IoT Internet of Things

IRI Internationalized Resource Identifier

LIFO Last In – First Out

LPG Labeled Property Graph

NoSQL Not Only SQL

RDBMS Relational Database Management System

RDF Resource Description Framework

RDFS RDF-Schema

RETORT Privacy-Aware Data Annotation Framework for the Internet of Things

SPARQL SPARQL Protocol and RDF Query Language

SQL structured query language

UML Unified Modeling Language

Kapitel 1

Einleitung

Die Vernetzung der Welt der Dinge mit dem Internet in einem Internet of Things (IoT) rückt immer mehr in den Fokus von Unternehmen. Das IoT steht dabei für die Vision einer vollständig vernetzten Welt. Nicht mehr nur PCs und mobile Endgeräte, sondern die ganze Welt der Dinge wird untereinander und mit dem Internet vernetzt [BZ18].

Den Inhalt des IoT bilden sogenannte Smart Things. Dabei handelt es sich um digitale oder physische Objekte, die in der Lage sind, Daten zu sammeln, zu speichern und zu verarbeiten, sowie die gesammelten Daten für andere Smart Things oder Anwendungen bereitzustellen. Außerdem sind sie in der Lage sich untereinander zu verbinden und zu interagieren. Aber auch die Interaktion mit Menschen ist möglich [KKSF10]. Mögliche Anwendungsbereiche umfassen dabei die Produktion, die Logistik oder die Gesundheitsbranche [Mey18]. Aber auch im privaten Bereich verbreiten sich Alltagsgegenstände, die durch Sensoren oder Vernetzung neue Fähigkeiten erhalten. So können zum Beispiel Lichtquellen, Jalousien oder die Heizung zentral, auch über Distanz, gesteuert werden [Har03]. Es gibt sogar Fußmatten, die registrieren können, wer auf ihnen steht und wie viel diese Person wiegt. Dadurch kann festgestellt werden, welche Personen zu Hause sind, wer nicht da ist und wer gerade zu Besuch ist [JLY04].

Im Gesundheitssektor hat das IoT das Potenzial, in vielen medizinischen Anwendungen die Versorgung zu verbessern. Ein paar Beispiele sind die Fernüberwachung des Gesundheitszustands, regelmäßige Überwachung von chronischen Krankheiten von zu Hause aus, Unterstützung der Selbstständigkeit in der Altenpflege und Fitnessprogramme [IKK+15]. Die Möglichkeiten des IoT werden hierbei durch den Einsatz mobiler Endgeräte erweitert [Ded18]. Dazu zählen neben Smartphones auch Wearables. Dies sind elektronische Geräte, die man am Körper trägt, zum Beispiel Smart Watches oder Smart Bands. Durch Wearables können regelmäßig Daten über die benutzende Person gesammelt werden. Es können zum Beispiel Vitalparameter des Körpers, körperliche Aktivität, Verhalten und andere kritische Parameter, die sich auf die Qualität des täglichen Lebens auswirken, erfasst werden [HYM14]. Einen Gewinn bringt es dabei, nicht nur auf die erfassten Daten der einzelnen Smart Things zugreifen zu können, sondern diese auch noch beliebig miteinander verknüpfen und weiter anreichern zu können. Die Verknüpfung mit modernen Methoden der Datenanalyse ermöglicht es so, neue Entscheidungsgrundlagen zu schaffen. Hierbei kommen verschiedene Techniken zum Einsatz, zum Beispiel Ontologien [GZS18; KNL+18], Informationsintegration [Eck11], Datenfusion [AMK+17; BSU18] oder Aggregation [RV06].

Die verschiedenen Sensoren innerhalb des IoT generieren kontinuierlich große Datenmengen. Mit diesen Daten sind genaue Modelle der Realität möglich. So können zum Beispiel bei medizinischen Analysen bisher unberücksichtigte Informationen über die Patienten[1] miteinbezogen werden. Dabei kann es sich unter anderem um die Ernährungsweise (durch den intelligenten Kühlschrank übermittelte Daten), das individuelle Stressniveau (erfasst durch Wearables) und andere Lebensgewohnheiten (erfasst durch Wearables und andere Sensoren im Haus) handeln. Auch genetische Daten können das ganzheitliche Bild des Patienten mitformen. All diese Daten beschreiben den Ausgangspunkt des Patienten für die Gesundheitsgestaltung detailliert [Krü19]. Jedoch stellt diese große Datenmenge für IoT-Anwendungen auch eine Herausforderung dar, da diese effizient mit einer begrenzten Anzahl an Ressourcen verarbeitbar sein soll [ZRL+17]. Mit diesen Herausforderungen beschäftigt sich das Forschungsgebiet Big Data, welches in Abschnitt 3.1 genauer definiert wird. Zudem gibt es Fälle, in denen nicht alle Daten allen Anwendungen bereitgestellt werden dürfen. Zum Beispiel soll die Versicherung keinen Zugriff auf die Ernährungsweise und Lebensgewohnheiten haben. Auch der Zugriff auf genetische Daten soll eingeschränkt werden. Dies dient dem Schutz der Privatsphäre der betroffenen Person und soll die freie Entwicklung und Entfaltung der Individuen ermöglichen. Geschützt wird die Privatsphäre durch das allgemeine Persönlichkeitsrecht, welches aus Art. 2 Abs. 1 Grundgesetz (freie Entfaltung der Persönlichkeit) in Verbindung mit Art. 1 Abs. 1 Grundgesetz (Menschenwürde) abgeleitet wird [Jur]. Zum Schutz der Privatsphäre bei der Verarbeitung personenbezogener Daten werden Datenschutzgesetze wie die Datenschutz-Grundverordnung (DSGVO) erlassen [ER16].

Ein möglicher Lösungsansatz, um die Herausforderungen von Big Data zu bewältigen und den Schutz der Privatsphäre zu gewährleisten ist, eine Datenreduktion durchzuführen. Eine Möglichkeit ist die Daten in einem sogenannten Data Lake zu halten. Das Konzept eines Data Lake wird in Abschnitt 3.2 erläutert. Der Data Lake kann in getrennte Zonen, das heißt in Unterbereichen, unterteilt werden. In diesen Zonen können die Daten bedarfsgerecht für verschiedene Anwendungen aufbereitet werden. So können in jeder Zone unterschiedliche Techniken der Datenreduktion zum Einsatz kommen, zum Beispiel Aggregation über ein Zeitfenster (z. B. Mittelwertbildung), Reduktion der zeitlichen Auflösung oder der betrachteten Merkmale oder Addition eines künstlichen Rauschens [SGG].

1.1 Ziele der Arbeit

Im Rahmen dieser Arbeit soll daher untersucht werden, wie eine Datenbereitstellungsplattform für datenintensive IoT-Anwendungen die Daten bedarfsgerecht bereitstellen kann. Bedarfsgerecht beinhaltet in diesem Kontext die benötigte Datenanreicherung, als auch die anschließende Datenreduktion. Zu diesem Zweck wird in dieser Arbeit RETORT vorgestellt, ein Privacy-Aware Data Annotation Framework for the Internet of Things. Eine konzeptuelle Vorarbeit für RETORT stammt von Stach et al. [SGG]. Diese entwickelten einen allgemeinen Entwurf einer Plattform, die datenschutzgerecht Datenanreicherungen durchführt. Dieser Entwurf wird innerhalb dieser Arbeit zu einer umsetzbaren Datenbereitstellungsplattform für datenintensive IoT-Anwendungen weiterentwickelt. Er wird konkretisiert und ergänzt, um die Herausforderungen der Verwaltung von Big Data sowie die Bedrohung der Privatsphäre durch die große Menge an Daten und Möglichkeiten

[1] In der folgenden Arbeit wird aus Gründen der besseren Lesbarkeit ausschließlich die männliche Form verwendet. Sie bezieht sich auf Personen aller Geschlechter.

der Verknüpfung dieser durch eine Datenreduktion zu lösen. Somit wird das Konzept von RETORT unter dem Aspekt der Verwaltung von Big Data und der Datenreduktion kritisch bewertet. Des Weiteren wird eine Umsetzungsstrategie des Konzepts von RETORT präsentiert.

1.2 Aufbau der Arbeit

Zunächst wird in Kapitel 2 ein Anwendungsszenario vorgestellt. Anhand einer Reihe von möglichen Anwendungsfällen der Benutzung von RETORT innerhalb des Anwendungsszenarios werden Anforderungen an das System ausgearbeitet. Kapitel 3 erklärt daraufhin Grundlagen, die zum Verständnis der restlichen Arbeit notwendig sind. Anschließend werden in Kapitel 4 verwandte Arbeiten vorgestellt und mit RETORT verglichen. In Kapitel 5 wird ein Konzept der erweiterten Architektur von RETORT erarbeitet. Kapitel 6 stellt basierend darauf einen Ansatz zur Implementierung vor, welcher prototypisch umgesetzt wird. Die Beschreibung des Prototyps erfolgt in Kapitel 7. In Kapitel 8 wird bewertet, inwiefern das System die in Kapitel 2 vorgestellten Anforderungen erfüllt. Schlussendlich werden in Kapitel 9 die Ergebnisse der Arbeit zusammengefasst und es wird ein Ausblick auf mögliche zukünftige Arbeiten gegeben.

Kapitel 2

Anwendungsszenario und Anforderungen an RETORT

Die Ergebnisse dieser Arbeit können in den verschiedensten Anwendungsszenarien des breiten Spektrums des Internet of Things angewendet werden. Ein Teilbereich hiervon ist der Fitness- und Healthcare-Bereich. In diesem Kapitel wird als Anwendungsszenario eine Fitnessakte vorgestellt. In einer Fitnessakte können Trainings- und Fitnessdaten gespeichert werden. Eine genaue Beschreibung erfolgt in Abschnitt 2.1. Ausgehend davon ergeben sich verschiedene Anwendungsfälle. In Abschnitt 2.2 wird beispielhaft ein Auszug der Anwendungsfälle erläutert. Anhand dieser Anwendungsfälle wird festgestellt, welche Anforderungen an RETORT gestellt werden müssen, um diesen und weiteren ähnlichen Anwendungsszenarien zu genügen. Die Anforderungen werden in Abschnitt 2.3 behandelt.

2.1 Fitnessakte als Teil der elektronischen Gesundheitsakte

Im Zuge des Patient-Empowerments soll das Rollenverständnis zwischen Arzt und Patient zu einer aktiven Partnerschaft verschoben werden. Dies bedeutet, dass der Patient durch Informationen, Mitwirkung und Mitentscheidung einen aktiveren Part im Behandlungsverlauf einnimmt. Dies soll den Erfolg einer Behandlung steigern [Grä04].

Das Patient-Empowerment soll durch die von Trill und Arendt [TA09] beschriebene webbasierte Fitnessakte als Element der elektronischen Gesundheitsakte (eGa) in Deutschland verstärkt werden. Eine webbasierte Gesundheitsakte wird dabei nach Sittig [Sit02] als eine über das Internet zugängliche Anwendung zur Erstellung, Betrachtung und Pflege einer Sammlung gesundheitlicher Daten der betroffenen Person definiert. In der eGa sollen dabei alle medizinischen Dokumente, Befunde und Informationen zusammengetragen und geordnet werden. Es können zum Beispiel Gesundheitsdaten wie Informationen über aktuelle und vergangene Erkrankungen, Diagnosen, Therapien und deren Verlauf enthalten sein. Auch der Impfstatus oder die aktuelle Medikation können Teil der eGa sein. Um Risiken für Krankheiten zu verhindern und zu vermeiden, können auch Präventionsdaten

enthalten sein, zum Beispiel ein Ernährungstagebuch oder Daten über das Stressniveau. Das Ziel ist eine lebenslange Dokumentation der medizinischen Historie ergänzt durch aktuelle Gesundheits- und Präventionsdaten, die immer und von überall aus (z. B. auch aus dem Ausland) erreichbar ist.

Durch die Möglichkeit der Zugriffsgewährung für Trainer, Ärzte, Physiotherapeuten und andere betroffenen Bereiche, besteht die Möglichkeit einer vernetzten und abgestimmten Betreuung durch die verschiedenen Berufsgruppen. So können die jeweiligen Erkenntnisse gebündelt und Behandlungen angepasst werden [TA09].

Einen Teil der eGa können auch Trainings- und Fitnessdaten bilden, erfasst innerhalb einer Fitnessakte. Durch regelmäßige sportliche Aktivität kann ein direkter positiver Einfluss auf den Gesundheitszustand ausgeübt werden. Mithilfe eines Trainingstagebuchs besteht dabei zum Beispiel die Möglichkeit, Fitnessdaten zu dokumentieren [TA09]. Des Weiteren gibt es innerhalb des IoT immer mehr Wearables (elektronische Geräte, die man am Körper trägt) oder Anwendungen, die fitness- und gesundheitsrelevante Daten aufzeichnen können. Zum Beispiel können verschiedene Sensoren laufend den Puls oder Blutzuckerspiegel erfassen. Fitnesstracker können die Aktivitäten des Benutzers verfolgen [HYM14]. Diese Daten können der eGa hinzugefügt werden. Innerhalb dieser Arbeit wird das Augenmerk auf die Fitnessakte als Unterkategorie der elektronische Gesundheitsakte gelegt. Im Folgenden wird die Funktion einer Fitnessakte genauer beschrieben.

Die in einer Fitnessakte enthaltenen Daten stammen aus unterschiedlichen Quellen und können strukturiert, teilstrukturiert oder unstrukturiert sein. Zum Beispiel können Vitalwerte, Bewegungsdaten, Bilder und Anmerkungen des Nutzers enthalten sein. Die Daten sind somit heterogen. Durch die kontinuierlich und mit kurzen Abständen anfallenden Datensätze wird eine umfangreiche Datenmenge gesammelt. Somit handelt es sich bei den enthaltenen Daten um Big Data. Big Data und dessen Eigenschaften werden in Abschnitt 3.1 definiert.

Über die Daten innerhalb der Fitnessakte hat die betroffene Person alleinige Verfügungsgewalt. Sie kann selber Daten zur Fitnessakte hinzufügen oder löschen. So kann sie eigenständig durchgeführte Trainingseinheiten protokollieren. Des Weiteren kann sie außenstehenden Personen die Berechtigung zur Dateneinsicht, sowie die Berechtigung zur Dateneingabe erteilen und auch wieder entziehen. Somit können außenstehende Personen auch Daten einsehen oder einfügen. Dadurch können zum Beispiel Wettkampferfolge durch die Veranstalter eingetragen werden oder Therapien auf den ganzheitlichen gesundheitlichen Zustand der Person abgestimmt werden. Außenstehende Personen sollen jedoch nicht Zugriff auf alle in der Akte gespeicherten Daten haben. Zum Beispiel sollen Wettkampfveranstalter keinen Zugriff auf die Trainings- und Gesundheitsdaten der Teilnehmenden haben. Die Trainer sollen aber wiederum die Trainingsdaten sehen können. Zur Berechtigungsvergabe ist daher ein Zugriffskontrollsystem notwendig. Verschiedene Möglichkeiten werden in der Arbeit von Sahafizadeh und Parsa [SP10] behandelt.

Durch die Berechtigung zur Dateneinsicht können außenstehende Personen auch Datenanalysen durchführen. Diese Möglichkeit stellt aber auch eine Bedrohung der Privatsphäre dar. Anhand eines Beispiels wird dieses Problem im Folgenden illustriert: In der Fitnessakte enthaltene Daten können zum Beispiel Fotos sein. Mithilfe dieser kann das gegessene Essen protokolliert werden. Auch können mithilfe eines Computers Vision Systems die einzelnen Lebensmittel identifiziert werden und die Kalorien aus einem einzelnen Lebensmittelbild geschätzt werden. Somit wird das Führen eines Ernährungstagebuchs weniger zeitaufwändig und einfacher [BJM+15]. Die enthaltenen Metadaten eines Fotos sind zum Beispiel der Zeitpunkt der Aufnahme, sowie die Koordinaten des Aufnahmeortes. Außerdem können die Namen der abgebildeten Personen angefügt sein. Durch diese

Metadaten können die Fotos entsprechend ihres Aufnahmezeitpunktes oder entsprechend der auf ihnen abgebildeten Personen angeordnet werden. Aufgrund der großen Datenmenge, sowie der vielen Metadaten eines Dateneintrags, können viele Informationen über den Benutzer abgeleitet werden, unter anderem, welche Person zu welchem Zeitpunkt an welchem Ort war. Für die oben genannten Anwendungen der Anordnung der Fotos wird aber jeweils nur ein Teil der Daten benötigt. Um dieses Problem zu lösen, soll zum Schutz der Privatsphäre bei der Datenanalyse durch außenstehende Personen die bereitgestellte Datenmenge reduziert werden. Dies kann zum Beispiel durch eine Reduktion des bereitgestellten Datenvolumens oder eine Reduktion der Anzahl an bereitgestellten Attributen der Daten geschehen.

Der Schwerpunkt dieser Arbeit liegt hauptsächlich auf der Datenreduktion und bedarfsgerechten Bereitstellung von Daten.

2.2 Anwendungsfälle

Die in Abschnitt 2.1 beschriebene Fitnessakte soll durch eine Plattform umgesetzt werden. Dazu muss die Plattform die im Folgenden beschriebenen Anwendungsfälle unterstützen.

Es gibt unterschiedliche Personengruppen, die die in Abschnitt 2.1 beschriebene Fitnessakte verwenden. Dabei handelt es sich zum Beispiel um die Personen, deren Daten in ihr gespeichert sind (im Folgenden betroffene Person), aber auch um Ärzte, Trainer oder Wissenschaftler (im Folgenden Datenabnehmer). Neben diesen beispielhaften Akteuren kann es natürlich noch weitere Akteure geben.

Anwendungsfall 1 Eine betroffene Person möchte alle über sie gespeicherten Daten einsehen. Hierzu müssen ihr alle Daten bereitgestellt werden, die sie betreffen. Diese Daten müssen für die Anzeige aufbereitet sein.

Anwendungsfall 2 Die Trainerin der betroffenen Person möchte die Fitness- und Trainingsdaten der betroffenen Person einsehen. Hierzu muss ein Auszug der über die betroffene Person gespeicherten Daten bereitgestellt werden. Diese Daten müssen auch für die Anzeige aufbereitet sein.

Anwendungsfall 3 Eine Wissenschaftlerin des Instituts für Präventivmedizin möchte basierend auf den Daten aller Benutzer der Fitnessakte eine Studie durchführen. Hierzu müssen die Daten anonymisiert oder pseudonymisiert werden. Die Daten müssen nicht aufbereitet werden, da die Wissenschaftlerin das nötige Wissen besitzt, die Daten zu verarbeiten und unterschiedliche Analysen durchführen möchte, für die die Daten in unterschiedlichen Formaten vorliegen müssen.

Jede dieser Gruppen hat unterschiedliche Bedürfnisse bei der Datenbereitstellung, zum Beispiel das Format der Daten betreffend. Auch unterscheiden sie sich in der Datenmenge, auf die sie Zugriff haben.

Es können bei der Verwendung der Fitnessakte auch neue Bedürfnisse entstehen. So benötigt die Trainerin zum Beispiel neben der Geschwindigkeit eines Laufes nun auch die zurückgelegten Höhenmeter, um die erbrachte Leistung einordnen zu können. Die Erstellung solcher neuer An-

wendungsfälle soll erleichtert werden, indem sie auf Basis bereits bestehender Anwendungsfälle definiert werden können. So müssen die bereits bestehenden Konfigurationen nur um die neuen Bedürfnisse erweitert werden.

Anwendungsfall 4 Die Trainerin möchte ihre Art der Datenbereitstellung anpassen. Dazu kann sie auf bereits definierte Anwendungsfälle zurückgreifen.

Jede Gruppe hat auch unterschiedliche Berechtigungen in Bezug auf die Daten. Je nachdem für welchen Zweck die Daten benötigt werden, soll eine unterschiedliche Menge an Daten bereitgestellt werden. Dies wird in Anwendungsfall 1 bis Anwendungsfall 3 beschrieben: Eine betroffene Person kann alle über sie gespeicherten Daten einsehen, ein Trainer einen Auszug der Daten über den jeweiligen Sportler. Ein Wissenschaftler kann Daten mehrerer Personen einsehen, jedoch nur anonymisiert oder pseudonymisiert. Dies dient unter anderem dem Schutz der Privatsphäre der betroffenen Personen.

Eine Datenreduktion kann durch eine Reduktion des bereitgestellten Datenvolumens geschehen. Das Datenvolumen kann reduziert werden, indem das bereitgestellte Zeitintervall der Daten beschränkt oder die zeitliche Auflösung reduziert werden. Auch das Herausfiltern bestimmter Tupel bzw. die Beschränkung der bereitgestellten Attribute kann das Datenvolumen reduzieren. Indem bestimmte Daten weggelassen werden, zum Beispiel bei einem Lauf die GPS Koordinaten der gelaufenen Strecke, kann die Anzahl an bereitgestellten Attributen der Daten reduziert werden. Die Daten können jedoch auch aggregiert und zu neuen Kenngrößen zusammengefasst werden. Dies kann durch statistische Methoden, wie den Mittelwert oder die Summe, geschehen. Aber auch andere Kenngrößen, die zum Beispiel das Verhältnis zweier Größen repräsentieren oder Aussagen, ob ein definierter Schwellenwert überschritten ist, können verwendet werden. Zum Beispiel kann aus Gewicht und Körpergröße der Body Mass Index (BMI) berechnet werden. Dabei handelt es sich um eine Kenngröße, die das Verhältnis zweier Größen, hier Gewicht und Körpergröße wiedergibt. So kann die Aussagekraft der Daten reduziert werden.

Diese verschiedenen Arten der Datenreduktion sind nicht pauschal auf jeden Anwendungsfall anwendbar. Es muss pro Anwendungsfall eine Art der Datenreduktion festgelegt werden. Dabei sollen alle benötigten Daten bereitgestellt werden, jedoch nicht mehr.

Anwendungsfall 5 Der Trainerin wird das Gewicht einer Sportlerin bereitgestellt. Die Sportlerin misst täglich ihr Gewicht. Der Trainerin wird jedoch statt den täglichen Werten nur der jeweilige Wochendurchschnitt bereitgestellt.

Aus diesen Anwendungsfällen ergeben sich eine Reihe von Aufgaben, die die Technologie zur Umsetzung der Fitnessakte erfüllen muss. Im Folgenden sind beispielhaft sieben beschrieben.

Ag1 Die Daten sollen für unterschiedliche Benutzergruppen angereichert und bereitgestellt werden. (Aufgrund Anwendungsfall 1 bis Anwendungsfall 3)

Ag2 Es können neue Anwendungsfälle definiert werden, die nicht von vornherein bekannt waren. (Aufgrund Anwendungsfall 4)

Ag3 Es soll eine Datenreduktion durchgeführt werden. (Aufgrund Anwendungsfall 5)

Ag4 Die bereitgestellten Daten sollen auf das für den Anwendungsfall notwendige Minimum reduziert werden. (Aufgrund Anwendungsfall 5)

Ag5 Die Datenreduktion und Datenbereitstellung soll an einen definierten Zweck gebunden sein. (Aufgrund Anwendungsfall 1 bis Anwendungsfall 3 und Anwendungsfall 5)

Ag6 Die Art der Datenreduktion soll von der Anwendung abhängig sein. (Aufgrund Anwendungsfall 5)

Ag7 Es sollen Templates zur Verfügung stehen, die für die Definition neuer Anwendungsfälle verwendet werden können. Diese sollen auf bereits definierten Anwendungsfällen basieren. So wird die Wiederverwendbarkeit bereits definierter Anwendungsfälle unterstützt. Diese Templates sollen die Kategorie der Anwendungsfälle sowie der in ihnen bereitgestellten Daten berücksichtigen. (Aufgrund Anwendungsfall 4)

Im Rahmen dieser Bachelorarbeit wird mit RETORT eine Plattform entwickelt, die die in diesem Abschnitt genannten Anwendungsfälle und Aufgaben erfüllt.

2.3 Anforderungen an RETORT

Aus den in Abschnitt 2.2 beschriebenen Anwendungsfällen und Aufgaben, die die Technologie zur Umsetzung der Fitnessakte erfüllen muss, lassen sich einige Anforderungen an RETORT ableiten. Im Folgenden werden sechs Anforderungen an RETORT definiert.

A1 *Flexibilität*: Die Datenbereitstellung soll an verschiedenen Nutzerbedürfnisse anpassbar sein. Je nachdem für welche Benutzergruppe die Daten bereitgestellt werden, kann sich zum Beispiel die Art oder der Umfang unterscheiden. (Abgeleitet aus Ag1)

A2 *Erweiterbarkeit*: Das System soll an neue Bedürfnisse anpassbar sein. So können zum Beispiel neue Anwendungsfälle definiert werden, die nicht von vorneherein bekannt waren. (Abgeleitet aus Ag2)

A3 *Einschränkung der Datenweitergabe*: Die Datenweitergabe soll eingeschränkt sein. Sie muss an einen klar definierten Zweck gebunden sein. (Abgeleitet aus Ag5)

A4 *Anwendungsbezogene Reduktion*: Die Art der Reduktion soll an den Anwendungsfall angepasst sein. (Abgeleitet aus Ag6)

A5 *Reduktion der Aussagekraft der Daten*: Die Aussagekraft der bereitgestellten Daten soll reduziert werden. (Abgeleitet aus Ag3 und Ag4)

A6 *Kategorie der Daten und des Anwendungsfalles berücksichtigen*: Die Kategorie der zu reduzierenden Daten und des entsprechenden Anwendungsfalls sollen bei der Datenreduktion berücksichtigt werden. (Abgeleitet aus Ag7)

KAPITEL 3

Grundlagen

In diesem Kapitel wird ein Überblick über die theoretischen Grundlagen gegeben, die zum Verständnis der Arbeit benötigt werden. Durch das IoT stehen Anwendungen große Mengen an heterogene Daten zur Verfügung. Diese Art der Daten wird als Big Data bezeichnet. In Abschnitt 3.1 wird der Begriff ‚Big Data' definiert. Um Big Data speichern und verarbeiten zu können, wurde das Konzept eines Data Lake entwickelt. Dieses wird in Abschnitt 3.2 vorgestellt. Anschließend werden in Abschnitt 3.3 relationale und NoSQL Datenbanken vorgestellt, da diese häufig die Grundlage der Datenverwaltung eines Data Lake bilden. Um das potenzielle Wissen, welches in einem Data Lake gespeichert ist, darstellen zu können, wird in Abschnitt 3.4 das Konzept einer Ontologie erklärt, sowie das Resource Description Framework (RDF) als formale Sprache für die Modellierung einer Ontologie vorgestellt.

3.1 Big Data

Durch die Verbreitung von Wearables (elektronische Geräte, die man am Körper trägt) und anderer Sensoren können regelmäßig Daten über die benutzende Person gesammelt werden. Dadurch wird kontinuierlich eine große Datenmenge generiert. Des Weiteren können die in einer Fitnessakte enthaltenen Daten, wie in Abschnitt 2.1 eingeführt, aus unterschiedlichen Quellen stammen und strukturiert, semistrukturiert oder unstrukturiert sein. Die Daten haben also eine heterogene Struktur. Dies sind zwei Eigenschaften von Big Data. Im Folgenden wird deshalb der Begriff Big Data genauer definiert.

Der Begriff ‚Big Data' bezeichnet Datensätze außergewöhnlicher Größe mit unterschiedlichen und komplizierten Strukturen. Dabei ist nicht nur der Umfang des Datenbestandes entscheidend, sondern auch die Kombination verschiedener technischer Herausforderungen der Datenverarbeitung. Es kann schwer sein, diese Daten mit herkömmlichen Methoden zu speichern, zu analysieren und zu visualisieren [SS13].

Die verbreitetste Definition des Begriffs Big Data stammt von Gartner Inc.. Diese definieren Big Data als ein umfangreiches, schnell zu verarbeitendes und/oder vielfältiges Informationskapital, das nach kostengünstigen, innovativen Formen der Informationsverarbeitung verlangt, um einen besseren Einblick und eine bessere Grundlage für Entscheidungsfindungen und Prozessautomatisierungen zu ermöglichen [Gar; MK16].

Diese Definition basiert auf den Eigenschaften *Volume*, *Velocity*, *Variety*, *Veracity* und *Value* der Big Data. Diese gehen auf Laney zurück, der im Jahr 2001 die Herausforderungen des Datenwachstums beschrieb [Lan01]. Durch diese Eigenschaften und die damit verbundenen Herausforderungen wird Big Data oft charakterisiert. Hsu et al. [HCH17] und Steinebach et al. [SWH+15] beschreiben die Eigenschaften von Big Data wie folgt.

Volume beschreibt die umfangreiche Größe des Datenbestandes. Diese Menge an Daten ist mit herkömmlichen Methoden kaum zu verarbeiten.

Velocity beschreibt zum einen die hohe Geschwindigkeit, mit der die Daten generiert werden, also die Dateneingangsgeschwindigkeit. Zum anderen beschreibt Velocity die Geschwindigkeit, mit der die Daten verarbeitet werden müssen (Datenausgangsgeschwindigkeit), um zum Beispiel Echtzeitverarbeitung zu ermöglichen.

Variety beschreibt die große Vielfalt der Datenformate durch die unterschiedliche Herkunft und die verschiedenen Typen der Daten. Es handelt sich dabei unter anderem um strukturierte, semistrukturierte und unstrukturierte Daten.

Veracity steht für die unterschiedliche Vertrauenswürdigkeit der Daten aufgrund der unterschiedlichen Herkünfte der Daten.

Value beschreibt den potenziellen Wert der Daten, wenn sie verarbeitet wurden und betont, dass letztendlich immer eine Wertschöpfung aus den Daten beabsichtigt ist.

3.2 Data Lake

Um Big Data zu speichern und zu verwalten, entstand das Konzept des Data Lake. Ein Data Lake ist ein zentraler Ort, an dem alle Daten unabhängig von der Quelle, der Struktur, der Größe, der Datenrate oder der Importmethode gespeichert werden können [Mat17; Sha18]. Die Daten werden im Rohformat gespeichert, so dass nicht die Notwendigkeit besteht, alle Anwendungsfälle vorab zu definieren. Es können aber ebenso vorverarbeitete Daten gespeichert werden. Somit vereint ein Data Lake strukturierte, semistrukturierte und nicht strukturierte Daten [GGH+20a]. Bei strukturierten Daten gibt es ein Modell, welches die Struktur der Daten beschreibt. Ein Beispiel sind die gespeicherten Patientenstammdaten. Die von einem Schrittzähler gelieferten Messdaten sind ein Beispiel für semistrukturierte Daten. Die Daten enthalten die Strukturinformationen selbst, z. B. als JSON-Objekt [ABS14]. Ein Beispiel für unstrukturierte Daten ist der Inhalt von Bildern, Videos oder Textdokumenten. Diese liegen nicht in einer formalisierten Struktur vor und gewinnen erst eine Struktur, indem sie in ein Schema gebracht werden [LL17]. Durch die große Flexibilität des Data Lake bezüglich der in ihm gespeicherten Daten und der Möglichkeit die Daten zu speichern, ohne sie zuerst strukturieren zu müssen, können verschiedene analytische Anwendungsfälle unterschiedlicher Nutzergruppen auf den Daten ausgeführt werden [GGH+20a]. So können zum einen regelmäßig aktualisierte Berichte und Dashboards erstellt werden, um die Mitglieder einer Organisation zu informieren. Es können aber auch Analysen der Daten durchgeführt werden und so neue Erkenntnisse erlangt werden [Rus17]. Diese Flexibilität unterstützt auch, dass sich die Struktur der Daten über die Zeit hinweg verändern kann, zum Beispiel bei der Verwendung eines Brustgurtes eines anderen Herstellers. Somit ist ein Data Lake eine skalierbare Datenmanagementplattform für analytische Zwecke, die die Daten in Rohform speichert, um Informationsverlust zu vermeiden [GGH+20a].

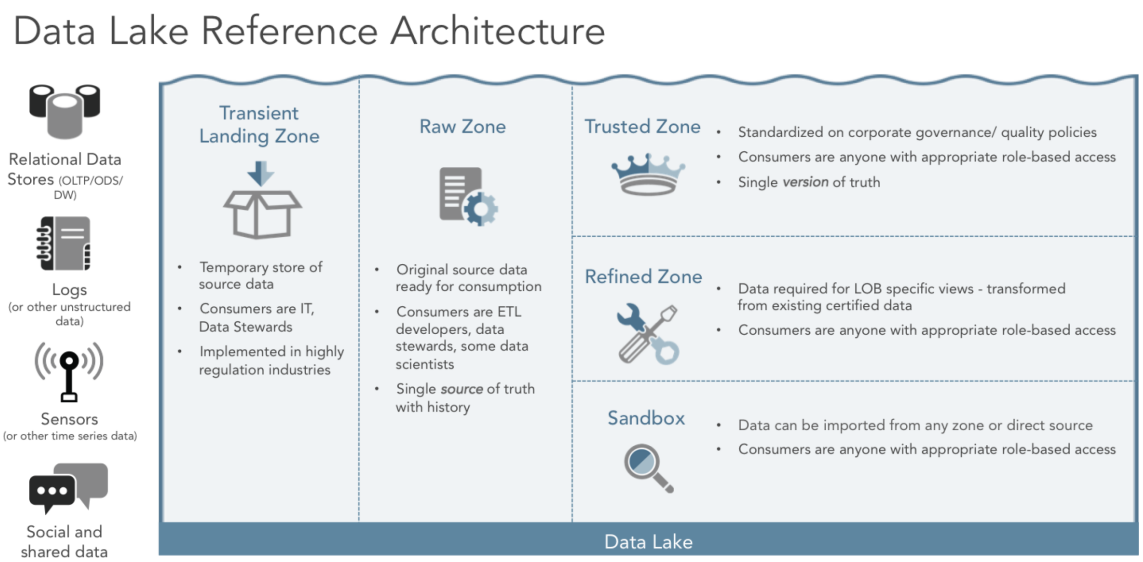


Abbildung 3.1: Zaloni Data Lake Referenz Architektur nach Sharma [Sha18].

Madsen [Mad15] beschreibt die ineffiziente Wiederholung von Verarbeitungsschritten, da viele Anwendungen denselben Verarbeitungsschritt der Rohdaten wiederholt durchführen. Um diese doppelte Rechenarbeit zu vermeiden, werden neben den unverarbeiteten Daten ebenfalls verschiedene Stadien der Verarbeitung der Daten in einem Data Lake gespeichert. Dazu werden Data Lakes in Zonen strukturiert. Die verschiedenen Zonen definieren dabei, welcher Grad der Verarbeitung in ihnen gespeichert ist [GGH+20b]. Sharma [Sha18] beschreibt eine mögliche Data Lake Architektur: die Zaloni Data Lake Referenz Architektur. Diese besteht aus vier Zonen und ist in Abbildung 3.1 dargestellt.

Die Definition eines Data Lake schreibt keine bestimmte Speichertechnologie vor. Er kann auf verschiedenen Technologien beruhen, zum Beispiel Hadoop, NoSQL oder relationale Datenbanken [it-]. Deshalb werden im nächsten Abschnitt verschiedene Datenbankmodelle vorgestellt.

3.3 Datenbankmodelle

In diesem Abschnitt werden verschiedene Datenbankmodelle zur Realisierung von Data Lakes vorgestellt. Zuerst wird auf relationale Datenbanken eingegangen (Abschnitt 3.3.1). Diese erwiesen sich jedoch durch ihr fest definiertes Schema als unzureichend im Umgang mit dem Umfang und der unterschiedlichen Struktur von Big Data. Deshalb wird nachfolgend auf NoSQL Datenbanken eingegangen (Abschnitt 3.3.2). Diese sind in der Lage Big Data zu verwalten [MH13]. Abschließend wird auf vier verschiedene Arten von NoSQL eingegangen: Schlüssel-Werte-Datenbanken, dokumentenorientierte Datenbanken, spaltenorientierte Datenbanken und Graphdatenbanken.

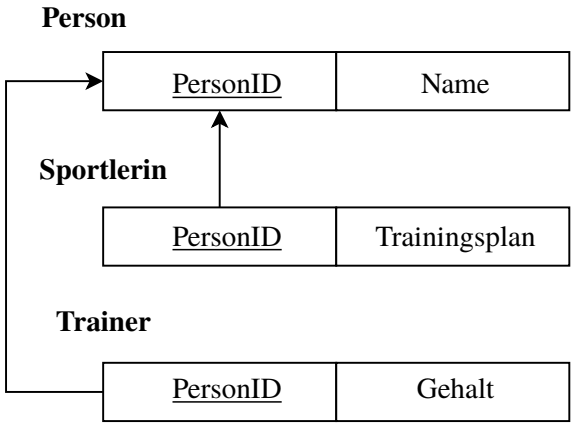


Abbildung 3.2: Relationenmodell eines Beispielsystems.

3.3.1 Relationale Datenbanken

Im Folgenden wird der Begriff der Relationalen Datenbank, wie von Unterstein und Matthiessen [UM12] beschrieben, erläutert. Relationale Datenbanken stellen Daten sowie Beziehungen zwischen Daten in tabellarischer Form dar. Die in einer Datenbank gesammelten Objekte werden durch Attribute beschrieben. Ein Tupel entspricht einer Menge von Attributen. In Abbildung 3.2 stellen zum Beispiel die Attribute *PersonID* und *Name* ein Tupel dar. Eine Menge von Tupeln mit denselben Attribut-Bezeichnern bildet eine Relation. Diese Menge an Tupeln muss einem Relationsschema entsprechen, welches durch eine Menge von Attribut-Bezeichnern mit ihren jeweiligen Wertebereichen gegeben ist.

Eine relationale Datenbank ist somit durch eine Menge von Relationen gegeben, die jeweils durch das entsprechende Relationsschema bestimmt sind. Dabei entspricht eine Relation einer Tabelle, ein Tupel einer Zeile einer Tabelle und ein Attribut einer Spalte einer Tabelle [MK16; UM12]. Das Relationsschema mit den Attribut-Bezeichnern und ihren jeweiligen Wertebereichen, sowie der Anzahl der Attribute ist vorab definiert und zur Laufzeit nur unter Aufwand anpassbar [Cat11].

Des Weiteren können durch Verknüpfungen (hierarchische) Beziehungen zwischen Relationen ausgedrückt werden [MK16]. Ein Beispiel ist in Abbildung 3.2 dargestellt. Es besteht aus Personen. Diese können Sportlerinnen oder Trainer sein. Eine Person hat eine ID und einen Namen. Eine Sportlerin hat noch zusätzlich einen Trainingsplan, ein Trainer ein Gehalt.

Eine relationale Datenbank eignet sich für Daten, die sich in Tabellen darstellen lassen. Auch ist ihr Schema unflexibel, also nur unter Aufwand änderbar. Durch ihr fest definiertes Schema erwiesen sich Relationale Datenbanken als unzureichend im Umgang mit dem Umfang und der unterschiedlichen Struktur von Big Data, speziell mit semistrukturierten und unstrukturierten Daten.

3.3.2 NoSQL Datenbanken

NoSQL Datenbanken sind eine Klasse an Datenbanken, die keinen relationalen Ansatz besitzen und in der Lage sind Big Data zu verarbeiten [MH13]. NoSQL bedeutet dabei ‚Not Only SQL'. Ein Grund für die Fähigkeit der Verarbeitung von Big Data ist, dass NoSQL-Datenbanken im Gegensatz

zu relationalen Datenbanken kein festes Schema besitzen, welches auf alle Datensätze angewandt wird. Sondern jeder Datensatz kann ein eigenes Schema besitzen, wodurch er sich selbst beschreibt. Somit ist das Schema der Daten flexibler und einfacher veränderbar. Dies unterstützt auch den Umgang mit der Heterogenität der Daten [GRR14].

Ein Vorteil von Data Lakes ist, dass sie beliebige Daten ohne vorherige Formatumwandlungen speichern können. Es ist also nicht nötig vor dem Laden der Daten ein Schema zu definieren. Die Definition eines Schemas erfolgt durch die Interpretation der Daten [it-]. Es werden also vor allem semistrukturierte und unstrukturierte Daten gespeichert. Um die Daten im Rohformat speichern zu können wird eine Flexibilität des Datenbankschemas benötigt. NoSQL bieten eine größere Flexibilität im Format und der Struktur der gespeicherten Daten als relationale Datenbanken. Sie besitzen kein festes Schema und ihre Datenstruktur ist nicht an eine Tabelle gebunden. Dies unterstützt die Speicherung von Daten ohne dessen Format ändern zu müssen.

Im Folgenden werden vier verschiedene Klassen von NoSQL Systemen, sowie ihre jeweiligen Anwendungsgebiete beschrieben. Zuerst Schlüssel-Werte-Datenbanken, dokumentenorientierte Datenbanken und spaltenorientierte Datenbanken. Eine Übersicht über die Eigenschaften dieser Klassen von NoSQL Systemen ist in Tabelle 3.2 dargestellt. Zum Schluss wird die Klasse der Graphdatenbanken vorgestellt.

Schlüssel-Werte-Datenbanken

Schlüssel-Werte-Datenbanken speichern Werte (engl. values) sowie für jeden Wert einen Schlüssel (engl. key), über den der Wert gefunden werden kann [Cat11]. Die Daten werden dabei als Schlüssel-Wert-Paare oder Hash-Tabelle gespeichert [GRR14].

Schlüssel-Werte-Datenbanken können mit sehr großen Mengen an Einträgen umgehen [OBLB17]. Sie eignen sich für die Echtzeitverarbeitung von Big Data, also allgemein Anwendungen, bei denen die Antworten innerhalb von Millisekunden benötigt werden [GRR14]. Sie sind generell eine gute Lösung, wenn nur ein Attribut für den Zugriff relevant ist und als Schlüssel verwendet werden kann [Cat11].

Ein Beispiel einer Schlüssel-Werte-Datenbank ist Redis[1].

Dokumentenorientierte Datenbanken

Dokumentenorientierte Datenbanken unterstützen durchsuchbare Schlüssel-Werte-Paare: Sie speichern Dokumente [Cat11]. Dokumente bezeichnen semistrukturierte Daten in Datensätzen [MK16]. Die Dokumente sowie ihr Inhalt werden indiziert und können somit auch abgefragt werden [Cat11]. Dokumentenorientierte Datenbanken sind also auf erster Ebene eine Art Schlüssel-Werte-Datenbanken, denn zu einem Schlüssel kann ein Datensatz (Dokument genannt) als Wert gespeichert werden. Auf der zweiten Ebene haben diese Dokumente aber eine eigene Struktur. Diese Struktur stellt eine Liste an Schlüssel-Werte-Paaren bzw. Attribut-Werte-Paaren dar, zum Beispiel im JSON-Format [MK16]. Für ein Dokument sind die Attribute nicht global definiert

[1] https://redis.io/

```json
{
  "ID": 1,
  "Vorname": "Carolin",
  "Nachname": "Schäfer",
  "Geburtsjahr": 1991,
  "Sportart": "Leichtathletik",
  "Disziplinen": [
    "Siebenkampf",
    "100m Hürden",
    "200m",
    "Weitsprung"
  ]
}
```

Listing 3.1: Beispiel eines JSON-Dokuments.

und die Namen der Attribute werden dynamisch definiert. Dadurch sind Dokumentenorientierte Datenbanken schemafrei. Das heißt, dass es nicht notwendig ist, vor dem Einfügen von Daten ein Schema zu definieren [MK16]. Alle Dokumente sind strukturell voneinander unabhängig und haben potenziell unterschiedliche und verschachtelte Attribute [GRR14]. Die Werte der Attribute können erneut Dokumente, Listen oder skalare Werte sein [Cat11]. Ein Beispiel eines JSON-Dokuments ist in Listing 3.1[2] zu sehen.

Dokumentenorientierte Datenbanken eignen sich für Anwendungen mit semistrukturierten Objekten, die anhand mehrerer Attribute abgefragt werden [Cat11]. Bei Schlüssel-Werte-Datenbanken können die gespeicherten Objekte nur aufgrund eines Attributes abgefragt werden. Zum Beispiel werden in einer Leichtathletikwettkampfverwaltung die Teilnehmenden in einer Dokumentenorientierten Datenbank gespeichert. Diese können nun basierend auf Name, Geburtsdatum oder der Disziplin, an der sie teilnehmen abgefragt werden. Auch sind sie ideal für Anwendungen, die eine Aggregation über eine Menge an Dokumenten durchführen [GRR14]. So kann in dem Leichtathletikwettkampf-Beispiel die minimale Laufzeit der Teilnehmer eines 100m Laufs berechnet werden und so der Gewinner ermittelt werden.

Ein Beispiel einer dokumentenorientierten Datenbank ist MongoDB[3].

Spaltenorientierte Datenbanken

Eine spaltenorientierte Datenbank verwendet ähnlich zu Relationalen Datenbanken ebenfalls die Begriffe Zeilen und Spalten. Die Daten werden im Gegensatz zu relationalen Datenbanken aber in unterschiedlich langen Zeilen, d. h. mit flexibler Anzahl an Spalten, gespeichert. Somit hat jede Zeile ein eigenes Set an Spalten. Zeilen werden dabei anhand eines Schlüssels identifiziert und

[2] Die Daten sind https://www.leichtathletik-datenbank.de/vereine/deutscher-leichtathletik-verband/suddeutschland/hessischer-leichtathletik-verband/rhein-main/frankfurt/lg-eintracht-frankfurt/athleten/65413-carolin-schfer (Stand 03.08.2020) entnommen.

[3] https://www.mongodb.com

Sportlerinnen

Schlüssel	Werte (Attributliste)		
sp01	Name	Sportart	
	Carolin Schäfer	Leichtathletik	
sp02	Name	Körpergröße	Augenfarbe
	Elisabeth Seitz[4]	163 cm	Blau
sp03	Name		
	Birgit Fischer		

Tabelle 3.1: Beispiel einer spaltenorientierten Datenbank mit der Spaltenfamilie Sportlerinnen.

sortiert [Bru12]. Eine Menge an Spalten, die oft zusammen verwendet werden, wird zu Spaltenfamilien zusammengefasst [V14]. Spaltenfamilien werden wiederum in Keyspaces zusammengefasst. Die Menge der Keyspaces bildet das Datenbanksystem [Bru12]. Ein Beispiel ist in Tabelle 3.1 dargestellt.

Spaltenorientierte Datenbanken sind geeignet für die Speicherung von Zeitfolgen durch ihre Struktur mit Zeilen und Spalten und der Eigenschaft, viel Schreib- und Leselast bei großer Datenmenge bewältigen zu können [Bru12].

Ein Beispiel einer spaltenorientierten Datenbank ist Apache Cassandra[5].

Dieses NoSQL Konzept ist nicht zu verwechseln mit einer besonderen Art relationaler Datenbanken, welche ihre Daten spaltenweise abspeichert, anstatt wie üblich zeilenweise.

In Tabelle 3.2 ist eine abschließende Übersicht über Schlüssel-Werte-Datenbanken, dokumentenorientierte Datenbanken und spaltenorientierte Datenbanken dargestellt. Die im folgenden Abschnitt vorgestellten Graphdatenbanken unterscheiden sich von diesen drei Arten an NoSQL Datenbanken dadurch, dass darauf ausgelegt sind Beziehungen zwischen Informationen darzustellen und zu verwalten. Dadurch eignen sie sich für andere Anwendungsfälle als die bisher vorgestellten Arten an NoSQL Datenbanken und werden deshalb separat behandelt.

Graphdatenbanken

Graphdatenbanken eignen sich, um Beziehungen zwischen Informationen darzustellen und zu verwalten. Dabei werden die Daten als Knoten eines gerichteten Graphen gespeichert. Die Beziehungen zwischen den Daten werden mithilfe der Kanten modelliert [AG08]. Als grundlegende Datenstruktur wird zum Beispiel ein Labeled Property Graph (LPG) verwendet. Dies ist ein Graph, bei dem die Knoten und Kanten jeweils einen Namen (engl. label) besitzen und Eigenschaften (engl. properties) aufweisen können. Die Eigenschaften werden als Attribut-Wert-Paare mit Attributnamen

[4] Die Daten sind https://bodysize.org/de/elisabeth-seitz/ (Stand 17.09.2020) entnommen.
[5] https://cassandra.apache.org/

	Schlüssel-Werte-Datenbank	Dokumentenorientierte Datenbank	Spaltenorientierte Datenbank
Datenstruktur	Schlüssel-Werte-Paare bzw. Hash-Tabellen	Dokumente z. B. JSON	Tabelle mit flexibler Anzahl Spalten pro Zeile und nicht festgelegtem Namen und Typ der Spalte
Index	Schlüssel	Gesamtes Dokument	Zeilenschlüssel
Unterstützt Hierarchische Daten	Nein	Ja	Nein
Aggregation	Eingeschränkt	Ja	Ja
Beispiel	Redis	MongoDB	Cassandra

Tabelle 3.2: Zusammenfassung der Eigenschaften von Schlüssel-Werte-Datenbanken, dokumentenorientierten Datenbanken und spaltenorientierten Datenbanken.

und zugehörigem Wert charakterisiert. Diese erlauben es die gespeicherten Informationen genauer zu charakterisieren [MK16]. Ein Beispiel eines LPGs ist in Abbildung 3.3 dargestellt. Der Graph beschreibt den Zusammenhang zwischen den Turnerinnen Kim Bui, Elisabeth Seitz und dem Verein MTV Stuttgart[6].

Ein Beispiel einer Graphdatenbank ist Neo4j[7].

3.4 Ontologie und RDF

Um das potenzielle Wissen, welches in einem Data Lake gespeichert ist, darstellen zu können, kann eine Ontologie verwendet werden.

Der Begriff der Ontologie stammt ursprünglich aus der Philosophie und steht dort für die Lehre vom Sein, genauer von den Bedingungen und Möglichkeiten des Seienden. Dieser Bereich ist eng verwandt mit der Erkenntnistheorie, die sich mit den Möglichkeiten und Grenzen des menschlichen Wahrnehmens und Erkennens auseinandersetzt [Hes02].

In der Informatik beschreibt der Begriff Ontologie eine formale Repräsentation von Wissen. Es existieren viele unterschiedliche Definitionen einer Ontologie, die kaum einheitlich sind [BHRZ10]. Die bekannteste Definition stammt von Gruber aus dem Jahr 1993 [Gru93]. Dieser definiert eine Ontologie als eine explizite Spezifikation einer Konzeptualisierung. Eine Konzeptualisierung ist dabei eine abstrakte, vereinfachte Sicht der Welt. Diese Sicht der Welt möchte man für einen

[6] Die Informationen sind https://www.dtb.de/turn-team-deutschland/team-geraetturnen/athletinnen-und-athleten/kim-bui/, https://de.wikipedia.org/wiki/Kim_Bui, https://www.dtb.de/turn-team-deutschland/team-geraetturnen/athletinnen-und-athleten/elisabeth-seitz/ und https://de.wikipedia.org/wiki/Elisabeth_Seitz (jeweils Stand 03.10.2020) entnommen.

[7] https://neo4j.com/

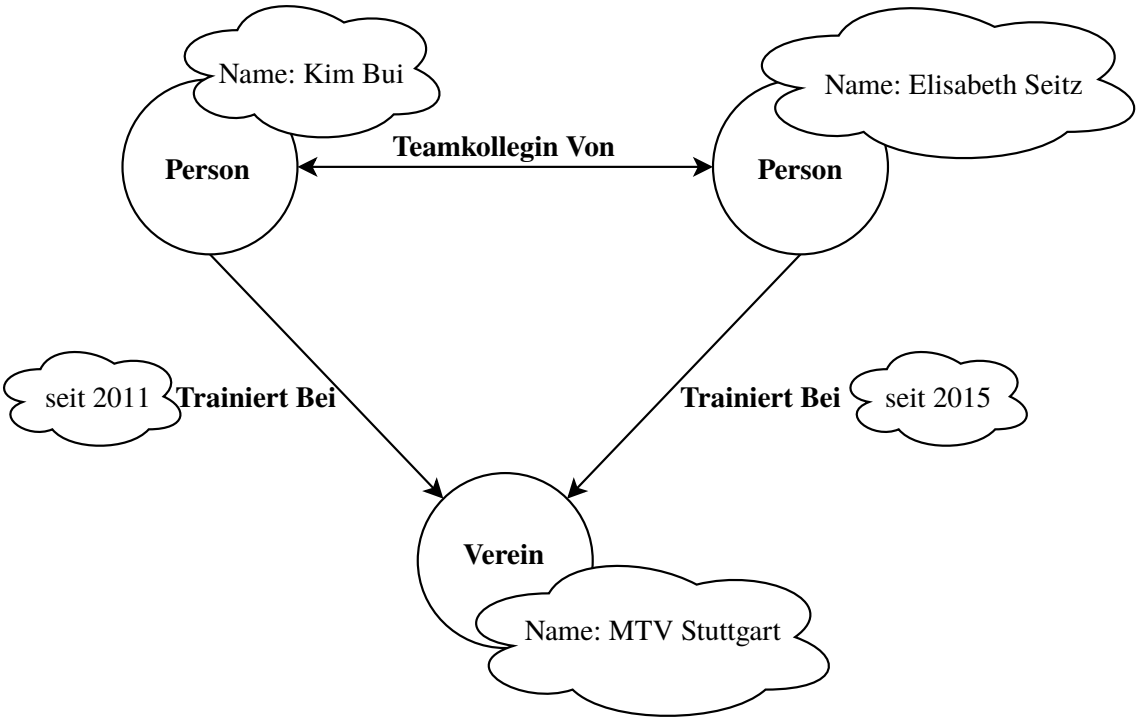


Abbildung 3.3: Ein Beispiel eines Labeled Property Graphen.

bestimmten Zweck darstellen. Dies geschieht durch die formale Definition einer Menge von Begriffen und der Beziehungen, die zwischen ihnen bestehen [BHL+12]. Eine Ontologie ist somit ein Netzwerk aus Daten, verknüpft durch Relationen. Dadurch entstehen semantischen Beschreibungen von Objekten oder Gesetzmäßigkeiten in einem Ausschnitt der Welt [BHRZ10]. Die Semantik von Daten und Informationen wird somit formalisiert und nutzbar gemacht. Eine Ontologie ermöglicht damit eine formale, digitale Weitergabe von Wissen zwischen Computeranwendungen sowie zwischen Computeranwendung und Menschen, aber auch zwischen Menschen [BHL+12].

Basierend auf Kuhlen et al. [KSS13] werden im Folgenden typische Bestandteile einer Ontologie dargestellt. Diese sind (hierarchisch geordnete) Klassen, Instanzen als konkrete Elemente der Klassen und Relationen als Beschreibung der Klassen sowie der Beziehungen zwischen ihnen. In Abbildung 3.4 ist eine schematische Darstellung einer Ontologie mit typischen Bestandteilen zum Thema Sport zu sehen. In der Abbildung sind die Klassen durch eine eckige Umrandung dargestellt. Die Klassen repräsentieren allgemeine Konzepte, die Objekte durch gemeinsame Eigenschaften bündeln sollen. Diese sind meist hierarchisch organisiert. Dabei erben Unterklassen alle Eigenschaften der Elternklasse. In dem betrachteten Beispiel sind die Klassen *Eishockeyverein* und *Fußballverein* Unterklassen der Klasse *Sportverein*. Diese sind durch eine *is_a* Relation verbunden, in der Abbildung durch durchgängige Pfeile dargestellt. Eine weitere Relation ist die *instance_of* Relation. Diese verbindet Instanzen und Klassen. Instanzen sind in der Abbildung durch eine ovale Umrandung dargestellt und repräsentieren konkrete Vertreter der Klassen. Im Beispiel ist *Borussia Mönchengladbach* eine Instanz der Klasse *Fußballverein*. Weitere Relationen, in der Abbildung als gestrichelte Pfeile dargestellt, können Klassen und Instanzen weiter spezifizieren. Zum Beispiel die Relation *ist_Stadion_von* bzw. *hat_Stadion* zwischen den Klassen *Fußballverein* und *Stadion*, sowie zwischen Instanzen dieser Klassen. Klassen und Instanzen können auch weitere Eigenschaf-

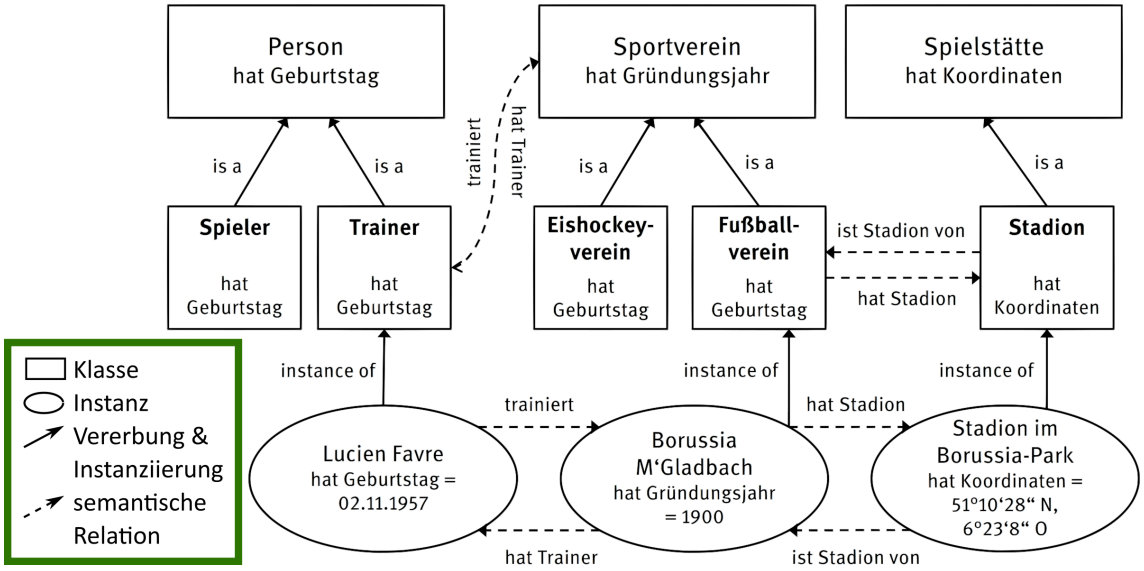


Abbildung 3.4: Schematische Darstellung einer Ontologie zum Thema Sport, bestehend aus Klassen und Klassenhierarchien, Instanzen, Relationen und Eigenschaften [KSS13].

ten besitzen, die sie genauer beschreiben. Dabei werden die Eigenschaften zwischen Eltern- und Kindklassen vererbt. Diese Eigenschaften sind auf Ebene der Instanzen mit konkreten Werten belegt. Beispielsweise hat die Klasse *Person*, sowie alle ihre Unterklassen und Instanzen, die Eigenschaft *hat_Geburtstag*. Diese wird in der Instanz *Lucien Favre* der Wert *02.11.1957* zugewiesen.

Eine Möglichkeit, eine Ontologie zu repräsentieren, ist die Darstellung anhand von Tripeln. Ein Beispiel hierzu ist das Resource Description Framework (RDF). Dieses wird im Folgenden nun genauer vorgestellt.

Das Resource Description Framework ist eine formale Sprache für die Modellierung einer Ontologie. Es stellt durch logische Aussagen eine strukturierte Beschreibung von Informationen über Ressourcen zur Verfügung [SR14]. RDF ist nicht nur für die Darstellung von Daten gedacht, sondern auch für die Kombination und Weiterverarbeitung der Informationen ohne Bedeutungsverlust [Hit08].

Das Datenmodell des RDF ist Graph-basiert. Es besteht aus einem Set an Tripeln, die jeweils aus einem Subjekt, einem Prädikat und einem Objekt bestehen. Ein RDF Tripel beschreibt den Zusammenhang zwischen dem Subjekt und dem Objekt. Dieser wird durch das Prädikat spezifiziert. In Abbildung 3.5 ist ein grundlegendes RDF Tripel mit den generischen Bezeichnern Subjekt, Prädikat und Objekt dargestellt. Die Elemente des Tripels werden mit eindeutigen Bezeichnern beschriftet, um zu vermeiden, dass die gleichen Bezeichner für unterschiedliche Ressourcen verwendet werden [Hit08]. Dadurch tritt ein Bezeichner maximal einmal pro RDF Graph auf. Als eindeutige Bezeichner werden Internationalized Resource Identifier (IRI) verwendet [DYB20]. Objekte können auch durch Literale beschrieben werden. Diese haben Datentypen, die den Bereich möglicher Werte definieren, z. B. Zahlen oder Zeichenketten [CWL14]. Enthält ein Objekt eine IRI, so stellt das Tripel die Beziehung zwischen den Ressourcen, die durch Subjekt und Objekt beschrieben werden, dar. Enthält das Objekt ein Literal, so stellt das Tripel ein Attribut der durch das Subjekt beschriebenen Ressource dar.

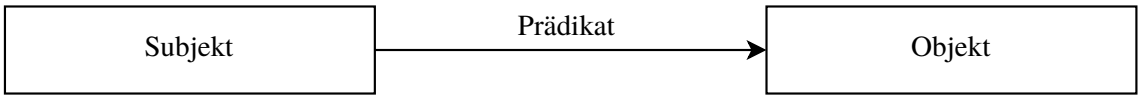


Abbildung 3.5: Ein grundlegendes RDF Tripel.

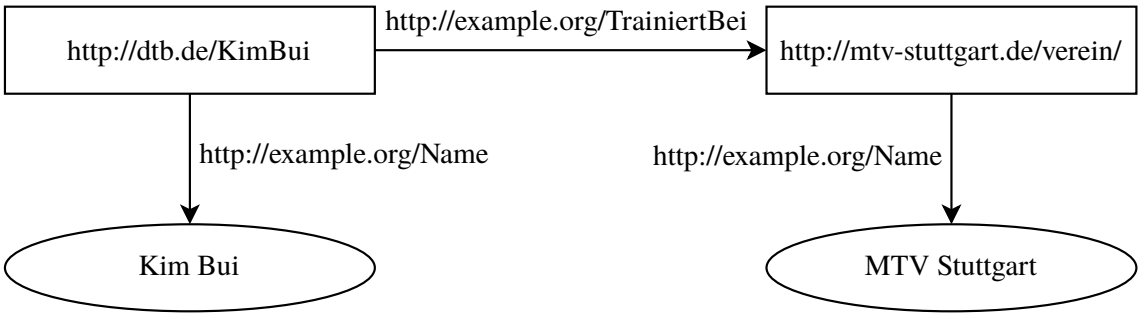


Abbildung 3.6: Ein RDF Graph zur Beschreibung der Beziehung zwischen der Turnerin Kim Bui und dem Verein MTV Stuttgart.

Das Set an Tripel wird RDF Graph genannt. Formell ist ein RDF Graph ein gerichteter Graph, bei dem jedes Tripel aus zwei Knoten und deren Beziehung zueinander besteht [CWL14]. In Abbildung 3.6 ist ein RDF Graph dargestellt[8]. Dieser beschreibt die Beziehung zwischen der Turnerin Kim Bui und dem Verein MTV Stuttgart. Dabei sind IRI-Bezeichner durch eine rechteckige Umrandung und Literale durch eine ovale Umrandung dargestellt.

Aufgrund ihrer Struktur können RDF Tripel in relationalen Datenbanken gespeichert werden. Diese Art der Speicherung ist allerdings nicht auf effiziente Abfragen des RDF Graphen ausgelegt. Triple Stores, auch RDF Stores genannt, sind darauf optimiert RDF Tripel zu speichern und zu verarbeiten. Sie ermöglichen eine effiziente Abfrage der Tripel [Rus01]. Triple Stores können als eine spezielle Art einer Graphdatenbank angesehen werden. Sie speichern die Tripel jedoch als unabhängige Fakten und nicht als zusammenhängende Struktur. Dies ist hinderlich bei der Traversierung von Beziehungen bzw. Teilgraphen. Triple Stores sind auf Abfragen nach direkte Zusammenhänge zwischen Subjekten und Objekten ausgelegt [Ngu18].

Für das Abfragen und die Bearbeitung von Daten, die im RDF Format gespeichert sind, wird eine RDF Abfragesprache verwendet. Diese unterstützt die Semantik und Inferenz der RDF Daten [HBEV04]. SPARQL, die von der W3C empfohlene RDF Abfragesprache, basiert auf einer Graphmustersuche [The13]. Abfragen von RDF Daten werden durchgeführt, indem Tripel-Muster, bestehend aus Subjekt, Prädikat und Objekt, angegeben werden. Diese stellen die Bedingung dar, die die Ergebnisse der Abfrage erfüllen müssen. Dabei können die Tripel Variablen enthalten. Anhand dieser Muster werden passende Tripel des RDF Graphen ermittelt. Das Ergebnis einer SPARQL Abfrage ist eine Menge an Ergebnissen, die aus einer Zuordnung der abgefragten Variablen zu den gespeicherten Daten besteht. SPARQL gilt als relationale Abfragesprache, da sie relationale und musterbasierte Operationen unterstützt [FLB+06].

[8] Die Daten sind https://www.dtb.de/turn-team-deutschland/team-geraetturnen/athletinnen-und-athleten/kim-bui/ (Stand 16.10.2020) entnommen.

KAPITEL 4

Verwandte Arbeiten

Dieses Kapitel bietet einen Überblick über verwandte Arbeiten zur Datenbereitstellung für datenintensive Internet of Things-Anwendungen. Entsprechend der Prozesskette der Datenverwaltung und Datenbereitstellung werden zuerst aus der Literatur bekannte Ansätze zur Datenverwaltung im IoT vorgestellt. Anschließend wird auf die Beschreibung der Semantik von Daten in Datenverwaltungsarchitekturen des IoT eingegangen. Am Ende des Kapitels werden bekannte Techniken zur Datenreduktion und Sicherung der Privatsphäre vorgestellt.

Datenverwaltungsarchitekturen für das Internet of Things. Aus der Vielzahl der Datenverwaltungsarchitekturen für das IoT [DRD+20], werden im Folgenden zwei Entwürfe beispielhaft genannt, da aus Platzgründen in dieser Arbeit nicht auf alle Architekturen eingegangen werden kann. Anschließend wird ein Modell eines Data Lake (siehe Abschnitt 3.2) vorgestellt, da es sich bei einem Data Lake um einen zentralen Ort handelt, an dem alle Daten unabhängig von der Quelle, der Struktur, der Größe, der Datenrate oder der Importmethode gespeichert werden können [Mat17; Sha18]. Diese Eigenschaften machen ihn zu einem passenden Modell für die Datenverwaltung und -verarbeitung von IoT-Daten.

Das Framework von Cai et al. [CXJV17] bietet einen funktionalen Rahmen, der die Bereiche Erfassung, Verwaltung, Entsorgung und Mining von IoT-Daten behandelt. Es besteht aus mehreren Modulen. Im Datenspeichermodul werden die unterschiedlichen Typen IoT Daten in unterschiedlichen Datenbank- oder Dateisystemen gespeichert, zum Beispiel im Hadoop Distributed File System, in Relationalen Datenbanken oder in NoSQL Datenbanken. Es gibt weitere Module zum Datenmanagement, zur Datenverarbeitung, zum Data Mining und zur Anwendungsoptimierung.

Ein anderer Entwurf stammt von Jiang et al. [JXC+14]. Diese schlagen ein Datenspeicher-Framework vor, das nicht nur eine effiziente Speicherung ermöglicht, sondern auch strukturierte und unstrukturierte Daten integriert. Es besteht aus einem Datenbank Modul und einer Dateiablage. Das Datenbankmodul dient der Speicherung von strukturierten Daten und besteht aus NoSQL und relationalen Datenbanken. Zur Speicherung von unstrukturierten Daten gibt es die Dateiablage, umgesetzt mit dem Hadoop Distributed File System. Die Datenverwaltung wird unterstützt durch ein Ressourcenkonfigurationsmodul. Dadurch können Datenressourcen und zugehörige Dienste basierend auf den Anforderungen der IoT-Anwendungen konfiguriert werden. Die Daten werden den IoT-Anwendungen durch das Service Module bereitgestellt. Dieses Modul ordnet die Metadaten

entsprechend der Konfiguration des Ressourcenkonfigurationsmoduls den in den Datenbanken und in der Dateiablage gespeicherten Datenentitäten und Dateien zu und stellt die Daten dann zur Verfügung.

Diese Ansätze überschneiden sich mit dem Konzept eines Data Lake, da sie zum Beispiel unterschiedliche Typen von Daten mit verschiedenen Strukturen verwalten und verarbeiten. Wie in Abschnitt 3.2 erwähnt kann ein Data Lake in mehrere Zonen unterteilt werden, in denen Daten in unterschiedlichen Stadien der Verarbeitung verwaltet werden. Dies wirkt der ineffizienten Wiederholung von Verarbeitungsschritten entgegen, wenn viele Anwendungen denselben Verarbeitungsschritt der Rohdaten wiederholt durchführen [Mad15]. Giebler et al. [GGH+20b] stellen basierend auf existierenden Zonenmodellen aus der Literatur ein Zonenreferenzmodell vor. Dabei definieren sie sechs verschiedene Zonen sowie deren Charakteristika, um bei der Implementierung einer zonenbasierten Data Lake Verwaltung eine Orientierungshilfe zu bieten.

Innerhalb des Zonenreferenzmodells kommen die Daten in der Landezone an. Von dort werden sie an die Rohzone weitergeleitet, in der sie persistiert werden. In der harmonisierten Zone werden die Daten integriert. Die destillierte Zone bereitet die Daten für die Analyse vor. Die Analyse kann in der explorativen Zone geschehen. Hier können Datenwissenschaftler die Daten untersuchen. Bereitgestellt werden die Daten in der Lieferzone. Sie werden dabei auf spezifische Anwendungen zugeschnitten.

Diese Architekturen betrachten nur die Herausforderungen der Verarbeitung von Big Data und nicht den Schutz der Privatsphäre der betroffenen Personen. Auch werden die semantischen Beziehungen zwischen Daten sowie die Möglichkeit der Verknüpfung von Daten, um neue Informationen zu generieren, nicht berücksichtigt.

Datenverwaltungsarchitekturen mit Semantik innerhalb des Internet of Things. Ein verbreiteter Ansatz zur Beschreibung der Semantik innerhalb Datenverwaltungsarchitekturen des IoT ist die Verwendung einer Ontologie [IA16] (zum Beispiel [AGD16]). In der Ontologie können die Objekte des IoT beschrieben werden oder das was sie an Daten produzieren. Es gibt verschiedene Projekte, die Ontologien oder Vokabular für das IoT und darüber hinaus definieren [SW16]. Ein Beispiel ist die Semantic Sensor Network Ontology[1]. Diese beschreibt Sensoren, Beobachtungen und verwandte Konzepte [W3C].

Ein Beispiel einer Datenverwaltungsarchitektur, die eine Ontologie verwendet, um die in der Architektur gespeicherten Daten semantisch zu beschreiben, ist die Datenverwaltungsinfrastruktur von Kondylakis et al. [KKTM18]. Diese ermöglicht die Verwaltung großer und heterogener Gesundheitsdatensätze. Die Plattform speichert Daten aus unterschiedlichen Quellen in einem Data Lake. Der Data Lake vereint verschiedene heterogene Datenbanken, die verschiedene Technologien zum Speichern und Bereitstellen der Daten verwenden. Ausgewählte Teilmengen des Data Lake werden semantisch angereichert, integriert und untersucht. Durch eine Ontologie wird ein virtuelles Schema aller auf der Plattform gespeicherten Daten erstellt. Diese beschreibt die von der Plattform benötigten und verarbeiteten Daten semantisch. So wird eine gemeinsame Darstellung des Wissens über die verschiedenen Informationsquellen hinweg ermöglicht. Die verfügbaren Daten werden der Ontologie zugeordnet, um einen einheitlichen Datenzugriff zu ermöglichen. Da diese

[1] https://www.w3.org/2005/Incubator/ssn/ssnx/ssn

Zuordnungen fehleranfällig und zeitaufwändig sind, werden Zuordnungen nur für ausgewählte Informationen, die integriert und semantisch angereichert werden müssen, durchgeführt. Mithilfe des Datenintegrationstools Exelixis [KP11] wird die Datenanreicherung und -integration basierend auf der hinterlegten Ontologie durchgeführt. Exelixis kann dabei auch bestehende Zuordnungen und Queries auf eine neue Version der Ontologie übertragen. Der Zugriff auf die Daten ist durch APIs gewährt. Dabei kann entweder auf einzelne Daten oder integrierte Informationen zugegriffen werden. Es ist auch möglich anonymisierte Daten gemäß den Sicherheitsanforderungen des entsprechenden Datennutzungsszenarios bereitzustellen.

Diese Architekturen betrachten zwar die semantischen Beziehungen zwischen Daten sowie die Möglichkeit der Verknüpfung von Daten, um neue Informationen zu generieren, jedoch berücksichtigen sie die Bedrohung der Privatsphäre durch die große Datenmenge und die Möglichkeit der Verknüpfung der Daten nicht.

Datenschutzbewusste Datenbereitstellung. Es gibt verschiedene Ansätze, die die Privatsphäre betroffener Personen bei der Datenbereitstellung innerhalb des IoT schützen [SSGM19]. Im Folgenden wird zuerst auf anwendungsspezifische Zugriffsregeln und anschließend auf kontextbasierte Zugriffsregeln eingegangen. Danach wird der Datenschutz in Zeitreihendaten behandelt, da Sensordaten oft Zeitreihendaten sind und die Eigenschaften dieser Art von Daten sich für den Datenschutz ausnutzen lassen.

Ein Ansatz ist, den Zugriff externer Anwendungen auf bestimmte Daten zu blockieren. Dabei kann der Zugriff pro externer Anwendungen pro Datenart definiert werden. Ein Beispiel hierfür ist PRIVACY-AVARE von Alpers et al. [ABF+18; AOP+17]. Mit PRIVACY-AVARE kann der Nutzer ein Berechtigungskonzept bezüglich des Datenzugriffs externer Anwendungen festlegen. Die Zugriffsberechtigungen können dabei pro Datenart vergeben werden. So kann kontrolliert werden, welche Daten von Sensoren oder anderen Datenquellen zu Anwendungen fließen. Der Nutzer kann zum Beispiel den Zugriff auf die Daten des GPS-Sensors oder einzelnen Attributen davon (Längen- oder Breitengrad) blockieren. Dieser Ansatz ist jedoch sehr einschränkend, da das Blockieren einer Datenart Auswirkungen auf alle Dienste einer Anwendung hat, die dieses Attribut verwenden möchten. Entweder werden die Standortdaten weitergegeben bzw. gesammelt oder nicht, es kann nicht zwischen verschiedenen Diensten differenziert werden. Auch wird nicht berücksichtigt, dass die Daten kombiniert und so neue Informationen generiert werden können, die eine Bedrohung der Privatsphäre darstellen.

ACCESSORS von Stach und Mitschang [SM19] behandelt diese Bedrohung der Privatsphäre, indem modelliert wird welche Informationen aus welchen Quellen ableitbar sind. So kann der Nutzer entscheiden, welche Informationen er bereitstellen möchte und welche nicht. Dabei muss er sich nicht damit beschäftigen aus welchen Daten diese Informationen erlangt werden können. Des Weiteren berücksichtigt ACCESSORS den Kontext, in dem Zugriffsregeln angewendet werden sollen und an den die Zugriffsberechtigungen gebunden sind. Dies ist eine Möglichkeit Zugriffsregeln feiner anzupassen. Diese Art der Zugriffskontrolle beschreiben Kim et al. [KMJ+05]. Dabei werden Nutzern Rollen zugewiesen und den Rollen verschiedenen Berechtigungen. Die Informationen über den Kontext entscheiden, welche dieser Rollen aktiv ist und dadurch welche Berechtigungen der Nutzer aktuell hat. In ACCESSORS [SM19] wird dieses Konzept auf IoT-Geräte angewendet. Ein Nachteil dieses Ansatzes kann durch die Berücksichtigung eines komplexen Kontextes entstehen: Die Konfiguration wird mit steigendem Umfang des Kontextes komplizierter.

Zeitliche Datenkorrelationen können auch ein Datenschutzproblem darstellen. Aus Ihnen ist die Identifizierung komplexer Ereignisse ermöglichen, aus denen Verhaltensmuster abgeleitet werden können [SGG]. Stach et al. [SDM+18] stellen hierzu ein musterbasiertes Verfahren zum Schutz der Privatsphäre vor. Die Daten werden unverändert zur Verfügung gestellt, nur die Zeitstempel werden teilweise verändert, sodass eine Umordnung der Daten im Datenstrom geschieht. Dabei werden jedoch auch weiterhin alle Daten zur Verfügung gestellt und der Datenzugriff nicht eingeschränkt. Auch kann es durch die veränderten Zeitstempel zu Zuordnungsproblemen kommen, wenn Daten angereichert oder verknüpft werden sollen, um neue Informationen zu generieren.

In Zeitreihendaten besitzen Ausreißer oft einen höheren Informationsgehalt und sollten deshalb geschützt werden [Sta19a]. Stach [Sta19a] stellt hierzu fünf Verfahren vor, die je nach Anwendungsfall zum Einsatz kommen können. Umgesetzt werden diese Verfahren in der Architektur von VAULT von Stach [Sta19b]. Durch Dateninterpolation können Wertebereiche mit hohem Informationsgehalt verborgen werden. Die Identifikation solcher Bereiche kann durch eine Ausreißererkennung mittels Vorhersagen unterstützt werden. Falls das gesamte Signal unschärfer gemacht werden soll, kann durch eine Fourier-Transformation eine Signalglättung durchgeführt werden. Anomalien können mithilfe von Wavelet-Transformationen hervorgehoben werden. Um die Daten noch weiter zu verdichten, kann auf den Anomalien eine Clusteranalyse durchgeführt werden. Kombinationen dieser fünf Verfahren sind dabei auch möglich. Durch diese Methoden ist ein angepasster Schutz der Privatsphäre möglich. Nach der Anwendung dieser Verfahren sind die Daten jedoch nicht mehr für alle Anwendungsfälle verwendbar, da Daten verloren gehen. Auch behandeln diese Verfahren nicht, dass Anwendungen der Zugriff auf ausgewählte Daten blockiert werden soll.

Zusammenfassung. Es gibt für die unterschiedlichen Aspekte der Datenverarbeitung und Datenbereitstellung Ansätze in der Literatur. Diese Ansätze vereinen jedoch nicht die Herausforderungen der Verwaltung von Big Data und den Schutz der Privatsphäre bei der Datenbereitstellung. Deshalb wird im Folgenden das Konzept von RETORT vorgestellt.

KAPITEL 5

Datenbereitstellungsplattform für datenintensive IoT-Anwendungen

Innerhalb des folgenden Kapitels wird die Konzeption einer Datenbereitstellungsplattform für datenintensive Internet of Things Anwendungen behandelt. Diese Datenbereitstellungsplattform heißt RETORT. Das im Folgenden vorgestellte Konzept basiert auf dem Konzept von Stach et al. [SGG].

RETORT kann für das in Kapitel 2 beschriebene Anwendungsszenario verwendet werden und erfüllt die in Abschnitt 2.3 genannten Anforderungen. Die Plattform ermöglicht das Verwalten von Big Data. Außerdem dient sie der kontrollierten Weitergabe und bedarfsgerechten Darbietung von Daten und Informationen. Bedarfsgerecht beinhaltet in diesem Kontext die benötigte Datenanreicherung als auch die anschließende Datenreduktion.

Die Daten können nicht nur aus den Daten der Anwendung bestehen, sondern auch aus Metadaten. Um das in Abschnitt 2.1 genannte Beispiel erneut aufzugreifen, sind in den Daten der Anwendung zum Beispiel Fotos enthalten. Die enthaltenen Metadaten eines Fotos sind in diesem Beispiel der Zeitpunkt der Aufnahme, sowie die Koordinaten des Aufnahmeortes. Außerdem können die Namen der abgebildeten Personen angefügt sein. Es besteht die Möglichkeit, mithilfe der Metadaten die Daten der Anwendung anzureichern, zum Beispiel die Koordinaten des Aufnahmeortes mit der Adresse des Ortes. Auch können aus den unterschiedlichen Datenpunkten neue Informationen abgeleitet werden, zum Beispiel welche Person zu welchem Zeitpunkt an welchem Ort was getan hat.

Die Anreicherung und Ableitung neuer Informationen stellt allerdings auch eine Bedrohung der Privatsphäre dar, da durch die große Menge an Daten und Möglichkeiten der Verknüpfung dieser Daten viele Informationen über den Benutzer bereitgestellt werden. Der Bedrohung der Privatsphäre sowie der Schwierigkeit der Verarbeitung von Big Data soll durch eine Datenreduktion begegnet werden. Der Aspekt der Verarbeitung von Big Data wird in dem Konzept von Stach et al. [SGG] nicht berücksichtigt. Auch die Datenreduktion ist nicht Teil davon. In dieser Arbeit wird deshalb vorgestellt, wie RETORT mit den Herausforderungen von Big Data umgehen kann. Auch wird das Konzept um den Aspekt der Datenreduktion erweitert. Des Weiteren wird ein Entwurf der Umsetzung des vorgestellten Konzepts präsentiert.

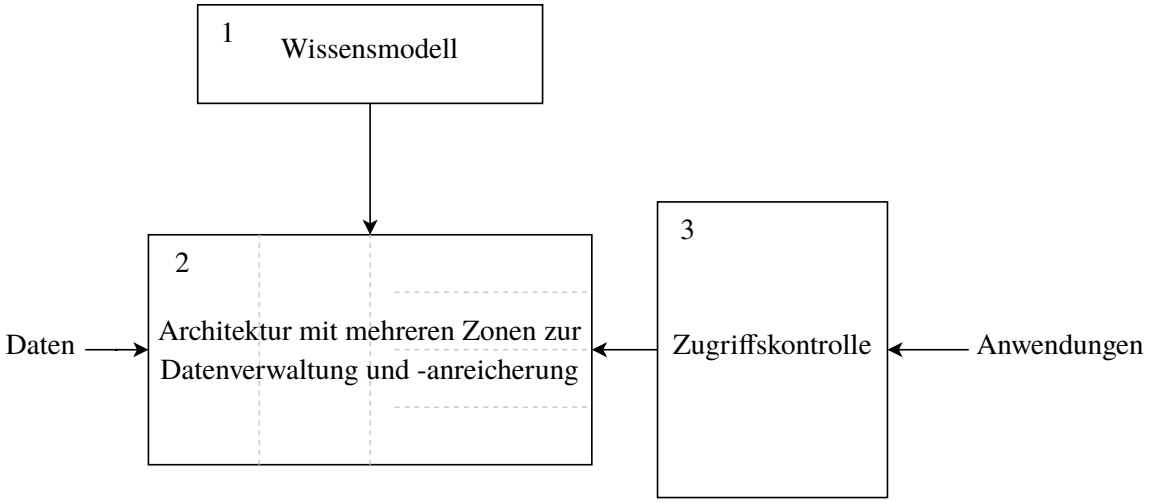
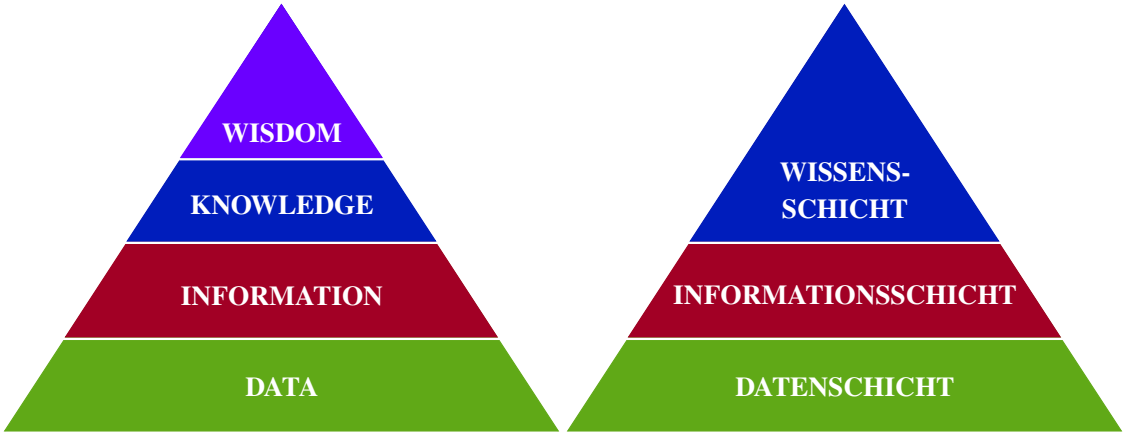


Abbildung 5.1: Die drei Komponenten von RETORT und deren Interaktion.

Basierend auf dem Entwurf von Stach et al. [SGG] besteht RETORT aus folgenden drei Komponenten: ein Wissensmodell, eine Architektur mit mehreren Zonen zur Datenverwaltung und -anreicherung und eine Zugriffskontrolle (siehe Abbildung 5.1). Das Wissensmodell beschreibt, welches Wissen aus welchen Datenanreicherungen abgeleitet werden kann. Auch zeigt es, welche Techniken zum Schutz der Privatsphäre im Verlauf der Datenanreicherung angewendet werden können, ohne den Nutzen des Ergebnisses zu beeinträchtigen. Dies wird genauer in Abschnitt 5.1 erläutert. In der Architektur mit mehreren Zonen zur Datenverwaltung und -anreicherung (Abschnitt 5.2) werden die Daten in getrennten Zonen für jedes Wissensmuster aufbereitet. Eine Zone entspricht einem spezifischen Anwendungsfall. Die Anwendungen, welche auf die Daten zugreifen, erhalten dabei jeweils nur Zugriff auf ausgewählte Anwendungsfälle. Dieser Zugriff wird durch einen Prozess reguliert, der eine Attribut-basierte Zugriffskontrolle darstellt (dritte Komponente). Bei einer Attribut-basierten Zugriffskontrolle werden Zugriffsregeln erstellt, die sich auf die Attribute von Entitäten, Operationen und den relevanten Kontext beziehen. So wird dynamisch bestimmt, auf welche Zonen ein Prozess Zugriff hat und welche Techniken zum Schutz der Privatsphäre angewendet werden müssen. Innerhalb dieser Arbeit wird nur auf die ersten zwei genannten Komponenten eingegangen und der Prozess zur Zugriffsverwaltung nicht behandelt.

Im Folgenden werden das Wissensmodell (Abschnitt 5.1) und die Architektur zur Datenverwaltung und -anreicherung (Abschnitt 5.2) genauer beschrieben. Auch wird genannt, welche der Anforderungen aus Abschnitt 2.3 durch die einzelnen Komponenten erfüllt werden oder welche der Eigenschaften von Big Data (vgl. Abschnitt 3.1) behandelt werden. Eine detaillierte Evaluation wie die Anforderungen erfüllt werden oder die Eigenschaften von Big Data behandelt werden, wäre an dieser Stelle zu umfangreich. Die Evaluation ist deshalb in Kapitel 8 zu finden.



(a) DIKW Pyramide.

(b) Das Wissensmodell, welches die erste Komponente von RETORT darstellt.

Abbildung 5.2: Die DIKW Pyramide und das Wissensmodell von RETORT.

5.1 Wissensmodell

In diesem Abschnitt wird die erste Komponente von RETORT, das Wissensmodell, vorgestellt. Das Wissensmodell behandelt die Eigenschaft *Value* der Big Data. Durch das Wissensmodell wird beschrieben, welche Informationen und welches Wissen aus den Daten abgeleitet werden können. Damit wird der Wert der Daten beschrieben.

Das Wissensmodell basiert auf dem Modell, dass Daten, Informationen und Wissen in einer hierarchischen Beziehung stehen. Dabei werden Informationen auf Basis von Daten abgeleitet und Wissen aus Informationen. Das ist die Grundlage der DIKW Pyramide [Ack89]. In Abbildung 5.2a ist diese dargestellt. Die Abkürzung steht für data-information-knowledge-wisdom, auf deutsch Daten-Informationen-Wissen-Weisheit [Row07]. Laut Ackoff beinhaltet jeder Typ, der weiter oben in der Hierarchie auftritt, die Kategorien, die unter ihm stehen [Ack89].

Basierend auf der DIKW Pyramide (siehe Abbildung 5.2a) besteht das in RETORT verwendete Modell aus drei Schichten. Diese sind in Abbildung 5.2b dargestellt. Die Komponente Weisheit ist dabei nicht Teil des Modells. Aufbauend auf Stach et al. [SGG] werden die drei Schichten wie folgt beschrieben.

Die unterste Schicht, Datenschicht genannt, beinhaltet alle Rohdaten, die zur Anreicherung verwendet werden können. Dabei sind die Daten in ihre einzelnen Komponenten zerlegt. Erfasst werden kann zum Beispiel der Puls des Anwenders. Um die Abhängigkeit des Pulses von äußeren Faktoren zu untersuchen, können zusätzlich auch die Koordinaten des Ortes der Messung bestimmt werden. In der Datenschicht sind in diesem Beispiel also der Wert der Pulsmessung und die Koordinaten getrennt enthalten. Die Beschreibung, wie die verfügbaren Daten interpretiert und verknüpft werden können, um neue Informationen zu generieren, erfolgt in der Informationsschicht. In dem oben eingeführten Beispiel kann aus den Koordinaten der entsprechende Ort abgeleitet werden. Die oberste Schicht heißt Wissensschicht. Hier wird beschrieben, welche Wissensmuster aus den Informationen abgeleitet werden können, wenn die Daten der Datenschicht angereichert werden. In diesem Beispiel

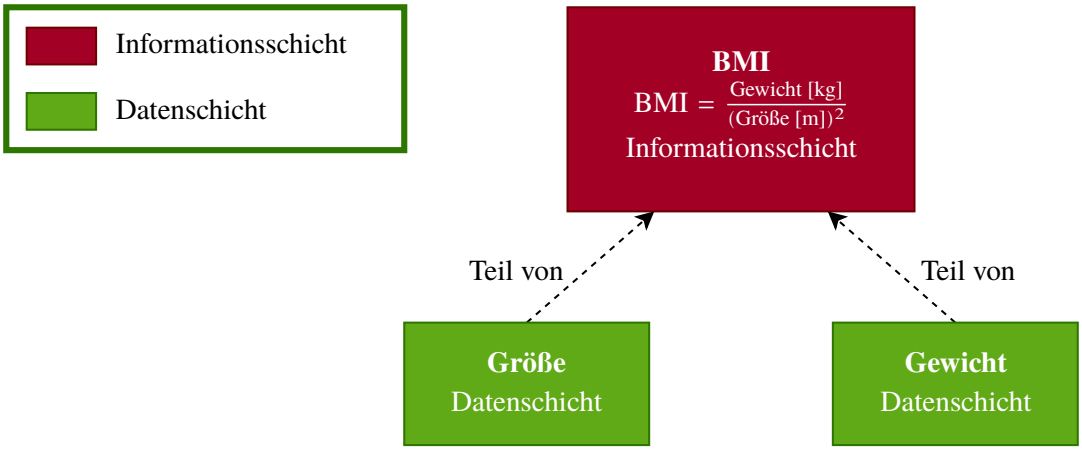


Abbildung 5.3: Ontologie innerhalb des Wissensmodells zur Berechnung des BMI.

kann der Ort mit dem entsprechenden Pulswert verknüpft werden und dadurch zusätzliches Wissen generiert werden, zum Beispiel inwiefern der Aufenthaltsort des Patienten eine Auswirkung auf dessen Puls hat.

Im Zuge dieser Arbeit wird eine Umsetzung des Wissensmodells vorgeschlagen: Das Wissensmodell wird durch eine Ontologie dargestellt. Die in Abschnitt 3.4 vorgestellte Instanzebene ist dabei kein Bestandteil der Ontologie. Ein beispielhafter Auszug der Ontologie ist in Abbildung 5.3 gezeigt. Innerhalb der Ontologie werden die Rohdaten der Datenschicht, die Informationen der Informationsschicht und die Wissensmuster der Wissensschicht als Klassen modelliert. Diese sind in der Abbildung mit rechteckigen Rahmen dargestellt. Dabei enthalten die Klassen der Informations- und Wissensschicht nicht nur Repräsentationen neuer Daten, sondern auch Vorschriften, wie die Daten verknüpft werden. Im Beispiel der Abbildung 5.3 enthält die Informationsschicht die Berechnungsvorschrift zur Berechnung des BMI aus den Daten Gewicht und Größe sowie die berechneten Daten des Body Mass Index (BMI). Die Verknüpfung der Daten bzw. Klassen erfolgt durch Relationen. Die Relationen sind in der Abbildung durch gestrichelte Pfeile dargestellt. In den Eigenschaften der Klassen ist festgehalten, zu welcher der drei Schichten des Wissensmodells sie gehören. Diese stehen in der Abbildung unter dem fett gedruckten Klassennamen. Die Eigenschaften der Klassen können auch weitere Informationen enthalten, wie zum Beispiel in der Abbildung die Formel zur Berechnung des BMI.

5.2 Datenverwaltung und -anreicherung

In diesem Abschnitt wird die zweite Komponente von RETORT vorgestellt. Diese beinhaltet eine Architektur mit mehreren Zonen zur Datenverwaltung und -anreicherung innerhalb von RETORT. In dieser Komponente von RETORT werden die Daten persistiert und entsprechend dem Wissensmodell angereichert. Die Grundidee der Komponente ist, Datenquellen und Datensenken voneinander zu isolieren und so den Datenfluss zu verwalten [SGG].

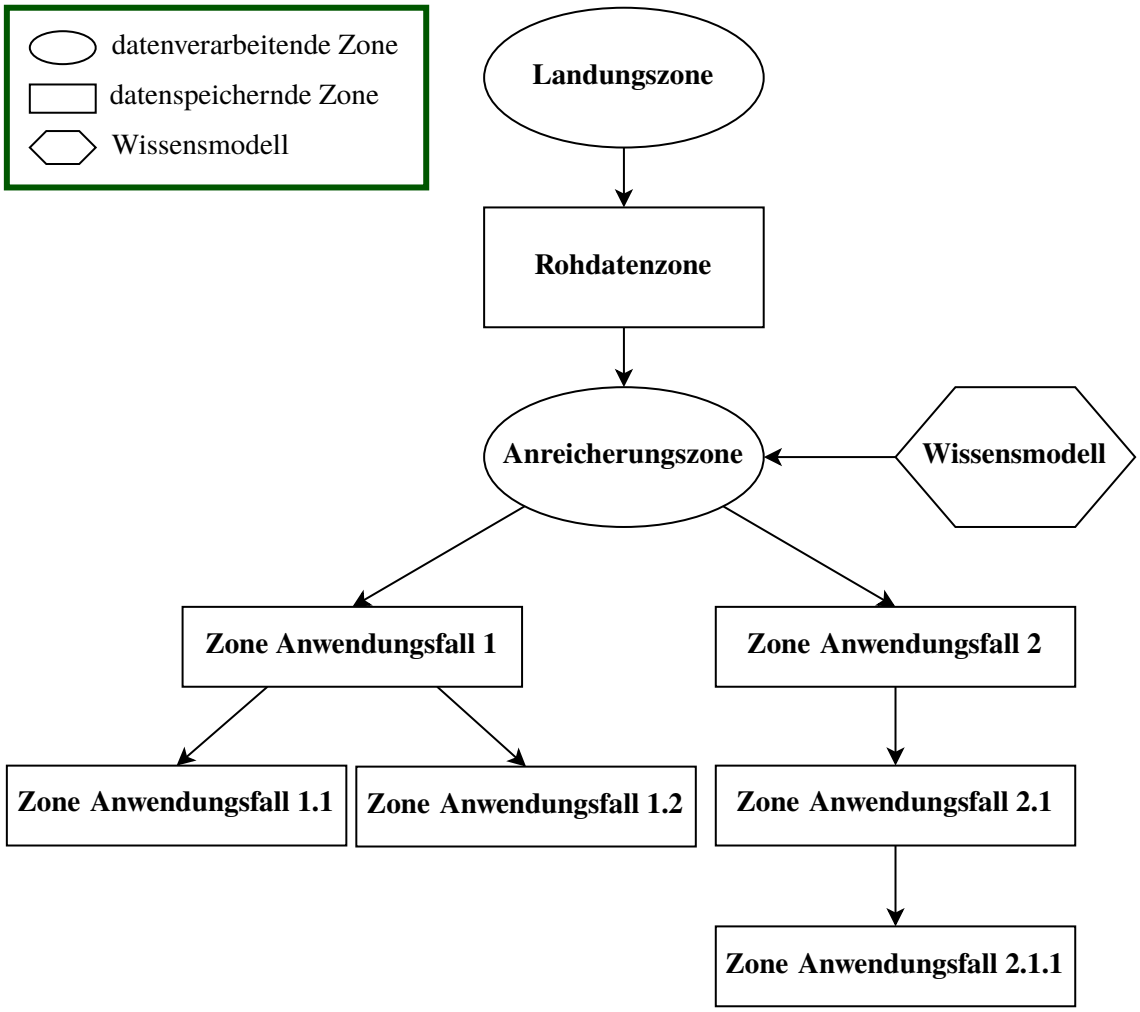


Abbildung 5.4: Die Architektur mit mehreren Zonen zur Datenverwaltung und -anreicherung, welche die zweite Komponente von RETORT darstellt.

Das verwendete Datenmanagementkonzept basiert auf der von Sharma eingeführten Architektur für Data Lakes. Diese beschreibt ein Multi-Zonen-Modell für die Verwaltung von Data Lakes [Sha18]. In RETORT wird die Architektur von Sharma wie nachfolgend beschrieben umgesetzt und entsprechend der Anwendung angepasst.

In einem Data Lake werden große heterogene Datenmengen gespeichert (Eigenschaften *Volume* und *Variety* von Big Data). Durch eine Reduktion des Datenvolumens werden die Daten für eine effiziente Verarbeitung vorbereitet. Dazu werden innerhalb der Architektur von RETORT die für einen Anwendungsfall relevanten Daten identifiziert und entsprechend der gewünschten Anwendung zusammengeführt. So werden die Daten gefiltert und aufbereitet. Da es ineffizient wäre, den Data Lake für einen Anwendungsfall wiederholt anzufragen, besteht die hier gewählte Architektur aus mehreren Zonen. Verschiedene Zonen beinhalten dabei die Daten in unterschiedlichen Stadien der Verarbeitung. Die verwendete Architektur ist in Abbildung 5.4 dargestellt. Es gibt Zonen, welche die Daten verarbeiten und Zonen, welche die Daten speichern. Diese sind in Abbildung 5.4 durch

ovale bzw. eckige Rahmen dargestellt. Dem Endanwender werden die Daten in verschiedenen Zonen entsprechend verschiedener Anwendungsfälle zur Verfügung gestellt. Er kann auswählen, welche Zone seiner Anwendung entspricht.

Durch dieses Konzept werden die Eigenschaften *Volume* und *Variety* von Big Data behandelt.

Im Folgenden werden die einzelnen Zonen in der Reihenfolge von oben nach unten genauer behandelt.

Landezone

Die Daten kommen in einer Landezone an. Es handelt sich dabei um eine Übergangszone, in der die Daten temporär gepuffert werden. Durch die Pufferung kann die Eigenschaft *Velocity* von Big Data behandelt werden. Diese beschreibt die hohe Geschwindigkeit, mit der die Daten generiert werden.

Der Puffer wird nach dem Prinzip First In – First Out (FIFO) umgesetzt (Beispiel: Queue). Im Vergleich zum Last In – First Out (LIFO) Prinzip (Beispiel: Stack) kann somit nicht der Fall eintreten, dass Daten aufgrund des großen Datenstroms nie verarbeitet werden. Die Rohdaten werden vor der Speicherung in Nutzdaten und Metadaten getrennt. Dadurch werden die ankommenden Daten in dieser Zone nicht nur gepuffert, sondern auch in ihre einzelnen Datenkomponenten zerlegt. Die Zerlegung unterstützt die Bereitstellung einzelner Attribute der Daten, sowie die neuen Rekombinationen mit Daten anderer Quellen. Dadurch wird der Umgang mit der Eigenschaft *Volume* von Big Data erleichtert.

Die Landezone ist somit eine datenverarbeitende Zone. Analog zu dem Beispiel aus Abschnitt 5.1 können bei einer Pulsmessung neben dem Messwert auch die Koordinaten des Messortes als Metadaten erfasst werden. Der Datenpunkt, bestehend aus Pulsmessung und Koordinaten, wird hier in zwei Datenpunkte zerlegt. Diese werden separat gespeichert. Bei der Zerlegung müssen eindeutige Identifier die Rekombination der Daten gewährleisten. Dies und die Verwendung eines Puffers sind jedoch nicht Schwerpunkte dieser Arbeit und werden in den folgenden Kapiteln nicht weiter behandelt.

Rohdatenzone

Die einzelnen Daten werden an die Rohdatenzone weitergeleitet. Diese Zone ist eine datenspeichernde Zone. In der Rohdatenzone werden die Daten wie sie die Landezone verlassen, im Rohformat, gespeichert. Da die Daten unverändert in Bezug auf Format und Inhalt der Daten gespeichert werden, gehen durch den Speicherprozess keine Informationen verloren. Entsprechend Anforderung A2 (Erweiterbarkeit) können so später noch neue Analysemethoden definiert werden, die andere Aspekte der Daten benötigen als die ursprünglich definierten Anwendungsfälle. Ein Beispiel für den Unterschied zwischen dem Rohformat und den verarbeiteten Daten ist die Speicherung von Bildern. Das Rohdatenformat eines Bildes ist das Dateiformat, bei dem die Kamera die Daten nach der Digitalisierung weitgehend ohne Bearbeitung auf das Speichermedium schreibt. Dieses Format enthält alle Informationen, die der Bildsensor bei der Aufnahme erfasst. Für die Speicherung im gängigen Format JPEG nimmt die Kamera Korrektur- und Kompressionsberechnungen vor. Bei der

Kompression werden zum Beispiel zusätzliche Bilddetails, die nicht sichtbar sind, vernichtet. Es gehen somit Bildinformationen verloren [Bau05]. Die so komprimierten Bilder können nicht mehr für jeden beliebigen Anwendungsfall verwendet werden.

Durch die Flexibilität eines Data Lake bezüglich der sich in ihm befindlichen Daten und der Möglichkeit, die Daten zu speichern, ohne sie zuerst strukturieren zu müssen, eignet er sich für die Verwaltung der Rohdaten. In RETORT werden die Daten der Rohdatenzone in einer NoSQL Datenbank persistiert. Diese Art der Datenbank besitzt kein festes Schema, welches auf alle Datensätze angewandt wird. Sondern jeder Datensatz kann ein eigenes Schema besitzen, wodurch er sich selbst beschreibt. Sie besitzen also eine größere Flexibilität im Format und der Struktur der gespeicherten Daten. Dadurch müssen die Rohdaten vor der Speicherung nicht an ein Schema angepasst werden, wodurch Daten verloren gehen könnten.

Der Zugriff auf die gespeicherten Rohdaten darf nur innerhalb der Plattform erfolgen und nicht von außerhalb. Durch die große Datenmenge und Datenvielfalt sowie durch Verknüpfungen der Daten können viele Erkenntnisse über die betroffene Person erlangt werden. Dies stellt, wie bereits erwähnt, eine Bedrohung der Privatsphäre dar. Deshalb haben externe Anwendungen keinen Zugriff auf die Rohdaten, sondern nur auf ausgewählte, eventuell vorverarbeitete Daten. Dieses Konzept erfüllt die Anforderung, dass die Datenweitergabe eingeschränkt wird (Anforderung A3).

Anreicherungszone und Zonen der Anwendungsfälle

Die Verarbeitung der Rohdaten geschieht in der Anreicherungszone. Bei dieser Zone handelt es sich um eine datenverarbeitende Zone. Die Daten werden entsprechend der Informations- und Wissensschicht des Wissensmodells aufbereitet und zusammengefügt. Die Informationen für die Verarbeitung werden aus der Ontologie des Wissensmodells mithilfe einer Ontologie-Abfragesprache ermittelt. Somit werden die benötigten Daten und Verarbeitungsschritte ermittelt. Zum Beispiel soll aus Gewicht und Körpergröße der BMI ermittelt werden. Die Berechnung des BMI aus Körpergröße und Gewicht anhand der Formel[1] $\text{BMI} = \frac{\text{Gewicht [kg]}}{(\text{Größe [m]})^2}$ kann aus der Ontologie abgefragt werden. Die Ontologie für dieses Beispiel ist in Abbildung 5.3 dargestellt. Durch die Zusammenfassung von Daten und die daraus erfolgende Berechnung anderer Kenngrößen, kann die Aussagekraft der Daten reduziert werden (Anforderung A5). Dem oben genannten Beispiel entsprechend enthält der BMI weniger Details als die Größe und das Gewicht, da der BMI nur ein Verhältnis der beiden Werte angibt. Somit werden weniger Informationen über die betroffene Person preisgegeben.

Die aufbereiteten Daten werden in getrennten Zonen für jedes Wissensmuster bereitgestellt. Dies sind die Zonen der Anwendungsfälle (datenspeichernde Zonen). Eine Zone entspricht dabei einem spezifischen Anwendungsfall. Dieser enthält nicht zwingend alle verfügbaren Daten, sondern eventuell nur eine Teilmenge.

Durch die definierten Anwendungsfälle wird Anforderung A1 (Flexibilität) erfüllt, da die Daten so an unterschiedliche Benutzergruppen angepasst zur Verfügung gestellt werden können. Dieses Konzept dient zum einen, wie oben genannt, dem Schutz der Privatsphäre der betroffenen Person. Zum anderen ermöglicht die Datenreduktion auch die Verarbeitung der Daten mit herkömmlichen Methoden und behandelt somit die Eigenschaft *Volume* von Big Data.

[1] Formel siehe https://www.apotheken-umschau.de/bmi-rechner (Stand 13.09.2020)

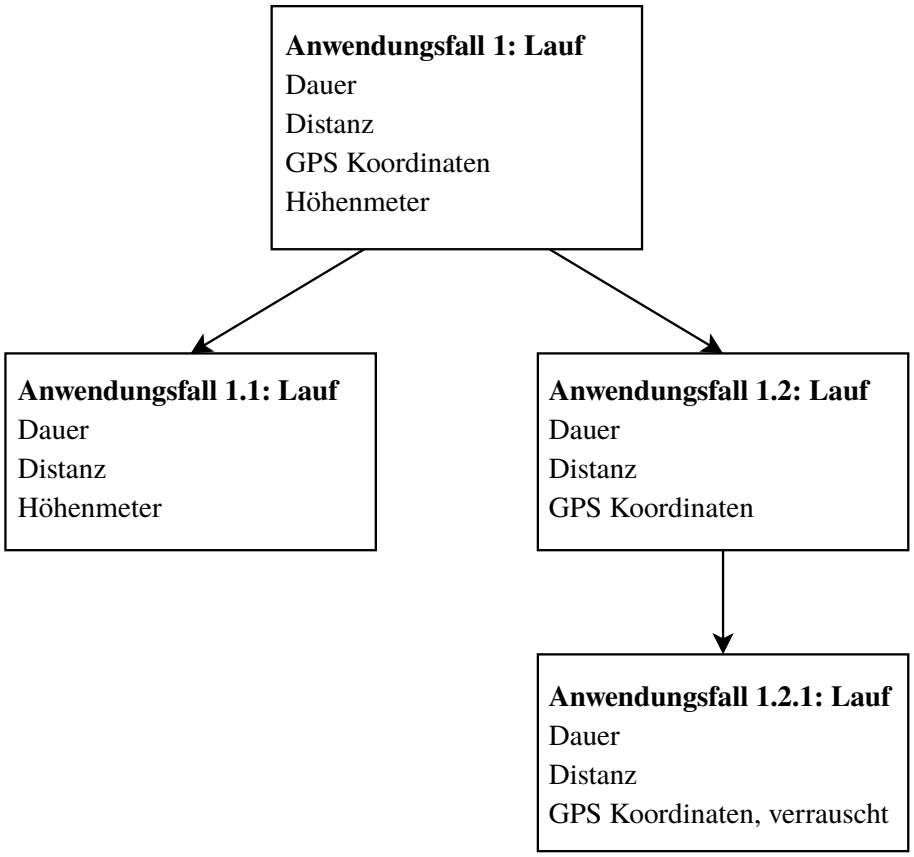


Abbildung 5.5: Vier Anwendungsfälle für Trainer, welche die Daten der Läufe eines Sportlers enthalten.

Im Rahmen dieser Arbeit wird das Konzept der Zonen der Anwendungsfälle von Stach et al. [SGG] erweitert. Die Anwendungsfälle werden hierarchisch in mehreren Baumstrukturen angeordnet. Dabei hat jeder Anwendungsfall mit Ausnahme der Wurzel genau einen Vorgänger. Dies dient der logischen Strukturierung der Anwendungsfälle: In einer Baumstruktur sind semantisch zusammenhängende Anwendungsfälle zusammengefasst. Zum Beispiel können in einem Baum die Anwendungsfälle, welche Daten für Trainer bereitstellen, zusammengefasst werden, in einem anderen die Daten für Physiotherapeuten. Es ist dabei darauf zu achten, dass die sich ergebenden Bäume logisch getrennt bleiben und sich keine thematischen Kreuzverbindungen ergeben. In Abbildung 5.5 ist beispielhaft ein Auszug aus den Anwendungsfällen für Trainer dargestellt. Dieser Teilbaum beinhaltet vier Anwendungsfälle, welche die Daten der Läufe eines Sportlers enthalten.

Dieses Konzept erfüllt Anforderung A6 (Kategorie der Daten des Anwendungsfalles berücksichtigen). Die Trennung der Anwendungsfälle nach Bereichen ermöglicht es, bei der Datenreduktion zu berücksichtigt, für welchen Bereich die Reduktion durchgeführt werden soll und zu welchem Bereich die zu reduzierenden Daten gehören.

Des Weiteren werden in Anwendungsfällen, die sich näher zur Wurzel befinden mehr bzw. genauere Daten bereitgestellt, als in entfernteren Anwendungsfällen. Mit einem Abstieg innerhalb der Baumstruktur werden die Anwendungsfälle also verfeinert. Dies ist ebenfalls in Abbildung 5.5 dargestellt. Die Anwendungsfälle auf der zweiten Ebene enthalten nur noch drei Attribute anstatt

den vier Attributen der ersten Ebene. Der Anwendungsfall auf Ebene drei enthält statt den exakten GPS Koordinaten verrauschte GPS Koordinaten. Es handelt sich hier somit um eine subtraktive Filterung, da in aufeinanderfolgenden Stufen des Filterns Ergebnisse eliminiert werden. Damit wird angepasst an den Anwendungsfall eine Reduktion durchgeführt. Dies ist nützlich, da Trainer zum Beispiel nie Blutwerte sehen sollten und diese somit gleich zu Beginn der Struktur für alle darunterliegenden Anwendungsfälle entfernt werden können. Durch dieses Konzept wird Anforderung A4 (Anwendungsbezogene Reduktion) erfüllt.

Diese Struktur ermöglicht es, dass ein Anwendungsfall als Grundlage für andere Anwendungsfälle wiederverwendet werden kann. Somit stellen bereits definierte Anwendungsfälle Templates dar und erleichtern die Definition neuer Anwendungsfälle. Dies unterstützt die Erfüllung von Anforderung A2 (Erweiterbarkeit).

Die Zonen der Anwendungsfälle können durch Sichten auf die Daten der Rohdatenzone umgesetzt werden oder indem sie eine Kopie der Daten der Rohdatenzone in der verarbeiteten Version enthalten. Dabei werden die bereitgestellten Daten nicht aus der Rohdatenzone gelöscht, da zum einen sonst Informationen durch die Verarbeitung verloren gehen können und zum anderen die Daten sonst neuen Anwendungsfällen nicht mehr zur Verfügung stehen. In RETORT werden die angereicherten Daten in den Zonen der Anwendungsfälle gespeichert. So muss nicht bei jedem Abrufen der Daten aus den Anwendungsfällen wiederholt die Anreicherung durchgeführt werden.

Durch die doppelte Speicherung von Daten in der Rohdatenzone und den Zonen der Anwendungsfälle entsteht ein Synchronisationsproblem: Wenn Daten in der Rohdatenzone hinzugefügt, aktualisiert oder gelöscht werden, ist diese Änderung nicht automatisch in den Zonen der Anwendungsfälle zu sehen. Deshalb sind die Daten in den Zonen der Anwendungsfälle nur als Cache zu verstehen und müssen in regelmäßigen Intervallen aktualisiert werden. Dies kann zum Beispiel alle 24 Stunden in der Nacht erfolgen. Während des Update Prozesses kann die Baumstruktur verwendet werden, um nicht wiederholt die gleichen Verarbeitungsschritte durchführen zu müssen. So können zuerst die Daten des Anwendungsfalls an der Wurzel aktualisiert werden. Die Anwendungsfälle der Kindknoten können dann basierend auf den Daten des Wurzelknotens aktualisiert werden, da sie per Definition nicht mehr Daten als der Wurzelknoten enthalten können. Für Anwendungsfälle, die immer die aktuellsten Daten benötigen, müssen diese bei jedem Aufruf neu berechnet werden.

Der Zugriff externer Anwendungen auf die Zonen der unterschiedlichen Anwendungsfälle geschieht nach Stach et al. über eine Datenschutzzone (nicht in Abbildung 5.4 dargestellt). Dabei können Datenschutzfilter auf die Daten der Zonen angewendet werden (Anforderung A5: Reduktion der Aussagekraft der Daten) [SGG].

Um zu kontrollieren, dass externe Anwendungen nur auf ausgewählte Anwendungsfälle Zugriff haben, betten Stach et al. die Zonen in eine Attribut-basierte Zugangskontrolle ein. Bei einer Attribut-basierten Zugriffskontrolle werden Zugriffsregeln erstellt, die sich auf die Attribute von Entitäten, Operationen und den relevanten Kontext beziehen. Dadurch ist eine präzisere Zugriffskontrolle möglich [HKF15]. So kann dynamisch bestimmt werden, auf welche Zonen ein Prozess Zugriff hat und welche Techniken zum Schutz der Privatsphäre angewendet werden müssen [SGG].

Diese zwei Komponenten sind jedoch nicht Schwerpunkt dieser Arbeit.

KAPITEL 6

Implementierungskonzept

Innerhalb dieser Arbeit wird ein Implementierungskonzept für die RETORT Plattform vorgestellt. Dieses Kapitel beschreibt basierend auf dem in Kapitel 5 vorgeschlagenen Entwurf das Implementierungskonzept der RETORT Plattform zur Bereitstellung von Daten für datenintensive IoT-Anwendungen. Dabei werden die einzelnen Teilaspekte der Architektur nacheinander behandelt. Zuerst beschreibt Abschnitt 6.1 die Umsetzung des Wissensmodells. Anschließend werden in Abschnitt 6.2 die Implementierungskonzepte der einzelnen Zonen vorgestellt.

6.1 Wissensmodell

In diesem Abschnitt wird die Umsetzung der ersten Komponente von RETORT, des Wissensmodells, erläutert.

Die Ontologie des Wissensmodells wird mithilfe des Resource Description Frameworks (RDF) beschrieben. RDF wird als der relevanteste Standard für Datenrepräsentation und Austausch im Semantic Web angesehen [HBEV04]. Eine Relation des Wissensmodells mit den zwei zugehörigen Klassen wird durch ein Tripel dargestellt [CWL14]. Dabei werden die Klassen durch die Subjekte bzw. Objekte repräsentiert und die Relationen durch die Prädikate. Die Eigenschaften der Klassen werden jeweils auch durch Tripel dargestellt. Dabei ist die beschriebene Klasse das Subjekt, der Attributbezeichner das Prädikat und die Eigenschaft das Objekt. Eine Eigenschaft, die jede Klasse in der Ontologie besitzt, ist die Schicht, zu der sie gehört. In diesem Fall stellt das Subjekt die betroffenen Daten bzw. Informationen dar und das Objekt die Schicht, zum Beispiel die „Datenschicht“. Das verbindende Prädikat ist „gehört_zu_Schicht“, kurz „Schicht“. Dies wird in Abbildung 6.1 verdeutlicht. Unten links in der Abbildung wird zum Beispiel die Zuordnung des Gewichts zur Datenschicht dargestellt.

In Abbildung 6.1 ist der RDF-Graph zur Beschreibung der Berechnung des Body Mass Index (BMI) zu sehen. Dabei sind IRIs durch rechteckige Rahmen und Literale durch ovale Rahmen dargestellt.

Innerhalb dieser Arbeit wird für die Umsetzung der Ontologie des Wissensmodells RDF verwendet, da RDF im Gegensatz zu LPGs Ontologiemodellierungssprachen unterstützt [DYB20]. Ein Beispiel einer Ontologiemodellierungssprache ist RDF-Schema (RDFS). Mit RDFS kann ein einheitliches Vokabular zur Beschreibung einer Anwendungsdomäne definiert werden. So können die Bezeichner

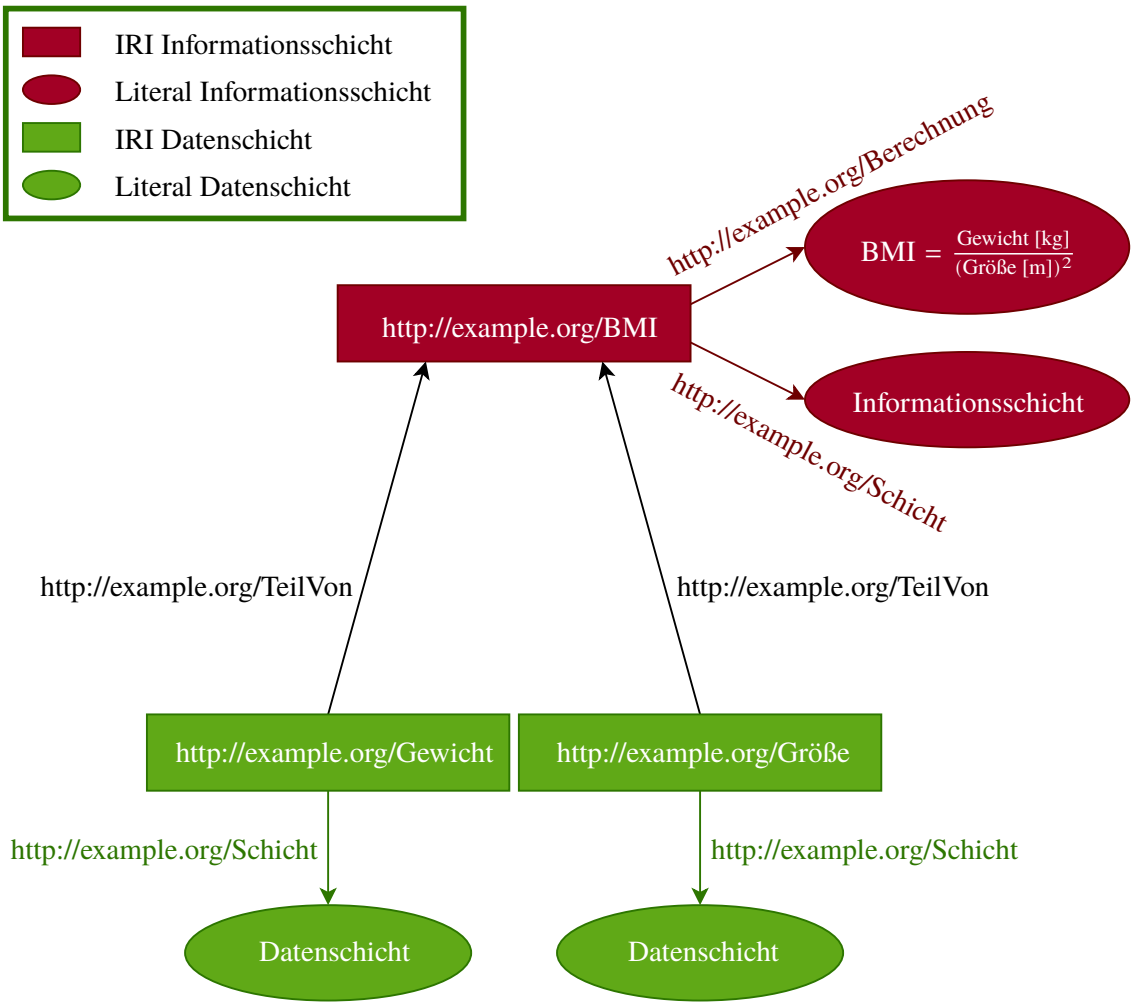


Abbildung 6.1: RDF-Graph zur Beschreibung der Berechnung des BMI.

über mehrere RDF Graphen hinweg vereinheitlicht werden [Hit08]. Ein Beispiel ist das Friend of a Friend (FOAF) Projekt[1]. Hier wird durch RDFS ein Vokabular zur Beschreibung von Personen, ihrer Aktivitäten und ihrer Beziehungen zu anderen Personen oder Objekten definiert [BM14].

Ein weiterer Vorteil von RDF ist, dass durch die globale Definition von Bezeichnern RDF im Gegensatz zu einem LPG auch in der Lage ist, Wissen zwischen mehreren Instanzen auszutauschen. Außerdem ist RDF dafür ausgelegt, Schlussfolgerungen aus dem repräsentierten Wissen zu ziehen [DYB20]. Ein LPG ist dagegen auf Graphprobleme ausgelegt, wie zum Beispiel der Ermittlung des kürzesten Pfades zwischen zwei Knoten oder der Ermittlung von Zusammenhangkomponenten [MK16].

Die RDF Tripel werden in einem Triple Store gespeichert. Diese Art der Datenbanken ist auf das effiziente Durchsuchen von gespeicherten Tripel ausgelegt. Es gibt verschiedene Arten, einen Triple Store technisch umzusetzen. Diese können in drei Kategorien eingeteilt werden: Triple Stores, die auf einem relationalen Ansatz basieren, NoSQL Triple Stores und native Triple Stores. Die am

[1] http://www.foaf-project.org/

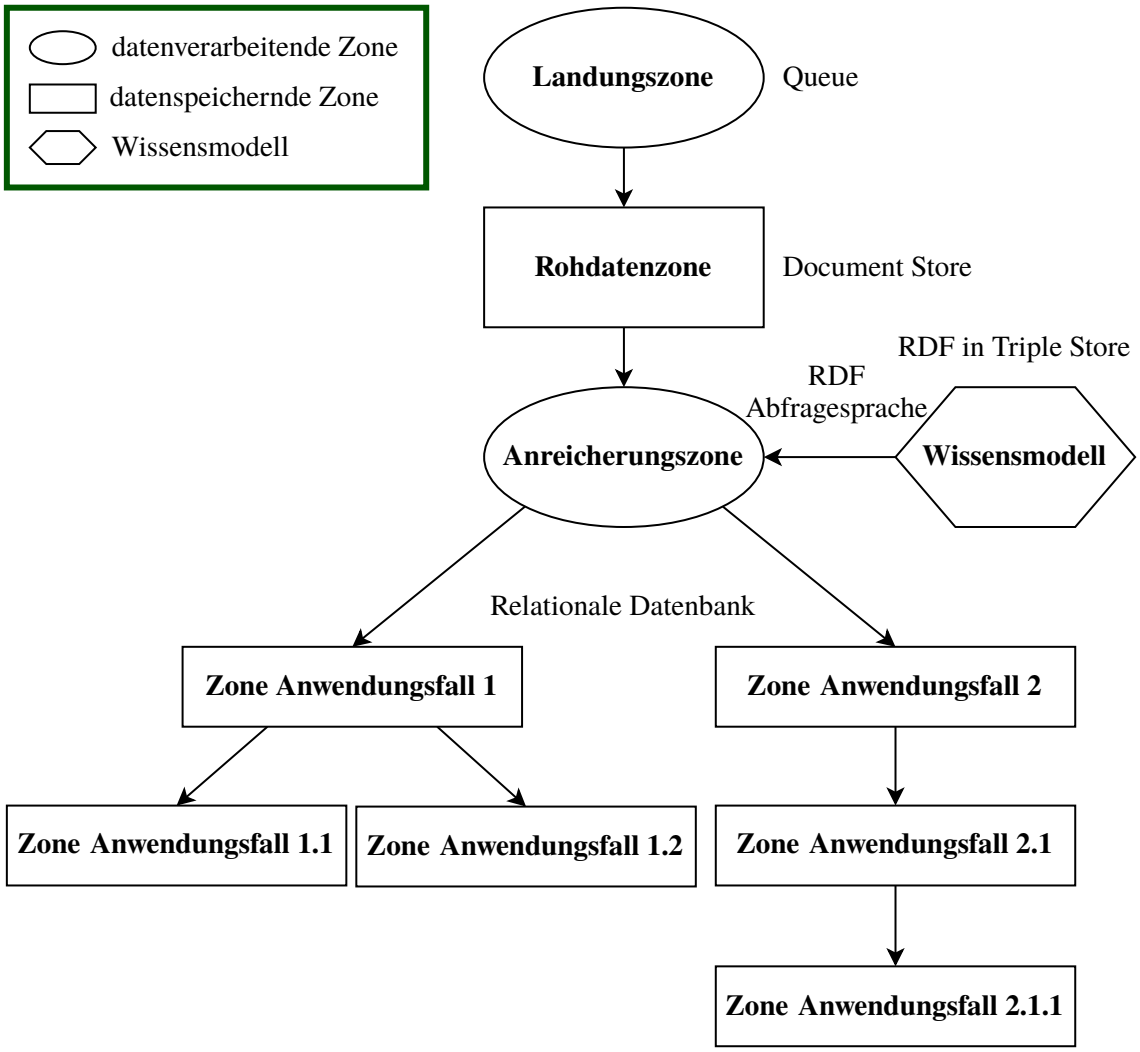


Abbildung 6.2: Die Architektur mit mehreren Zonen zur Datenverwaltung und -anreicherung, welche die zweite Komponente von RETORT darstellt.

häufigsten verwendeten Triple Stores sind native Triple Stores [sol20b]. Native Triple Stores werden unabhängig von relationalen und NoSQL Datenbanken entwickelt. Sie nutzen das RDF Modell, um die Daten effizient zu speichern und abzufragen [BB19]. Es wird hier vorgeschlagen einen nativen Triple Store zu verwenden. Da in verschiedenen Untersuchungen (zum Beispiel [PZLN18]) festgestellt wurde, dass es keinen Triple Store gibt, der in allen Arten von Anfragen überlegen ist, kann sich die Empfehlung aber auch abhängig von der Struktur des Wissensmodells unterscheiden. Eine detaillierte Evaluation dieses Zusammenhangs wird zukünftigen Arbeiten überlassen.

6.2 Datenverwaltung und -verarbeitung

In diesem Abschnitt wird die Umsetzung der zweiten Komponente von RETORT präsentiert. Diese beinhaltet eine Architektur mit mehreren Zonen zur Datenverwaltung und -anreicherung innerhalb von RETORT. In Abbildung 6.2 ist die Architektur dargestellt.

Die einzelnen Zonen werden im Folgenden in der Reihenfolge von oben nach unten genauer behandelt. Die Landezone ist dabei nicht Schwerpunkt dieser Arbeit und wird nicht genauer vorgestellt. Diese Zone unterstützt die technische Umsetzung der Datenbereitstellungsplattform. Jedoch hat sie keinen Einfluss auf die Datenbereitstellung und Datenreduktion, da der Nutzer keinen Zugriff auf sie hat.

Rohdatenzone

Innerhalb der Rohdatenzone werden die zerlegten Daten in einer dokumentenorientierten Datenbank persistiert. Ein JSON-Datei kann zum Beispiel eine Pulsmessung und die Koordinaten des Ortes, an dem die Messung stattgefunden hat, beinhalten. Diese wird in zwei JSON-Dateien zerlegt. Dabei beinhaltet die eine Datei die Pulsmessung und die andere Datei die Koordinaten des Ortes der Messung.

In Schlüssel-Werte-Datenbanken können Daten nur durch ihren Schlüssel gefunden werden. Dokumentenorientierte Datenbanken hingegen erlauben das Suchen nach Daten auch basierend auf dem Inhalt eines Dokuments [OBLB17]. Auch in spaltenorientierten Datenbanken können die Daten nur per Zeilenschlüssel abgefragt werden [RE10]. Somit sind dokumentenorientierte Datenbanken in der Abfrage der gespeicherten Daten flexibler.

Des Weiteren unterstützen dokumentenorientierte Datenbanken verschachtelte und hierarchische Daten. Im Vergleich zu Schlüssel-Werte-Datenbanken, welche keine Struktur in den gespeicherten Werten haben müssen, unterstützen dokumentenorientierten Datenbanken dadurch eine Strukturierung der Daten. Auch spaltenorientierte Datenbanken ermöglichen das Speichern von verschachtelten Elementen nicht. Die Sensordaten, welche in RETORT ankommen, können verschachtelt sein. In Listing 6.1 ist ein Beispiel einer verschachtelten JSON-Datei dargestellt, die ein Pulsmesser übermitteln könnte. Wenn bei jeder Messung gespeichert werden soll, welcher Sensor diese Messung durchgeführt hat, ist somit eine Unterstützung der Datenbank für verschachtelte Daten nützlich. Durch die Speicherung der Quelle der Daten, kann bei der Verwendung der Daten zum Beispiel entschieden werden, wie vertrauenswürdig die Daten basierend auf der Vertrauenswürdigkeit der Quelle sind. Dies entspricht der Eigenschaft *Veracity* von Big Data.

Ein weiterer Punkt ist, dass dokumentenorientierte Datenbanken die Aggregation von Daten ermöglichen. Dies unterscheidet sie von Schlüssel-Werte-Datenbanken. So kann zum Beispiel die Summe oder der Mittelwert eines Arrays berechnet werden. Durch diese Möglichkeit wird die Verarbeitung der Daten in der Anreicherungszone unterstützt.

Graphdatenbanken unterscheiden sich von diesen drei Arten an NoSQL Datenbanken dadurch, dass sie darauf ausgelegt sind Beziehungen zwischen Informationen darzustellen und zu verwalten. Diese Eigenschaft wird hier jedoch nicht benötigt. Somit sind Graphdatenbanken nicht geeignet zur Verwaltung der Daten in der Rohdatenzone.

```json
{
  "id": 1,
  "sensor": {
    "type": "pulse",
    "model": "Fitbit Charge HT"
    "id": "12345"
  },
  "value": 60
}
```

Listing 6.1: Beispiel einer verschachtelten JSON-Datei eines Sensors.

Aufgrund der Eigenschaften von dokumentenorientierten Datenbanken wird dieses Konzept gewählt. Schlüssel-Werte-Datenbanken und spaltenorientierte Datenbanken erfüllen diese Eigenschaften nicht. Auch Graphdatenbanken sind für diesen Anwendungszweck ungeeignet.

Anreicherungszone und Zonen der Anwendungsfälle

Um die Daten in der Anreicherungszone anreichern und verknüpfen zu können, muss die Ontologie des Wissensmodells abgefragt werden. Da die Daten des Wissensmodells im RDF Format gespeichert sind, wird dazu eine RDF Abfragesprache verwendet. Hierzu wird eine Sprache der SPARQL Familie verwendet, da diese die meistverbreitete Sprachfamilie ist. Der Vertreter SPARQL ist eine W3C-Empfehlung [The13].

Für die Anreicherung und Verknüpfung der Daten gibt es verschiedene Techniken. Die Daten können aggregiert werden. Das bedeutet, dass sie in einer zusammengefassten Form dargestellt werden. Abbasian Dehkordi et al. [AFR+20] untersuchen verschiedene Techniken der Datenaggregation in IoT Sensornetzen. Auch kann eine Informationsintegration durchgeführt werden, die Daten also in eine einheitliche Datenstruktur überführt werden. Eckstein [Eck11] stellt hierfür verschiedenen Integrationsansätze vor. Eine weitere Möglichkeit ist, eine Datenfusion durchzuführen. Alam et al. [AMK+17] stellen hierfür verschiedenen Möglichkeiten vor. Eine detaillierte Evaluation der möglichen Ansätze wird zukünftigen Arbeiten überlassen.

Die angereicherten Daten werden in einer relationalen Datenbank gespeichert. So muss nicht bei jedem Abrufen der Daten aus den Anwendungsfällen wiederholt die Anreicherung durchgeführt werden. Dies stellt die Umsetzung der Zonen der Anwendungsfälle dar.

Relationale Datenbanken haben gegenüber NoSQL Datenbanken den Vorteil, dass sie Fremdschlüssel unterstützen [EDU20]. Dabei handelt es sich um eine Spalte in einer Tabelle, die auf den Primärschlüssel einer anderen oder derselben Tabelle verweist [Sch07]. Dies unterstützt die Normalisierung der Datenbanken, wodurch redundante Informationen vermieden werden können [MK16]. Des Weiteren erfüllen relationale Datenbanken das ACID Prinzip (Atomicity, Consistency, Isolation, Durability), NoSQL Datenbanken hingegen das BASE Prinzip (Basically Available, Soft State, Eventually Consistent). Der Unterschied hinsichtlich der Konsistenz ist, dass durch ACID jederzeit Konsistenz gewährleistet ist. Bei BASE wird die Konsistenz verzögert etabliert. So kann es passieren, dass einzelne Knoten noch nicht konsistent nachgeführt sind [MK16]. Bei der Datenbereitstellung

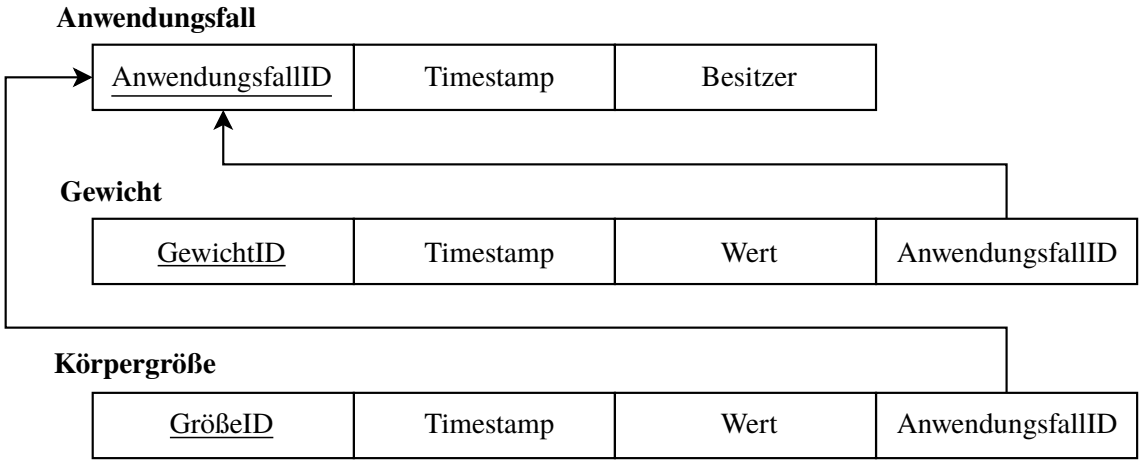
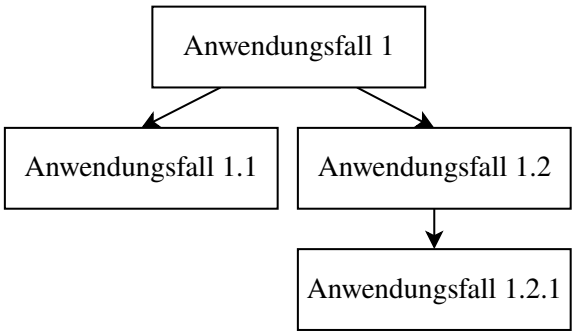


Abbildung 6.3: Relationenmodell eines Systems, in dem in einem Anwendungsfall das Gewicht und die Körpergröße bereitgestellt werden.

hat jedoch die Konsistenz eine hohe Priorität, um eine inkorrekte Datenweitergabe zu vermeiden. Durch die klar definierten Anwendungsfälle ist außerdem das Schema der Zone festgelegt. Es muss nur geändert werden, wenn neue Anwendungsfälle definiert werden. Bei der Definition neuer Anwendungsfälle muss das Schema also um die neuen Anwendungsfälle erweitert werden. Somit wird keine Datenbank mit einem so dynamischen Schema, wie es NoSQL Datenbanken besitzen, benötigt.

Innerhalb der relationalen Datenbank gibt es eine Tabelle, die eine Auflistung der Anwendungsfalle enthält. In ihr können auch zusätzliche Informationen zu den Anwendungsfällen gespeichert werden, zum Beispiel das Erstelldatum und die zugriffsberechtigten Personen bzw. Anwendungen. Des Weiteren existiert pro Datenkategorie eine Tabelle, in der die entsprechenden Daten gespeichert sind. Wenn zum Beispiel in zwei Anwendungsfällen das Gewicht vorkommt, existiert eine Tabelle, die Gewichtsdaten beinhaltet. Falls Anwendungsfälle nicht alle in der Tabelle gespeicherten Attribute beinhalten, werden diese Attribute mit NULL belegt. Die Tabellen mit den Daten enthalten den betroffenen Anwendungsfall als Foreign Key. In Abbildung 6.3 ist beispielhaft das Relationenmodell eines Systems gezeigt. In diesem System werden in einem Anwendungsfall das Gewicht und die Körpergröße bereitgestellt.

Die hierarchische Struktur der Anwendungsfälle ist durch eine weitere Tabelle gespeichert. Diese Tabelle enthält die Spalten ‚parent' und ‚child'. Diese sind jeweils über eine Foreign Key der Id des Anwendungsfalls zugeordnet. In Abbildung 6.4 ist links eine Baumstruktur aus Anwendungsfällen abgebildet. Rechts ist die entsprechende Tabelle dargestellt.



parent	**child**
Anwendungsfall 1	Anwendungsfall 1.1
Anwendungsfall 1	Anwendungsfall 1.2
Anwendungsfall 1.2	Anwendungsfall 1.2.1

Abbildung 6.4: Links: Baumstruktur aus Anwendungsfällen, rechts: Entsprechende Tabelle.

Kapitel 7

Prototypische Umsetzung

Im Rahmen einer prototypischen Implementierung soll nun die Umsetzbarkeit der zuvor entwickelten Konzepte gezeigt werden. Der Prototyp setzt die Speicherung der Rohdaten, deren Anreicherung und die Bereitstellung der angereicherten Daten in spezifizierten Anwendungsfällen um. Somit können, bezogen auf das in Abschnitt 2.1 beschriebene Szenario, Fitnessdaten verwaltet und verschiedenen Benutzergruppen zur Verfügung gestellt werden.

Nachfolgend werden verschiedene Aspekte der prototypischen Umsetzung beschrieben. Dabei wird zunächst auf die verwendeten Daten eingegangen (Abschnitt 7.1). Anschließend wird in Abschnitt 7.2 die Umsetzung der Architektur erläutert.

7.1 Verwendete Daten

Das System wird mit einem realen Datensatz getestet. Dieser ist an dem Szenario aus Abschnitt 2.1 orientiert. Die Daten werden mit einer Fitbit Charge HT gesammelt und beschreiben die Aktivität einer Person über ein Jahr hinweg. Ein Eintrag entspricht dabei einem Tag. [1] Der Datensatz wird durch künstlich generierte Daten über den Gewichtsverlauf der Person erweitert und die Größe der entsprechenden Person wird auf 1,80 m festgelegt. Durch diese zusätzlichen Daten kann neben der Aktivität der Person auch der BMI betrachtet werden. In Tabelle 7.1 ist ein beispielhafter Datensatz dargestellt.

Gewicht	Schritte	Dauer Aktivität 1	Dauer Aktivität 2	Dauer Aktivität 3
64.6	8391	136	36	30

Tabelle 7.1: Beispielhafter Datensatz.

[1] Die Daten sind https://www.kaggle.com/alketcecaj/one-year-of-fitbit-chargehr-data (Stand 16.07.2020) entnommen.

```
[
 {
   "_id": 1,
   "steps": 905
 },
 {
   "_id": 2,
   "steps": 18925
 }
]
```

Listing 7.1: Auszug aus einer JSON-Datei. Enthalten ist die Anzahl der Schritte pro Tag.

Es existieren insgesamt 366 Dateneinträge. Diese liegen aufgeteilt in mehrere JSON-Dateien vor. Eine Datei entspricht einer Datenkategorie, wie zum Beispiel der Schrittanzahl. In Listing 7.1 ist ein Auszug einer JSON-Datei dargestellt. Diese Datei enthält beispielhaft die Anzahl der Schritte pro Tag.

7.2 Umsetzung der Architektur

In diesem Abschnitt wird die Umsetzung der Architektur von RETORT innerhalb des Prototyps beschrieben. Als Programmiersprache wird Python[2] in der Version 3.8 gewählt. Python bietet im Vergleich zu anderen Programmiersprachen eine Vielzahl an Bibliotheken zur Unterstützung der Datenverarbeitung, wie zum Beispiel die Bibliotheken pandas[3] oder scikit-learn[4]. Daher wurde Python zu einer beliebten Programmiersprache für Datenanalyseaufgaben [McK18]. Die Verbreitung der Programmiersprache Python und dessen Eignung zur Datenanalyse und zum maschinellen Lernen sind der Grund, warum sie für diesen Prototyp gewählt wird. Dies unterstützt die Einbindung der Datenanalyse sowie des maschinellen Lernens in das System.

Die Architektur ist in Abbildung 7.1 veranschaulicht. Die verschiedenen datenverarbeitenden Zonen werden durch Python Module umgesetzt. Für die Rohdatenzone wird die dokumentenorientierte Datenbank MongoDB[5] verwendet und für die Zonen der Anwendungsfälle wird die relationale Datenbank MySQL[6] verwendet. Das Wissensmodell wird zunächst auch durch ein Python Modul umgesetzt. In einer Erweiterung des Prototyps wird es durch RDF/XML[7] und Berkeley DB[8] umgesetzt.

[2] https://www.python.org/

[3] pandas bietet umfangreiche Datenstrukturen und Funktionen für ein schnelles, einfaches und ausdrucksstarkes Arbeiten mit strukturierten oder tabellarischen Daten [McK18]. https://pandas.pydata.org/

[4] scikit-learn ist ein Machine-Learning-Toolkit für die Programmiersprache Python [McK18]. https://scikit-learn.org/

[5] https://www.mongodb.com

[6] https://www.mysql.com/

[7] http://www.w3.org/TR/2014/REC-rdf-syntax-grammar-20140225/

[8] https://www.oracle.com/database/technologies/related/berkeleydb.html

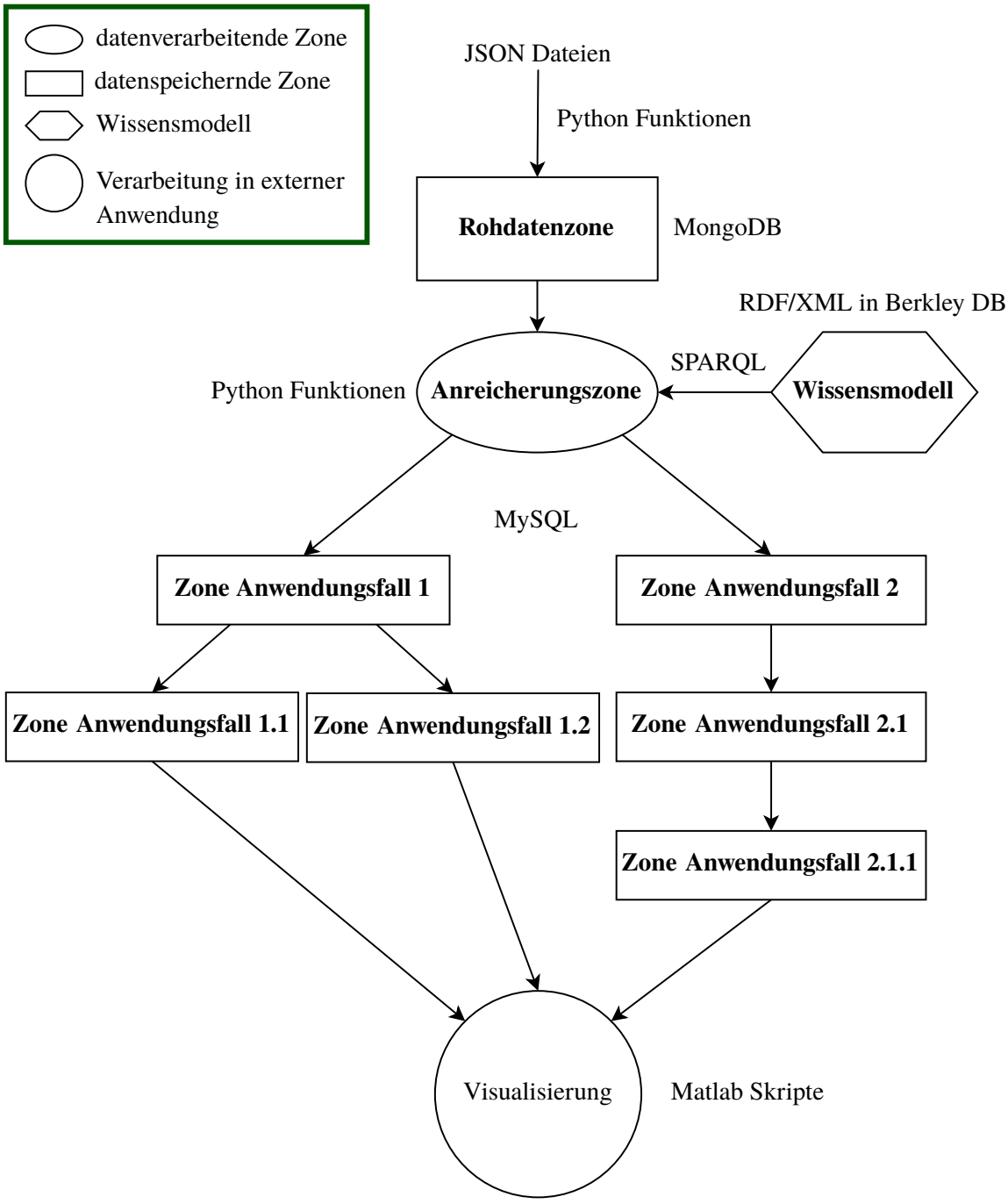


Abbildung 7.1: Architektur des Prototypen.

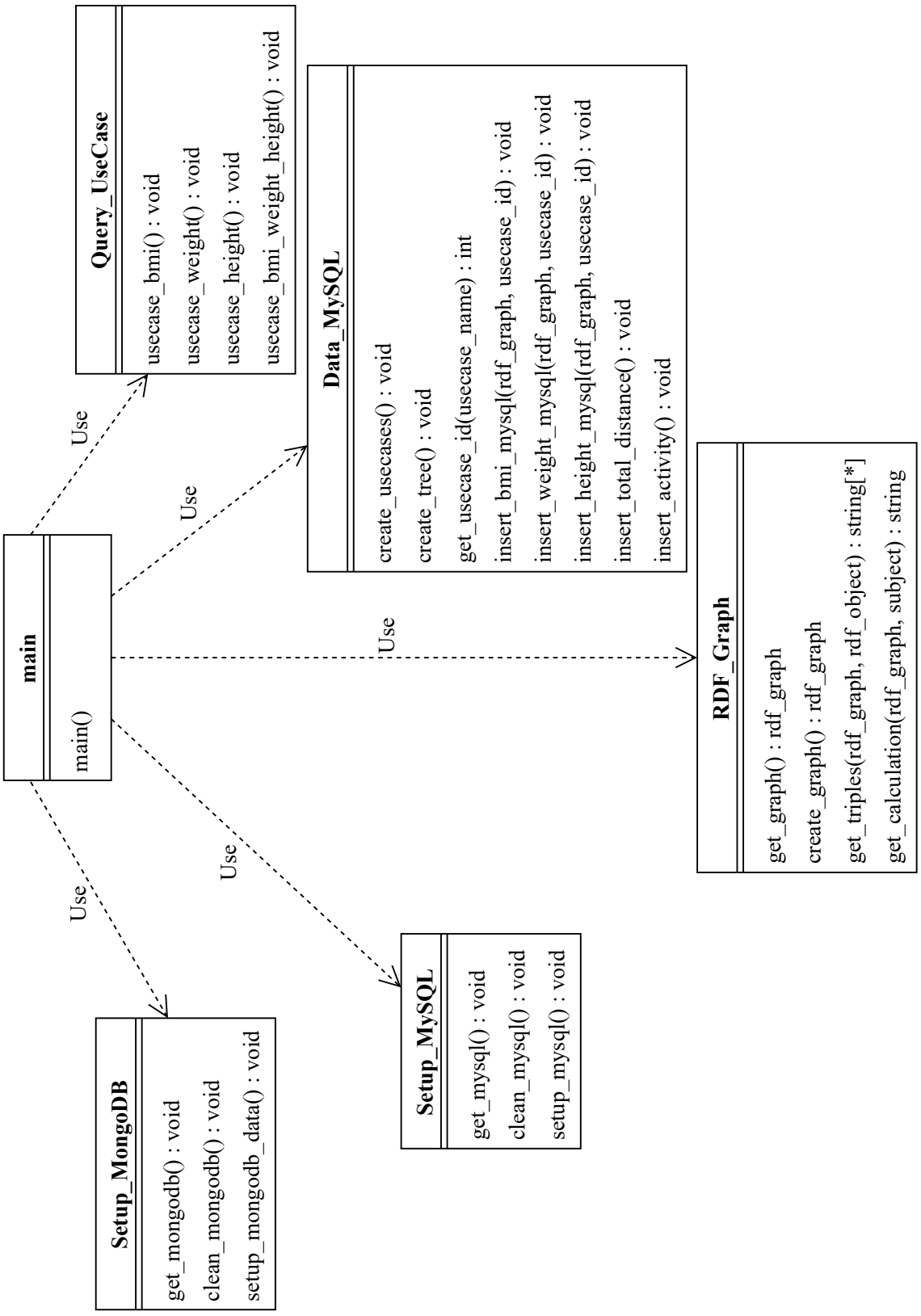


Abbildung 7.2: Auszug aus dem UML Klassendiagramm des Prototyps.

In Abbildung 7.2 ist ein Auszug aus dem UML Klassendiagramm des Prototyps dargestellt, das die Strukturierung des Prototyps in Python Module verdeutlicht. Die verwendeten Daten liegen, wie in Abschnitt 7.1 beschrieben als JSON-Dateien vor. Diese Daten werden durch das Python Modul `Setup_MongoDB` in der MongoDB (Rohdatenzone) gespeichert. Wie die Daten angereichert werden sollen, ist in einem weiteren Python Modul bzw. in der Berkeley DB (Wissensmodell) festgehalten. Die Implementierung hierfür erfolgt im Python Modul `RDF_Graph`. Die Anreicherung der Daten (Anreicherungszone) sowie Speicherung der angereicherten Daten in der MySQL Datenbank (Zonen der Anwendungsfälle) erfolgt im Python Modul `Data_MySQL`. Hier wird die Anreicherung mithilfe mehrerer Python Funktionen durchgeführt. Zur Abfrage des Wissensmodells müssen die entsprechenden Python Funktionen aufgerufen werden bzw. die Berkeley DB mit SPARQL abgefragt werden. Die angereicherten Daten werden anschließend in der MySQL Datenbank gespeichert. Innerhalb des Python Moduls `Query_UseCase` sind Funktionen enthalten, welche die Daten der MySQL Datenbank entsprechend der Anwendungsfälle als CSV-Dateien bereitstellen. Wenn externe Anwendungen die Daten der Zonen der Anwendungsfälle abfragen, werden die Daten durch dieses Python Modul bereitgestellt. Diese Daten können nun zum Beispiel mithilfe von MATLAB[9] Skripten visualisiert werden. Innerhalb des Moduls `Query_UseCase` geschieht somit die Anpassung der Daten an das Dateiformat, welches die externen Anwendungen benötigen. Auch kann hier eine Zugriffskontrolle durchgeführt werden.

Im Folgenden wird genauer auf die einzelnen Zonen der Architektur eingegangen. Zuerst wird in Abschnitt 7.2.1 die Umsetzung der Rohdatenzone beschrieben. Anschließend wird erklärt wie das Wissensmodell und die Anreicherungszone umgesetzt werden (Abschnitt 7.2.2). Zum Schluss folgt die Beschreibung der Implementierung der Zonen der Anwendungsfälle (Abschnitt 7.2.3).

7.2.1 Rohdatenzone

Für die Rohdatenzone wird die dokumentenorientierte Datenbank MongoDB[10], Version 4.2 verwendet. Diese ist die bekannteste bzw. beliebteste dokumentenorientierte Datenbank [sol20a] und wird deshalb hier verwendet.

Innerhalb der MongoDB werden die Dokumente im BSON-Format (Binary JSON) gespeichert [Kle16]. Dies ermöglicht die Speicherung von verschachtelten Daten, wobei die Daten trotz Verschachtlung abfragbar und indizierbar sind. Somit können verschachtelte Daten oder Hierarchien verwaltet werden. Durch die Speicherung der Dokumente in einem JSON-ähnlichen Format behandelt MongoDB den Big-Data Aspekt der Variety: Es können semistrukturierte Daten gespeichert werden [Tre14]. Die grundlegende Struktur einer MongoDB-Instanz wird anhand der Struktur der für die Implementierung des Prototypen verwendete MongoDB-Instanz im Folgenden erläutert. Diese ist in Abbildung 7.3 dargestellt. Ein Server (mongod in der Abbildung) verwaltet mehrere Datenbanken (z. B. data lake retort in der Abbildung). Diese enthalten mehrere Sammlungen an Dokumenten, auch Collections genannt (z. B. steps in der Abbildung). Die Collections stellen dabei keine technischen Anforderungen an die in ihnen enthaltenen Dokumente. Somit ist MongoDB schemalos [Tre14]. Innerhalb einer Collection können sich beliebig viele Dokumente befinden [Kle16].

[9] https://de.mathworks.com/products/matlab.html
[10] https://www.mongodb.com

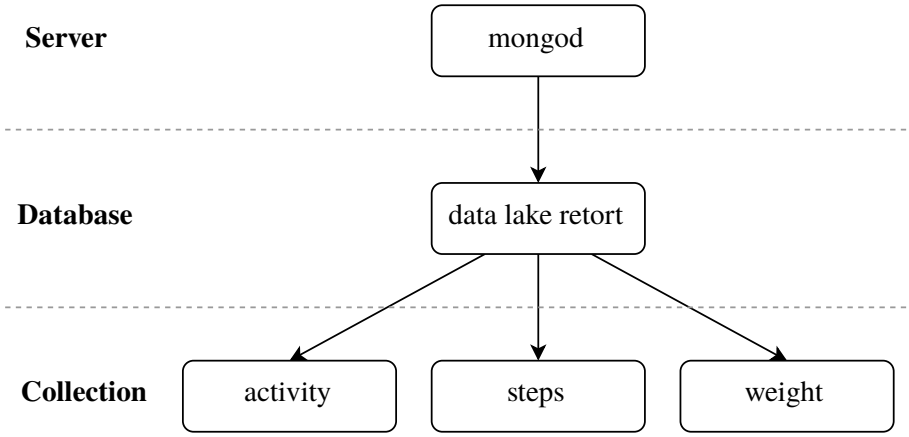


Abbildung 7.3: Struktur der verwendeten MongoDB-Instanz.

Weight

id	weight [kg]

Steps

id	steps

Activity

id	date	activity	duration [min]

Tabelle 7.2: Struktur der Collections der verwendeten MongoDB-Instanz.

Wie in Abschnitt 7.1 erwähnt, liegen die zu speichernden Daten innerhalb dieses Prototyps bereits in mehreren JSON-Dateien vor. Das in Abschnitt 7.1 erwähnte Beispiel einer dieser JSON-Dateien ist in Listing 7.1 dargestellt. Es wird ein Auszug der JSON-Datei dargestellt, welche die Anzahl der Schritte pro Tag enthält. Die Daten der JSON-Dateien werden durch ein für diese Anwendung entwickeltes Python-Modul eingelesen und in der MongoDB persistiert. Dabei wird jede Datenkategorie in einer eigenen Collection abgelegt. Ein Beispiel einer Datenkategorie ist die Schrittanzahl, welche in der Collection steps abgelegt wird. Somit ergibt sich die in Abbildung 7.3 dargestellte Struktur. Die Struktur der Collections innerhalb der MongoDB ist in Tabelle 7.2 dargestellt. In dieser Anwendung hat jedes Dokument innerhalb einer Collection die gleiche Struktur. Dies ergibt sich durch die Zerlegung der Daten in Datenkategorien und die Verwendung einer Collection pro Datenkategorie. So sind in einer Collection nur Daten gleicher Struktur enthalten.

7.2.2 Wissensmodell und Anreicherungszone

Aus den verwendeten Daten lassen sich durch Verknüpfung Informationen und Wissen ableiten. Dies ist im Entwurf von RETORT in Kapitel 5 beschrieben. Die Information, wie die Daten verknüpft werden können und welche Informationen durch die Verknüpfung entstehen, sind innerhalb des

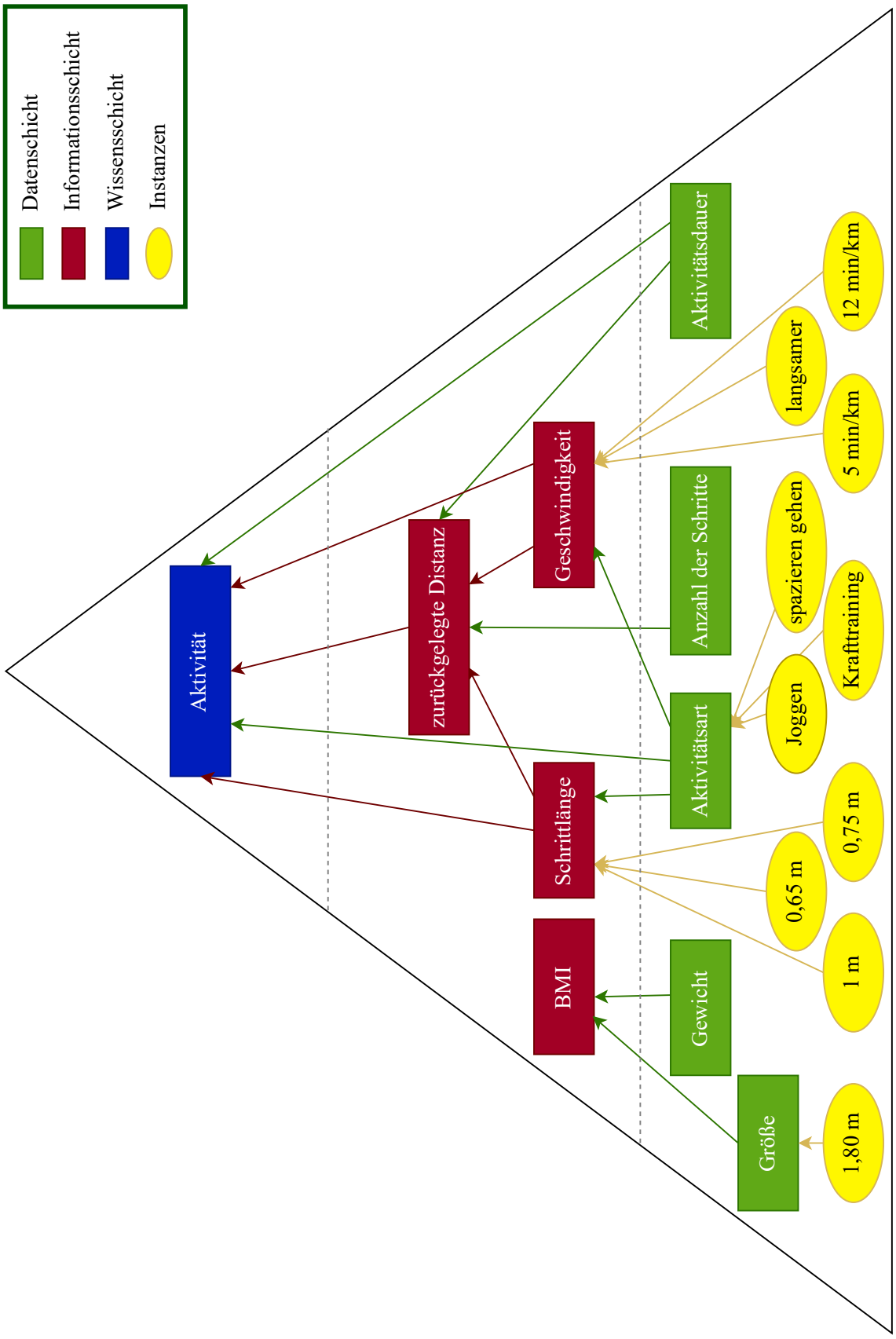


Abbildung 7.4: Wissensmodell des Prototyps.

Wissensmodells modelliert. In Abbildung 7.4 ist das Wissensmodell des Prototyps dargestellt. In diesem wird unter anderem aus dem Gewicht und der Körpergröße der Body Mass Index (BMI) über die Formel[11] $\text{BMI} = \frac{\text{Gewicht [kg]}}{(\text{Größe [m]})^2}$ berechnet. Somit wird eine neue Information generiert.

In Abschnitt 6.1 wurde beschrieben, dass das Wissensmodell mithilfe von RDF umgesetzt wird. Die am häufigsten verwendete RDF Syntax ist eine XML-basierte Schreibweise, da praktisch jede gebräuchliche Programmiersprache Unterstützung bei der Verarbeitung von XML anbietet. Diese Sprache wird RDF/XML genannt. Die Unterschiede der Datenmodelle von XML (Bäume) und RDF (Graphen) sind dabei kein Hindernis, da mit XML lediglich die syntaktische Struktur der Kodierung eines RDF Graphen vorgegeben wird. Durch den hierarchischen Aufbau von XML muss die Kodierung der RDF Tripel auch hierarchisch erfolgen. Die Tripel werden deshalb in RDF/XML nach dem Subjekt gruppiert [Hit08].

In Listing 7.2 ist die Ontologie aus Abbildung 5.3 in RDF/XML dargestellt. Diese beschreibt die Berechnung des BMI aus der Größe und dem Gewicht. In Zeile 2 und 3 werden die Namespaces `rdf` und `ex` definiert. Diese Definition vereinfacht den späteren Code. Namespaces definieren die in ihnen erlaubten Elemente (`Description`) und deren Attribute (`about`). Der Namespace `ex` enthält dabei eine selbst definierte Struktur an Elementen (hier: `TeilVon` und `Schicht`), die spezifisch für die Struktur der dargestellten Ontologie ist. Anschließend werden nach Subjekt gruppiert die Tripel beschrieben. Das Element `rdf:Description` beschreibt ein Subjekt mit zugehörigen Tripeln. Mit `rdf:about` wird das Subjekt benannt. In Zeile 6 wird das Prädikat Teil_von mit `ex:partOf` benannt. Anschließend durch `rdf:ressource` das Objekt.

Die in RDF/XML beschriebenen Daten können mit der RDF Abfragesprache SPARQL abgefragt und bearbeitet werden. Diese Sprache ist eine W3C Empfehlung [The13].

In einer ersten Version des Prototyps wird das Wissensmodell als Python Modul implementiert, nicht mit RDF/XML. In dieser Version liegt das Augenmerk auf der Datenverwaltung und -verarbeitung und nicht auf einer strukturierten Repräsentation des Wissens. Innerhalb des Python Moduls, welches das Wissensmodell darstellt, entspricht eine Verknüpfung von Daten einer Funktion. Die Funktion erhält die zu verknüpfenden Daten als Parameter und der Rückgabewert der Funktion ist die neue Information. Mithilfe dieser Funktionen können in der Anreicherungszone aus den vorhandenen Daten neue Informationen abgeleitet werden. Die Funktion zur Berechnung des BMI ist beispielhaft in Listing 7.3 dargestellt. In einem weiteren Python-Modul ist beschrieben, welche Funktionen zur Anreicherung der Daten aufgerufen werden müssen. Dabei ist die Folge von Funktionsaufrufen für einen Anwendungsfall in einer Funktion zusammengefasst, sodass pro Anwendungsfall nur die entsprechende Funktion aufgerufen werden muss. Diese Funktion, die beschreibt welche Funktionen zur Berechnung des BMI aufgerufen werden müssen, ist in Listing 7.4 dargestellt. Sie speichert die Daten auch in einer MySQL Datenbank. Darauf wird im folgenden Abschnitt 7.2.3 genauer eingegangen.

In einer Erweiterung des Prototyps ist das Wissensmodell in RDF mithilfe des Python-Pakets RDFLib[12] umgesetzt. Dieses Paket erleichtert die Erstellung und Abfrage von RDF Graphen und bietet hierfür verschiedene Methoden. In Listing 7.5 ist beispielhaft ein Python Skript zur Erstellung eines RDF Graphen zu sehen. Dieser beschreibt die Berechnung des BMI. Der Aufruf in Zeile 24 hat Listing 7.2 als Ausgabe. In Zeile 27 wird beispielhaft abgefragt, welche Daten für die Formel

[11] Formel siehe https://www.apotheken-umschau.de/bmi-rechner (Stand 13.09.2020)
[12] https://rdflib.readthedocs.io/en/stable/

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:ex="http://example.org/">

  <rdf:Description rdf:about="http://example.org/weight">
    <ex:partOf rdf:resource="http://example.org/bmi"/>
    <ex:layer>Data Layer</ex:layer>
  </rdf:Description>

  <rdf:Description rdf:about="http://example.org/height">
    <ex:partOf rdf:resource="http://example.org/bmi"/>
    <ex:layer>Data Layer</ex:layer>
  </rdf:Description>

  <rdf:Description rdf:about="http://example.org/bmi">
    <ex:layer>Information Layer</ex:layer>
    <ex:calculation>bmi = weight / pow(height, 2)</ex:calculation>
  </rdf:Description>

</rdf:RDF>
```

Listing 7.2: Ontologie aus Abbildung 5.3 zur Berechnung des BMI in RDF/XML.

```python
def calculate_bmi(weight, height):
    bmi = weight / pow(height, 2)
    return bmi
```

Listing 7.3: Python Funktion zur Berechnung des BMI.

```python
def insert_bmi_mysql():
    db = get_mysql()
    cursor = db.cursor()

    sql = "INSERT INTO bmi (bmi, weight, height) VALUES (%s, %s, %s)"
    val = list()

    height = get_height()

    for entry in get_mongodb().weight.find():
        val.append((calculate_bmi(entry["weight"], height), entry["weight"], height))

    cursor.executemany(sql, val)

    db.commit()
```

Listing 7.4: Python Funktion zur Anreicherung von Größe und Gewicht und Speicherung des BMIs, der Größe und des Gewichts in der MySQL Datenbank.

```python
from rdflib import Graph, Literal, RDF, URIRef, Namespace

# create a Graph
graph = Graph(identifier="Knowledge_Model_RETORT")

# create a namespace
ex = Namespace("http://example.org/")
graph.bind("ex", "http://example.org/")

# add triples
graph.add((ex.weight, ex.layer, Literal("Data Layer")))
graph.add((ex.height, ex.layer, Literal("Data Layer")))

graph.add((ex.weight, ex.partOf, ex.bmi))
graph.add((ex.height, ex.partOf, ex.bmi))

graph.add((ex.bmi, ex.calculation, Literal("bmi = weight / pow(height, 2)")))
graph.add((ex.bmi, ex.layer, Literal("Information Layer")))

# print the entire Graph in the RDF/XML format
print(graph.serialize(format="xml").decode("utf-8"))

# get all subjects of triples with predicate ex.partOf and object ex.bmi
data = graph.subjects(ex.partOf, ex.bmi)

# print the URIs of the subjects
for entry in data:
    print(entry)
```

Listing 7.5: Python Skript zur Erstellung eines RDF Graphen zur Beschreibung der Berechnung des BMI.

zur Berechnung des BMI benötigt werden. Der Aufruf in Zeile 31 liefert daraufhin die Ausgabe `http://example.org/weight` und `http://example.org/height`. Auf diese Weise können die benötigten Informationen für die Anreicherungszone aus dem Wissensmodell abgefragt werden. Die Ergebnisse der Abfragen werden mithilfe eines Python Moduls den entsprechenden Datenbankabfragen der MongoDB zugeordnet. Die benötigten Rohdaten können aus der MongoDB abgefragt und entsprechend dem Wissensmodell verarbeitet werden.

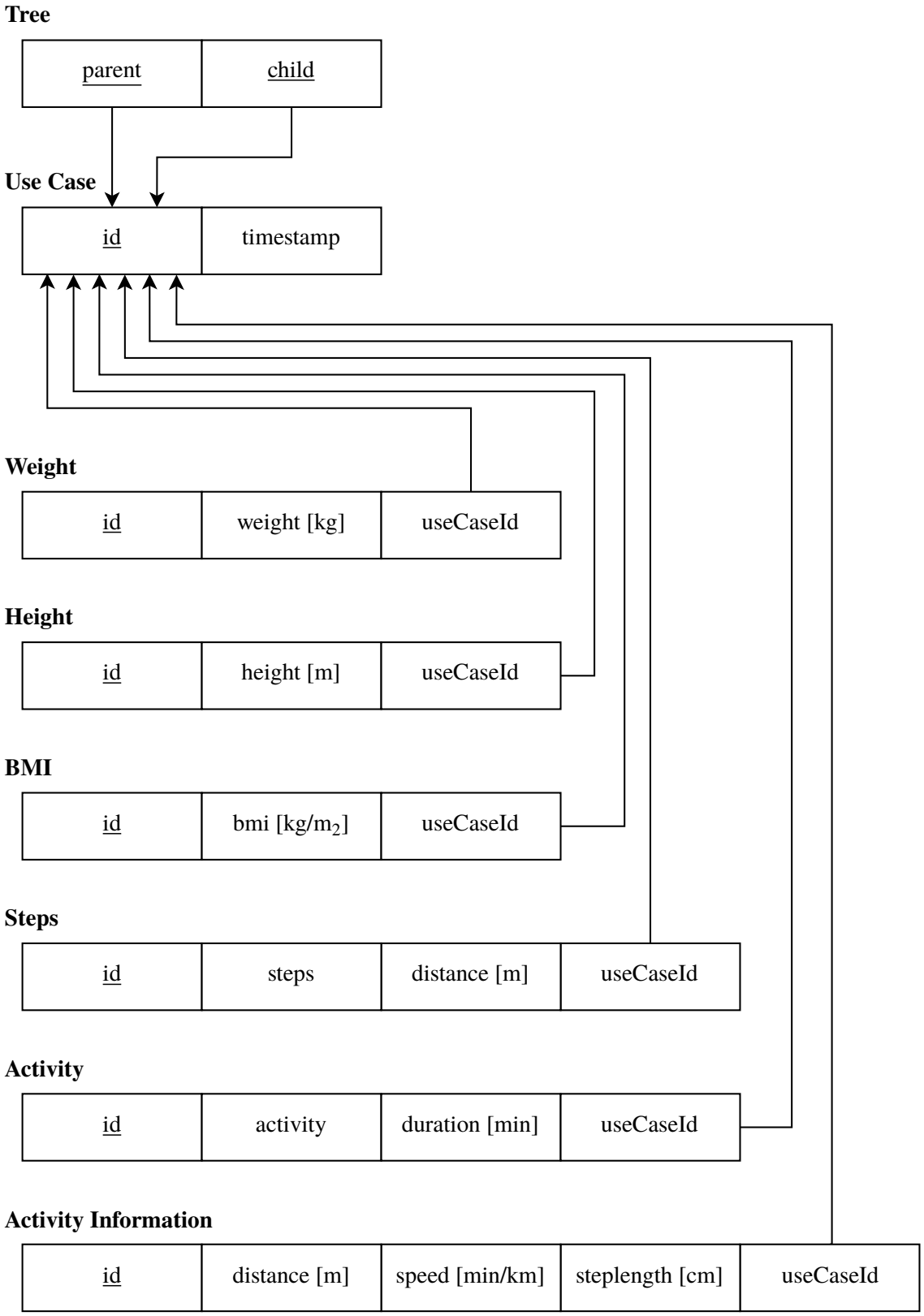


Abbildung 7.5: Relationales Modell der MySQL Datenbank.

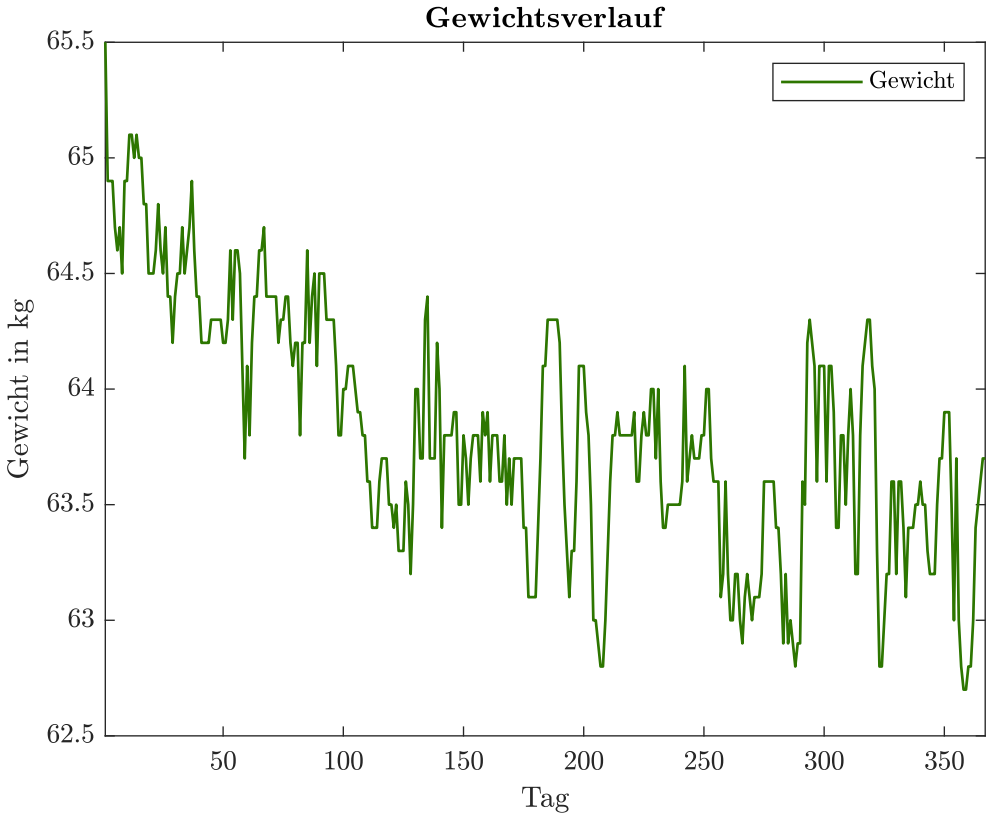


Abbildung 7.6: Darstellung des Gewichtsverlaufs mit täglichen Werten.

7.2.3 Anwendungsfälle

Die angereicherten Daten werden in der relationalen Datenbank MySQL[13], Version 5.7 persistiert. Dabei handelt es sich um eine verbreitete relationale Datenbank [sol20c]. Das relationale Modell der verwendeten MySQL Datenbank ist in Abbildung 7.5 dargestellt.

Innerhalb des Prototyps werden die Daten der Anwendungsfälle, die in der MySQL Datenbank gespeichert sind, externen Anwendungen beispielhaft als CSV-Dateien bereitgestellt. An dieser Stelle ist zum Beispiel auch eine Zugriffskontrolle möglich.

Die Endanwender können die bereitgestellten Daten der Anwendungsfälle zum Beispiel visualisieren. Dies lässt sich mithilfe von MATLAB[14] Skripten umsetzen. Ein Anwendungsfall ist zum Beispiel die Bereitstellung von allen Gewichtsdaten einer Person. Eine Visualisierung dieser Daten ist in Abbildung 7.6 dargestellt.

[13] https://www.mysql.com/
[14] https://de.mathworks.com/products/matlab.html

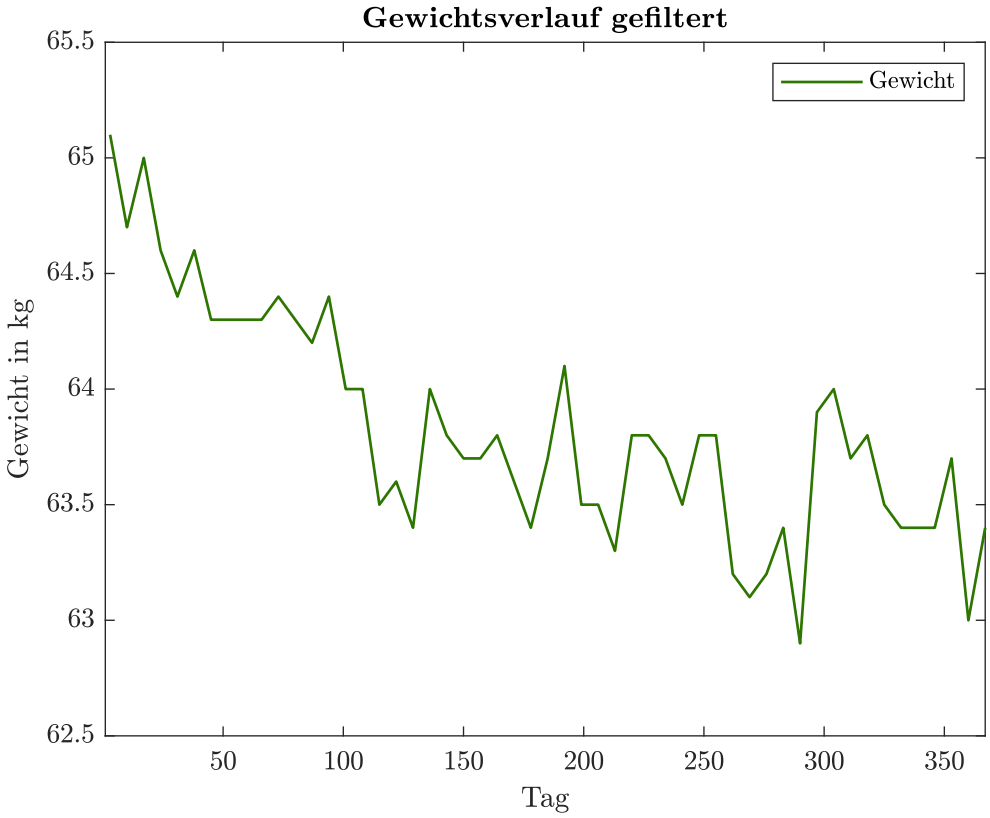


Abbildung 7.7: Darstellung des Gewichtsverlaufs mit wöchentlichen Werten.

Innerhalb eines anderen Anwendungsfalles werden diese Gewichtsdaten nur zeitlich gefiltert bereitgestellt. Es wird statt dem täglichen Wert des Gewichts nur ein Wochendurchschnitt zur Verfügung gestellt. Dies entspricht Anforderung A5 (Reduktion der Aussagekraft der Daten). In Abbildung 7.7 ist der Gewichtsverlauf mit reduzierter zeitlicher Auflösung dargestellt.

KAPITEL 8

Evaluation

In diesem Kapitel wird RETORT evaluiert. Da RETORT auf datenintensive IoT-Anwendungen ausgelegt ist, soll es in der Lage sein Big Data zu verwalten. Deshalb wird zunächst in Abschnitt 8.1 untersucht, wie RETORT die Herausforderungen der Verarbeitung von Big Data löst. In Abschnitt 8.2 wird anschließend untersucht, wie die in Abschnitt 2.3 definierten Anforderungen in RETORT umgesetzt werden.

8.1 Verarbeitung von Big Data

RETORT ist eine Datenbereitstellungsplattform für datenintensive IoT-Anwendungen. Dabei stellt vor allem die große Datenmenge eine Herausforderung an die Plattform dar, sowie die damit einhergehende unterschiedliche Strukturierung der Daten. Allerdings müssen auch die anderen in Abschnitt 3.1 genannten Herausforderungen der Verarbeitung von Big Data gelöst werden.

In diesem Abschnitt wird untersucht, wie RETORT die Herausforderungen der Verarbeitung von Big Data löst. Dabei wird nacheinander auf die fünf Eigenschaften von Big Data eingegangen, welche in Abschnitt 3.1 genauer beschrieben wurden.

Volume. Volume bezieht sich auf die Datenmenge, die mit herkömmlichen Speicher- und Analysemethoden kaum zu verarbeiten ist.

Um diese große Datenmenge speichern zu können, verwendet RETORT einen Data Lake. Dieser ist darauf ausgelegt, mit großen Datenmenge umzugehen [MT16]: Data Lakes werden so konzipiert, dass sie (unbegrenzt) skaliert werden können, ohne dass die Leistung wesentlich beeinträchtigt wird [Gor19].

Das Konzept eines Data Lake wird in dieser Arbeit für die Verwendung mit großen Datenmengen angepasst. Die Umsetzung der Rohdatenzone mit einer NoSQL Datenbank ermöglicht die Verarbeitung von großen Datenmengen [HELD11]. Die Rohdaten werden dabei in einzelne Komponenten zerlegt gespeichert. Dies unterstützt die Bereitstellung einzelner Attribute der Daten, sowie die neuen Rekombinationen mit Daten anderer Quellen. Dadurch kann vor der Datenbereitstellung eine Datenreduktion durchgeführt werden. Dies bedeutet, dass zum Beispiel eine zeitliche Filterung

durchgeführt werden kann, unter anderem durch eine Beschränkung auf die Daten eines Zeitintervalls, zum Beispiel des letzten Monats, oder eine Reduktion der zeitlichen Auflösung. Auch das Herausfiltern bestimmter Tupel bzw. die Beschränkung der bereitgestellten Attribute kann das Datenvolumen reduzieren. Durch die Reduktion des Datenvolumens sind externe Anwendungen, die nicht mit der Eigenschaft Volume von Big Data umgehen können, in der Lage diese Daten zu verarbeiten.

Durch die Verwendung eines Data Lake, welcher eine NoSQL Datenbank zur Speicherung der großen Datenmenge beinhaltet, kann RETORT mit der Herausforderung Volume von Big Data umgehen. Die Datenreduktion vor der Bereitstellung der Daten ermöglicht es auch externen Anwendungen diese Daten zu verwenden, ohne in der Lage sein zu müssen, die großen Datenmenge verwalten zu können.

Velocity. Velocity hat zwei Komponenten: Zum einen die hohe Dateneingangsgeschwindigkeit, also die hohe Geschwindigkeit, mit der die Daten generiert werden, zum anderen die hohe Datenausgangsgeschwindigkeit, also die Geschwindigkeit, mit der die Daten verarbeitet werden müssen, um zum Beispiel Echtzeitdatenverarbeitung zu ermöglichen.

Die hohe Dateneingangsgeschwindigkeit wird in RETORT durch die Verwendung einer Landezone adressiert. In dieser Zone werden die Daten gepuffert, sodass die nachfolgende Rohdatenzone die Daten nacheinander abarbeiten kann. Dadurch kann RETORT auch mit temporal begrenzten Anstiegen in der Dateneingangsgeschwindigkeit umgehen.

Eine Echtzeitdatenverarbeitung ist mit dem vorgestellten Konzept für RETORT durch das Caching (siehe Abschnitt 6.2) nicht möglich. Die angereicherten Daten werden in den Zonen der Anwendungsfälle gespeichert, um nicht bei jedem Abrufen der Daten wiederholt die Anreicherung durchführen zu müssen. Dadurch sind neue Daten nicht automatisch sofort in den Zonen der Anwendungsfälle sichtbar, sondern erst, nachdem die gespeicherten Daten aktualisiert wurden. Für die Echtzeitdatenverarbeitung müsste bei jedem Abruf der Zone die Daten neu berechnet werden.

Innerhalb des gewählten Anwendungsszenarios der Fitnessakte ist Echtzeitdatenverarbeitung jedoch nicht elementar. Die Sammlung der Fitnessdaten verliert durch die fehlende Echtzeitdatenverarbeitung nicht an Gültigkeit. Deshalb wird in dieser Arbeit die Echtzeitdatenverarbeitung nicht vorausgesetzt, sondern das Thema der effizienten Umsetzung einer Plattform mit Echtzeitdatenverarbeitung zukünftigen Arbeiten überlassen. Neben der Umsetzung der Echtzeitdatenverarbeitung kann zum Beispiel ein Bereinigungskonzept entworfen werden um das Wachstum der gespeicherten Datenmenge einzudämmen. Dabei kann es sich um ein Konzept handeln, das Echtzeitdaten, die ihre Gültigkeit verloren haben und nicht mehr benötigt werden, löscht. Auch kann an diese Daten eine Lebensdauer annotiert werden, die die Aufbewahrungszeit innerhalb der Zonen der Anwendungsfälle beschreibt, sodass ungültige oder veraltete Daten nicht bereitgestellt werden.

Variety. Variety bezieht sich auf die unterschiedliche Strukturierung der Daten durch die unterschiedliche Herkunft und die verschiedenen Typen der Daten. Unterschiedliche IoT Geräte können unterschiedliche Schemata besitzen und die Daten in unterschiedlichen Einheiten bereitstellen.

Um diese Daten ohne Datenverlust speichern zu können verwendet RETORT einen Data Lake. Dieser bietet die nötige Flexibilität bezüglich der sich in ihm befindlichen Daten [Fan15]. Die Daten können gespeichert werden, ohne sie zuerst strukturieren zu müssen. Dies wird unterstützt durch

die Verwendung einer NoSQL Datenbank zur Speicherung der Rohdaten. NoSQL Datenbanken ermöglichen es, dass Daten mit unterschiedlicher Struktur in ihnen gespeichert werden [OBLB17], indem zum Beispiel die Daten eine Beschreibung ihrer Struktur selbst enthalten [ABS14].

Durch die Aufbereitung der Daten können diese Unterschiede in Struktur und Format angeglichen werden sowie in das für die jeweilige externe Anwendung benötigte Ausgangsformat konvertiert werden. Dies geschieht in der Anreicherungszone von RETORT. Die Daten können entsprechend der Anwendung, für die sie bereitgestellt werden sollen, angepasst werden. Somit müssen externe Anwendungen nicht die Herausforderung der unterschiedlichen Strukturierung der Daten meistern und das Eingangsschema nicht in die benötigte Strukturierung für die Verarbeitung übertragen.

Zusammengefasst wird die Herausforderung Variety durch eine angepasste Technologie und eine Aufbereitung der Daten bewältigt.

Veracity. Veracity beschreibt die unterschiedliche Vertrauenswürdigkeit der Daten aufgrund ihrer unterschiedlichen Herkünfte.

Dieser Aspekt von Big Data und dessen Umsetzung in RETORT ist kein Schwerpunkt dieser Arbeit, da innerhalb des Anwendungsszenarios der Fitnessakte der Benutzer die Datenhoheit hat. Er entscheidet also wer seiner Akte Daten hinzufügen darf. In diesem Fall kann der Benutzer über die Vertrauenswürdigkeit der Quellen entscheiden und kontrollieren welche Daten seiner Akte hinzugefügt werden.

Die Entwicklung eines konkreten Konzepts wird zukünftigen Arbeiten überlassen. Eine Möglichkeit mit der unterschiedlichen Vertrauenswürdigkeit von Datenquellen umzugehen ist, eine Referenz auf die Datenquelle bei den Daten zu speichern. So kann bei der Verarbeitung der Daten entschieden werden, wie vertrauenswürdig diese sind.

Um fehlerhafte Daten auszuschließen, können Wertebereiche definiert werden, in denen der gemessene Wert liegen muss, um gültig zu sein. Dabei muss jedoch auf korrekte Schwellenwerte geachtet werden, um auftretende Anomalien bei den betreffenden Personen nicht als falsche Messwerte einzuordnen.

Eine Möglichkeit sicherzustellen, dass die Daten vor der Übertragung nicht manipuliert werden können und die Integrität der Daten gewährleistet ist, besteht darin, dass sich Sensoren vor der Übermittlung von Daten authentifizieren müssen. Dabei wird überprüft, dass die Software und Hardware der Datenquelle nicht manipuliert wurden und die übermittelten Daten dadurch legitim sind (vgl. [SGPM20]). Dies kann zum Beispiel durch eine digitale Signatur geschehen. Gritti et al. [GOM18] schlagen hierzu einen Ansatz für das IoT vor.

Diese Ansätze sind beispielhafte Möglichkeiten, wie die Vertrauenswürdigkeit der Daten überprüft und sichergestellt werden kann.

Value. Value beschreibt den potenziellen Wert der Daten, wenn sie verarbeitet wurden.

„Daten sind das Öl des 21. Jahrhunderts“ [Ste18] ist ein häufig gezogener Vergleich, da Daten heutzutage einen großen Wert besitzen. Genau wie Öl, müssen diese Rohdaten zunächst allerdings raffiniert werden, um als Grundlage für verschiedene Anwendungen gewinnbringend nutzbar zu sein [Bra14].

In RETORT ist diese Aufbereitung der Daten durch das Wissensmodell beschrieben. Dieses basiert auf der DIKW Pyramide. Dabei handelt es sich um ein Modell, welches beschreibt wie Wissen entsteht. Im Wissensmodell von RETORT ist festgehalten, wie die verfügbaren Daten verknüpft und angereichert werden können. Auch ist beschrieben, welche Wissensmuster daraus ableitbar sind. Somit beschreibt das Wissensmuster den Wert der Daten und wie dieser generiert werden kann.

Aufbereitet werden die Daten in der Anreicherungszone. In dieser Zone wird also der Wert der Daten durch die im Wissensmodell beschriebenen Anreicherungen und Verknüpfungen erzeugt.

8.2 Anforderungsevaluation

In diesem Abschnitt wird untersucht, ob das in Kapitel 6 vorgestellte Konzept mit der in Kapitel 6 vorgestellten Implementierungsstrategie die in Abschnitt 2.3 definierten Anforderungen erfüllt.

Anforderung A1 — Flexibilität. Anforderung A1 fordert, dass die Datenbereitstellung an verschiedene Nutzerbedürfnisse anpassbar sein soll. Die Daten sollen an die Benutzergruppe angepasst zur Verfügung gestellt werden.

In RETORT werden die Daten dem Endnutzer in verschiedenen Anwendungsfällen bereitgestellt. Dabei sind die Anwendungsfälle an die Bedürfnisse der Benutzergruppe angepasst. Es wird zum einen ein angepasster Auszug der Daten bereitgestellt und zum anderen werden die Daten entsprechend dem Anwendungsfall aufbereitet. Dies wird durch die individuelle Aufbereitung der Daten pro Anwendungsfall ermöglicht. Da die Daten im Rohformat gespeichert werden und so während des Speicherprozesses keine Daten verloren gehen, ist es auch möglich, diese in unterschiedliche Formate umzuwandeln und so das Format der Daten der Benutzergruppe anzupassen. Somit wird Anforderung A1 erfüllt.

Anforderung A2 — Erweiterbarkeit. Anforderung A2 fordert, dass das System an neue Bedürfnisse anpassbar sein soll. Es sollen zum Beispiel neue Anwendungsfälle definierbar sein, die nicht von vornherein bekannt waren.

Die Daten werden innerhalb von RETORT im Rohformat gespeichert. Dadurch gehen beim Speicherprozess keine Informationen verloren. Ohne diese Flexibilität bezüglich der Heterogenität im Format und der Struktur der Daten müsste vor der Speicherung eine Aufbereitung und Transformation der Daten durchgeführt werden, während der Informationen verloren gehen können. Auch können neue Anwendungsfälle definiert werden, die Eigenschaften der Daten verwenden, die bis dahin noch nicht benötigt wurden.

Die neuen Anwendungsfälle können der Ontologie des Wissensmodells hinzugefügt werden, da diese erweiterbar ist. Durch die zentrale Definition des Wissens wird auch die Erweiterung und Pflege bereits definierter Anwendungsfälle unterstützt. Soll zum Beispiel eine genauere Formel zur Berechnung des BMI verwendet werden, muss diese Formel nicht in allen Anwendungsfällen angepasst werden, sondern nur in der Ontologie des Wissensmodells.

Die Strukturierung der Anwendungsfälle in mehreren Baumstrukturen, erleichtert es zudem neue Anwendungsfälle zu definieren, da bereits definierte Teilbäume als Grundlage für neue Anwendungsfälle wiederverwendet werden können. Diese Struktur erleichtert es auch bereits bestehende Anwendungsfälle anzupassen: Durch das Entfernen von Daten aus dem Anwendungsfall der Wurzel, werden diese Daten auch in allen Anwendungsfällen der Kindknoten nicht mehr bereitgestellt. So müssen die Daten nicht aus allen Anwendungsfällen einzeln entfernt werden.

Diese verschiedenen Eigenschaften von RETORT erfüllen Anforderung A2.

Anforderung A3 — Einschränkung der Datenweitergabe. Anforderung A3 fordert, dass es möglich sein soll, die Datenweitergabe einzuschränken und sie an einen klar definierten Zweck gebunden sein soll. Diese Anforderung legt das Augenmerk auf die Einschränkung des Zugriffs auf die Daten.

Externe Anwendungen haben keinen Zugriff auf die in RETORT im Rohformat gespeicherten Daten. Der Zugriff ist nur auf die Daten in ausgewählte Anwendungsfälle gewährt. Diese Anwendungsfälle sind klar definiert: Es ist festgelegt welche Daten wie bereitgestellt werden. Die Daten der einzelnen Anwendungsfälle können sich somit zum Beispiel darin unterscheiden, dass unterschiedliche Teilmengen an Attributen bereitgestellt werden. Die bereitgestellte Datenmenge und das Format der bereitgestellten Daten kann sich auch unterscheiden.

Die Zonen der Anwendungsfälle, in denen die Daten bereitgestellt werden, sind voneinander abgegrenzt. Externe Anwendungen haben nur Zugriff auf ausgewählte Anwendungsfälle, der Zugriff auf Daten anderer Anwendungsfälle ist nicht erlaubt. Die Zugriffserlaubnis externer Anwendungen wird durch eine Zugriffskontrolle verwaltet, die die dritte Komponente von RETORT darstellt. Externe Anwendungen können also nur auf die Daten der Anwendungsfälle zugreifen, für die der Zugriff erlaubt ist.

Durch dieses Konzept wird Anforderung A3 erfüllt.

Anforderung A4 — Anwendungsbezogene Reduktion. Anforderung A4 fordert, dass die Art der Datenreduktion an den Anwendungsfall angepasst sein soll.

In RETORT sind die Anwendungsfälle in mehreren Baumstrukturen angeordnet. Ein Baum entspricht dabei einer Kategorie der Anwendungsfälle. Ein Abstieg innerhalb der Baumstruktur verfeinert die Anwendungsfälle. Anwendungsfälle, die sich näher an der Wurzel befinden beinhalten mehr bzw. genauere Daten. Somit wird eine subtraktive Filterung durchgeführt. Da die Anwendungsfälle nach Kategorie geordnet sind, kann die Filterung an die Anforderungen dieser Anwendungsfälle angepasst werden.

Bei der Filterung kann es sich unter anderem um eine zeitliche Filterung handeln. Diese kann durchgeführt werden, indem das bereitgestellte Zeitintervall der Daten beschränkt wird oder die zeitliche Auflösung reduziert wird. Es können auch bestimmte Tupel herausgefiltert werden bzw. die bereitgestellten Attribute beschränkt werden. Die Filterung kann auch durch eine Aggregation von Daten durchgeführt werden. Es können zum Beispiel neue Kenngrößen berechnet werden. Eine weitere Möglichkeit die Filterung durchzuführen ist, die Daten zu anonymisieren oder pseudonymisieren.

Durch diese Struktur wird Anforderung A4 erfüllt.

Anforderung A5 — Reduktion der Aussagekraft der Daten. Anforderung A5 fordert, dass die Aussagekraft der bereitgestellten Daten reduziert werden soll. Dabei legt diese Anforderung das Augenmerk auf die datenschutzkonforme Filterung der Daten vor der Bereitstellung.

In RETORT haben externe Anwendungen nur Zugriff auf ausgewählte Anwendungsfälle, nicht jedoch auf die gesammelten Daten im Rohformat. Vor der Bereitstellung der Daten in den Anwendungsfällen wird eine Datenreduktion durchgeführt. Diese Reduktion des Datenvolumens kann durch eine zeitliche Filterung geschehen. Dabei kann das bereitgestellte Zeitintervall der Daten beschränkt oder die zeitliche Auflösung reduziert werden. Auch das Herausfiltern bestimmter Tupel bzw. die Beschränkung der bereitgestellten Attribute kann das Datenvolumen reduzieren. Dadurch stehen den externen Anwendungen weniger Daten zur Verfügung. Aus dieser reduzierten Datenmenge sind weniger Informationen ableitbar als aus der gesamten Datenmenge, wodurch die Daten eine geringere Aussagekraft besitzen.

Eine weitere Komponente der Datenreduktion vor der Bereitstellung ist, dass die Daten vorverarbeitet werden. Es können unter anderem neue Kenngrößen berechnet werden, die eine geringere Aussagekraft besitzen. Diese Kenngrößen werden innerhalb des Wissensmodells definiert. Es kann sich dabei zum Beispiel um statistische Methoden, wie zum Beispiel den Mittelwert oder die Summe handeln. Aber auch andere Kenngrößen, die zum Beispiel das Verhältnis zweier Größen repräsentieren oder Aussagen, ob ein definierter Schwellenwert überschritten ist, können verwendet werden. Ein Beispiel ist, statt Gewicht und Körpergröße nur den Body Mass Index (BMI) bereitzustellen. Dabei handelt es sich um eine Kenngröße, die das Verhältnis zweier Größen, hier Gewicht und Körpergröße wiedergibt. Diese Kenngrößen reduzieren die Aussagekraft der Daten zusätzlich.

Die Aussagekraft der Daten kann auch reduziert werden, indem sie anonymisiert oder pseudonymisiert werden. Nach §3 Abs. 6 BDSG a.F. bedeutet anonymisieren das Verändern personenbezogener Daten derart, dass sie nicht mehr oder nur mit einem unverhältnismäßig großen Aufwand an Zeit, Kosten und Arbeitskraft einer bestimmten oder bestimmbaren natürlichen Person zugeordnet werden können. Unter Pseudonymisierung versteht man nach Art. 4 Nr. 5 DSGVO die Verarbeitung personenbezogener Daten in einer Weise, dass sie ohne zusätzliche Informationen nicht mehr einer spezifischen betroffenen Person zugeordnet werden können. Durch die erschwerte Zuordnung der Daten zu den jeweils betroffenen Personen wird die Aussagekraft der Daten reduziert.

Somit wird durch diese Datenreduktionen Anforderung A5 erfüllt.

Anforderung A6 — Kategorie der Daten und des Anwendungsfalles berücksichtigen. Anforderung A6 fordert, dass die Kategorie der zu reduzierenden Daten und des entsprechenden Anwendungsfalles bei der Datenreduktion berücksichtigt werden sollen.

In RETORT sind die Anwendungsfälle in mehreren Baumstrukturen angeordnet. Diese sind getrennt nach der Kategorie der Anwendungsfälle. Durch die Trennung der Anwendungsfälle nach Kategorie, kann bei der Datenreduktion berücksichtigt werden, für welche Kategorie die Datenreduktion durchgeführt wird. Im Wissensmodell ist festgelegt, wie die Daten verknüpft und angereichert werden. Bei der Erstellung des Wissensmodells kann auf die Kategorie der Daten eingegangen werden. Die im Wissensmodell festgehaltenen Anweisungen werden in der Anreicherungszone befolgt. Somit hat die bei der Erstellung des Wissensmodells berücksichtigte Kategorie der Daten einen Einfluss auf die Anreicherung und Datenreduktion. Dadurch wird Anforderung A6 erfüllt.

Kapitel 9

Zusammenfassung und Ausblick

Die Entwicklung von IoT-fähigen Geräten schreitet stetig voran. Dadurch entstehen zahlreiche neue technische Anwendungsmöglichkeiten und unterschiedliche IoT-Anwendungen. Das Internet of Things hält in vielen Bereichen des alltäglichen Lebens Einzug, im Privatbereich als Smart Home, in die Gesundheitsbranche oder in die Industrie 4.0. Mit der Verbreitung des IoT steigt auch die Anzahl an verschiedenen Sensoren, die regelmäßig Daten sammeln können. Dadurch wird kontinuierlich eine große heterogene Datenmenge generiert, die IoT-Anwendungen vor eine Herausforderung stellt. Zudem gibt es Fälle in denen nicht alle Daten allen Anwendungen bereitgestellt werden dürfen. Auch können durch die Anreicherung und Verknüpfung von Daten neue Informationen generiert werden. Bei der Datenbereitstellung muss die Privatsphäre der betroffenen Personen geschützt werden.

In dieser Arbeit wird daher RETORT entwickelt und implementiert. RETORT ist eine Datenbereitstellungsplattform für datenintensive IoT-Anwendungen, die Daten bedarfsgerecht bereitstellen kann. Bedarfsgerecht beinhaltet in diesem Kontext die benötigte Datenanreicherung, als auch die anschließende Datenreduktion um Big Data verarbeiten zu können und die Privatsphäre der betroffenen Person zu schützen. Die Daten werden hierzu in einem Data Lake gehalten, der aus mehreren Zonen besteht: Landezone, Rohdatenzone, Anreicherungszone und Zonen der Anwendungsfälle. Die Daten werden in der Rohdatenzone im Rohformat persistiert. In der Anreicherungszone werden sie für die Bereitstellung verarbeitet. In einem Wissensmodell wird beschrieben, wie die verfügbaren Daten in der Anreicherungszone angereichert und verknüpft werden können und welche Informationen und welches Wissen daraus abgeleitet werden kann. Bei der Verarbeitung der Daten wird auch eine Datenreduktion durchgeführt. So wird zum einen die Verarbeitung der Daten mit herkömmlichen Methoden ermöglicht. Zum anderen dient dies dem Schutz der Privatsphäre der betroffenen Person. Die verarbeiteten Daten werden externen Anwendungen in den Zonen der Anwendungsfälle zur Verfügung gestellt. Dabei kann es sich um Auszüge der Rohdaten, verarbeitete Daten oder abgeleitete Informationen handeln.

All diese Punkte werden in dieser Arbeit in einem Implementierungskonzept für RETORT zusammengefasst. Dieses stellt eine mögliche Umsetzungsstrategie dar. Um die Realisierung von RETORT zu validieren, wird das vorgestellte Implementierungskonzept prototypisch umgesetzt. Der Prototyp orientiert sich dabei an dem definierten Anwendungsszenario einer Fitnessakte aus dem Gesundheitsbereich. Er ist in der Lage Daten mit verschiedenen Schemata zu verwalten, diese Daten zu aggregieren und anzureichern und die Ergebnisse der Aggregation und Anreicherung bereitzustellen.

Des Weiteren wird RETORT anhand der Verarbeitung von Big Data und definierten Anforderungen evaluiert. Bei der Evaluation bezüglich der Verarbeitung von Big Data zeigt sich, dass es in RETORT Konzepte gibt, die mit den meisten Herausforderungen der Verarbeitung von Big Data umgehen können. Bei der Evaluation von RETORT anhand der zuvor definierten Anforderungen zeigt sich, dass RETORT alle Anforderungen erfüllt.

Ausblick

Im Folgenden werden verschiedene Aspekte erläutert, in denen RETORT in zukünftigen Arbeiten erweitert oder konkretisiert werden kann.

Bereinigungskonzepte für die Rohdaten. Durch die vielen unterschiedlichen Quellen der Daten, welche in RETORT gesammelt werden, und die stetige Verbreitung von IoT Geräten, kann es sein, dass unterschiedliche Sensoren die gleiche Eigenschaft messen. In zukünftigen Arbeiten können Konzepte entwickelt werden, wie mit dieser Überschneidung umgegangen werden kann. Dabei müssen auch Strategien entwickelt werden für den Fall, dass die Messungen nicht übereinstimmen.

Durch die Verbreitung von Low Cost oder Low Power Sensoren kann die Genauigkeit der Daten nicht mehr gewährleistet werden. Eine Möglichkeit dies auszugleichen ist, mehrere Sensoren zu verwenden und eine Datenfusion durchzuführen [AMK+17]. In zukünftigen Arbeiten kann dies genauer untersucht werden und ein Konzept zur Einbindung in RETORT entwickelt werden. Dabei kann auch berücksichtigt werden, wie mit Daten aus nicht vertrauenswürdigen Quellen umgegangen werden soll.

Auch andere Bereinigungskonzepte für die Rohdaten können untersucht werden, damit die gesammelte Datenmenge beherrschbar bleibt. Es kann zum Beispiel ein Bereinigungskonzept entworfen werden, welches Echtzeitdaten, die ihre Gültigkeit verloren haben und nicht mehr benötigt werden, löscht, um das Wachstum der gespeicherten Datenmenge einzudämmen. Dabei ist darauf zu achten, dass die Anforderungen an RETORT weiter erfüllt bleiben.

***Veracity* — Vertrauenswürdigkeit der Daten.** Wie in Abschnitt 8.1 erläutert, ist dieser Aspekt von Big Data und dessen Umsetzung in RETORT kein Schwerpunkt dieser Arbeit, da innerhalb des Anwendungsszenarios der Fitnessakte der Benutzer die Datenhoheit hat. Die Entwicklung eines konkreten Konzeptes wird zukünftigen Arbeiten überlassen. Es können Konzepte entwickelt werden, die unterschiedliche Aspekte behandeln, wie gewährleistet werden kann, dass die Daten vertrauenswürdig sind. Um zu entscheiden, ob die Quelle der Daten vertrauenswürdig ist, kann zusätzlich zu den Daten eine Referenz auf die Datenquelle gespeichert werden. Dadurch kann bei der Verarbeitung der Daten über deren Vertrauenswürdigkeit basierend auf der Vertrauenswürdigkeit der Quelle entschieden werden.

Ein weiterer Aspekt ist, sicherzustellen, dass die Daten nicht manipuliert wurden. Dies kann durch eine Authentifizierung der Sensoren geschehen. Dabei kann durch eine digitale Signatur überprüft werden, dass die Software und Hardware der Datenquelle nicht manipuliert wurden und die übermittelten Daten dadurch legitim sind [GOM18].

***Velocity* — Echtzeitdatenverarbeitung.** In Abschnitt 6.2 wird vorgeschlagen, die angereicherten Daten in einer relationalen Datenbank zu speichern, um nicht bei jedem Abrufen der Daten wiederholt die Anreicherung durchführen zu müssen. Wie in Kapitel 8 erläutert wird, ist so eine Echtzeitdatenverarbeitung erschwert, da neue Daten nicht automatisch in den Zonen der Anwendungsfälle sichtbar sind. In zukünftigen Arbeiten kann zwischen der Verwendung eines Caches und der Echtzeitdatenverarbeitung abgewogen werden. Es können auch Konzepte entwickelt werden, die beide Eigenschaften vereinen.

Um die Echtzeitdatenverarbeitung effizient durchzuführen, kann Stream Processing verwendet werden. Dabei werden die Daten direkt nach ihrem Empfang kontinuierlich verarbeitet und analysiert. Dies stellt einen gegensätzlichen Ansatz zum Batch Processing dar, bei dem die Daten zunächst gesammelt werden, in Datenbanken gespeichert werden und erst im Nachgang bei Bedarf verarbeitet und analysiert werden [LL19].

Einen Ansatz, diese beiden Arten der Verarbeitung zu vereinen, stellt die Lambda Architektur dar [MW15]. Diese kombiniert das Batch Processing mit einer zusätzlichen Echtzeitkomponente [WGFR16]. Dieses Konzept kann auf RETORT übertragen werden.

Konkretisierung verschiedener Aspekte des Konzepts von RETORT. Nicht alle Komponenten von RETORT sind ein Schwerpunkt dieser Arbeit und werden detailliert beschrieben.

Die Landezone wird nur grundlegend behandelt. Es wird empfohlen den Puffer nach dem Prinzip First In – First Out (FIFO) umzusetzen. In zukünftigen Arbeiten kann dieser Puffer konkretisiert werden. Auch kann die Zerlegung der Daten in einzelne Datenkomponenten untersucht werden. Ein Aspekt davon ist unter anderem, wie die Datenkomponenten nach der Zerlegung den einzelnen Messungen noch zuordenbar sind.

Auch die Anreicherungszone kann konkretisiert werden. Es können verschiedene Konzepte der Datenanreicherung und -verknüpfung untersucht werden und in RETORT integriert werden.

Des Weiteren wird die dritte Komponente von RETORT, die Zugriffskontrolle, innerhalb dieser Arbeit nicht konkret behandelt und kann deshalb Thema einer zukünftigen Arbeit sein. Dabei können verschiedenen Ansätze evaluiert und ihre Verwendung in RETORT untersucht werden. In dieser Arbeit wird die attributbasierte Zugriffskontrolle aufgrund der Möglichkeiten der feinen Berechtigungsvergabe vorgeschlagen. Dieser Vorschlag kann basierend auf den zukünftigen Untersuchungen bewertet werden.

Literaturverzeichnis

[ABF+18] S. Alpers, S. Betz, A. Fritsch, A. Oberweis, G. Schiefer, M. Wagner. „Citizen Empowerment by a Technical Approach for Privacy Enforcement“. In: *Proceedings of the 8th International Conference on Cloud Computing and Services Science*. INSTICC. SCITEPRESS - Science und Technology Publications, 2018, S. 589–595. ISBN: 978-989-758-295-0. DOI: 10.5220/0006789805890595 (zitiert auf S. 41).

[ABS14] S. Abiteboul, P. Buneman, D. Suciu. *Data on the Web: From Relations to Semistructured Data and XML*. San Francisco: Morgan Kaufmann, 19. Feb. 2014. ISBN: 9781558606227 (zitiert auf S. 28, 77).

[Ack89] R. L. Ackoff. „From Data to Wisdom“. In: *Journal of Applied Systems Analysis* 16 (1989), S. 3–9 (zitiert auf S. 45).

[AFR+20] S. Abbasian Dehkordi, K. Farajzadeh, J. Rezazadeh, R. Farahbakhsh, K. Sandrasegaran, M. A. Dehkordi. „A survey on data aggregation techniques in IoT sensor networks“. In: *Wireless Networks* 26.2 (Feb. 2020), S. 1243–1263. ISSN: 1572-8196. DOI: 10.1007/s11276-019-02142-z (zitiert auf S. 57).

[AG08] R. Angles, C. Gutierrez. „Survey of Graph Database Models“. In: *ACM Computing Surveys* 40.1 (Feb. 2008), S. 1–39. ISSN: 0360-0300. DOI: 10.1145/1322432.1322433 (zitiert auf S. 33).

[AGD16] M. Abramovici, J. C. Göbel, H. B. Dang. „Semantic data management for the development and continuous reconfiguration of smart products and systems“. In: *CIRP Annals* 65.1 (2016), S. 185–188. DOI: 10.1016/j.cirp.2016.04.051 (zitiert auf S. 40).

[AMK+17] F. Alam, R. Mehmood, I. Katib, N. N. Albogami, A. Albeshri. „Data Fusion and IoT for Smart Ubiquitous Environments: A Survey“. In: *IEEE Access* 5 (2017), S. 9533–9554. DOI: 10.1109/access.2017.2697839 (zitiert auf S. 17, 57, 82).

[AOP+17] S. Alpers, A. Oberweis, M. Pieper, S. Betz, A. Fritsch, G. Schiefer, M. Wagner. „PRIVACY-AVARE: An approach to manage and distribute privacy settings“. In: *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*. IEEE, Dez. 2017, S. 1460–1468. DOI: 10.1109/compcomm.2017.8322784 (zitiert auf S. 41).

[Bau05] H. Baumann. *Photoshop-Basiswissen : Edition DOCMA*. München: Addison-Wesley, 2005. ISBN: 9783827323149 (zitiert auf S. 49).

[BB19] M. Banane, A. Belangour. „A Survey on RDF Data Store Based on NoSQL Systems for the Semantic Web Applications“. In: *Advanced Intelligent Systems for Sustainable Development (AI2SD'2018)*. Hrsg. von M. Ezziyyani. Cham: Springer International Publishing, 2019, S. 444–451. ISBN: 978-3-030-11928-7. DOI: 10.1007/978-3-030-11928-7_40 (zitiert auf S. 55).

[BHL+12] J. Busse, B. Humm, C. Lübbert, F. Moelter, A. Reibold, M. Rewald, V. Schlüter, B. Seiler, E. Tegtmeier, T. Zeh. „Was bedeutet eigentlich Ontologie? Ein Begriff aus der Philosophie im Licht verschiedener Disziplinen“. In: *Informatik-Spektrum* 37.4 (12. Juni 2012), S. 286–297. ISSN: 1432-122X. DOI: 10.1007/s00287-012-0619-2 (zitiert auf S. 35).

[BHRZ10] W. Bartussek, B. Humm, A. Reibold, T. Zeh. „Zur Definition von „Ontologie in den Informationswissenschaften““. In: (3. Nov. 2010). URL: http://www.ontologenkreis.org/docs/Definition-Ontologie_Sammlung.pdf (besucht am 13. 06. 2020) (zitiert auf S. 34, 35).

[BJM+15] O. Beijbom, N. Joshi, D. Morris, S. Saponas, S. Khullar. „Menu-Match: Restaurant-Specific Food Logging from Images“. In: *2015 IEEE Winter Conference on Applications of Computer Vision*. IEEE, Jan. 2015, S. 844–851. DOI: 10.1109/wacv.2015.117 (zitiert auf S. 22).

[BM14] D. Brickley, L. Miller. *FOAF Vocabulary Specification 0.99*. 14. Jan. 2014. URL: http://xmlns.com/foaf/spec/20140114.html (besucht am 18. 09. 2020) (zitiert auf S. 54).

[Bra14] B. Braun. „Daten als „Öl des 21. Jahrhunderts““. In: *Controlling & Management Review* 58.4 (Aug. 2014), S. 88–89. ISSN: 2195-8270. DOI: 10.1365/s12176-014-0989-1 (zitiert auf S. 77).

[Bru12] F. Bruckner. *Wide Column Stores*. Techn. Ber. Hochschule Mannheim Fakultät für Informatik, 15. Juni 2012. 10 S. (zitiert auf S. 33).

[BSU18] Z. Baloch, F. K. Shaikh, M. A. Unar. „A context-aware data fusion approach for health-IoT“. In: *International Journal of Information Technology* 10.3 (Feb. 2018), S. 241–245. DOI: 10.1007/s41870-018-0116-1 (zitiert auf S. 17).

[BZ18] A. Boes, A. Ziegler. *Der Aufstieg des Internet of Things. Disruptiver Wandel für die deutsche Wirtschaft?* Forschungsber. München: Institut für Sozialwissenschaftliche Forschung e.V, 2018. 32 S. (zitiert auf S. 17).

[Cat11] R. Cattell. „Scalable SQL and NoSQL data stores“. In: *ACM SIGMOD Record* 39.4 (Mai 2011), S. 12–27. ISSN: 0163-5808. DOI: 10.1145/1978915.1978919 (zitiert auf S. 30–32).

[CWL14] R. Cyganiak, D. Wood, M. Lanthaler. *RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation 25 February 2014*. W3C. 25. Feb. 2014. URL: http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/ (besucht am 11. 06. 2020) (zitiert auf S. 36, 37, 53).

[CXJV17] H. Cai, B. Xu, L. Jiang, A. V. Vasilakos. „IoT-Based Big Data Storage Systems in Cloud Computing: Perspectives and Challenges“. In: *IEEE Internet of Things Journal* 4.1 (Feb. 2017), S. 75–87. ISSN: 2327-4662. DOI: 10.1109/JIOT.2016.2619369 (zitiert auf S. 39).

[Ded18] Dedicated Developers. *How Mobile Apps are Leveraging the Internet of Things (IoT)*. 17. Mai 2018. URL: https://www.businessofapps.com/news/how-mobile-apps-are-leveraging-the-internet-of-things-iot/ (besucht am 21. 09. 2020) (zitiert auf S. 17).

[DRD+20] B. Diène, J. J. Rodrigues, O. Diallo, E. H. M. Ndoye, V. V. Korotaev. „Data management techniques for Internet of Things“. In: *Mechanical Systems and Signal Processing* 138 (Apr. 2020), S. 106564. DOI: 10.1016/j.ymssp.2019.106564 (zitiert auf S. 39).

[DYB20] A. J. A. Donkers, D. Yang, N. H. G. Baken. „Linked Data for Smart Homes: Comparing RDF and Labeled Property Graphs“. In: *LDAC 2020 Linked Data in Architecture and Construction*. Hrsg. von M. Poveda-Villalón, A. Roxin, K. McGlinn, P. Pauwels. CEUR Workshop Proceedings. CEUR-WS.org, 2020, S. 23–36 (zitiert auf S. 36, 53, 54).

[Eck11] S. Eckstein. „Informationsintegration“. In: *Informationsmanagement in der Systembiologie: Datenbanken, Integration, Modellierung*. Berlin, Heidelberg: Springer Berlin Heidelberg, 31. März 2011, S. 95–151. ISBN: 978-3-642-18234-1. DOI: 10.1007/978-3-642-18234-1_4 (zitiert auf S. 17, 57).

[EDU20] EDUCBA. *Introduction to Relational Database Advantages*. 2020. URL: https://www.educba.com/relational-database-advantages/ (besucht am 17. 10. 2020) (zitiert auf S. 57).

[ER16] Europäisches Parlament, Rat der Europäischen Union. *Verordnung (EU) 2016/679 des Europäischen Parlaments und des Rates*. 27. Apr. 2016. URL: http://data.europa.eu/eli/reg/2016/679/oj (besucht am 21. 09. 2020) (zitiert auf S. 18).

[Fan15] H. Fang. „Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem“. In: *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*. IEEE, Juni 2015, S. 820–824. DOI: 10.1109/cyber.2015.7288049 (zitiert auf S. 76).

[FLB+06] T. Furche, B. Linse, F. Bry, D. Plexousakis, G. Gottlob. „RDF Querying: Language Constructs and Evaluation Methods Compared“. In: *Reasoning Web*. Springer Berlin Heidelberg, 2006, S. 1–52. ISBN: 978-3-540-38412-0. DOI: 10.1007/11837787_1 (zitiert auf S. 37).

[Gar] Gartner Inc. *Big Data*. URL: https://www.gartner.com/en/information-technology/glossary/big-data (besucht am 11. 07. 2020) (zitiert auf S. 27).

[GGH+20a] C. Giebler, C. Gröger, E. Hoos, R. Eichler, H. Schwarz, B. Mitschang. „Data Lakes auf den Grund gegangen“. In: *Datenbank-Spektrum* 20.1 (Jan. 2020), S. 57–69. DOI: 10.1007/s13222-020-00332-0 (zitiert auf S. 28).

[GGH+20b] C. Giebler, C. Gröger, E. Hoos, H. Schwarz, B. Mitschang. „A Zone Reference Model for Enterprise-Grade Data Lake Management“. In: *2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)*. IEEE, Okt. 2020, S. 57–66. DOI: 10.1109/edoc49727.2020.00017 (zitiert auf S. 29, 40).

[GOM18] C. Gritti, M. Onen, R. Molva. „CHARIOT: Cloud-Assisted Access Control for the Internet of Things“. In: *2018 16th Annual Conference on Privacy, Security and Trust (PST)*. IEEE, Aug. 2018, S. 1–6. DOI: 10.1109/pst.2018.8514217 (zitiert auf S. 77, 82).

[Gor19] A. Gorelik. *The Enterprise Big Data Lake*. O'Reilly UK Ltd., 1. Apr. 2019. ISBN: 1491931558 (zitiert auf S. 75).

[Grä04] P. G. von Grätz. *Vernetzte Medizin: Patienten-Empowerment und Netzinfrastrukuren in der Medizin des 21. Jahrhunderts*. Hannover: Heise, 2004. ISBN: 9783936931198 (zitiert auf S. 21).

[GRR14] V. N. Gudivada, D. Rao, V. V. Raghavan. „NoSQL Systems for Big Data Management“. In: *2014 IEEE World Congress on Services*. IEEE, Juni 2014, S. 190–197. DOI: 10.1109/services.2014.42 (zitiert auf S. 31, 32).

[Gru93] T. R. Gruber. „A translation approach to portable ontology specifications“. In: *Knowledge Acquisition* 5.2 (Juni 1993), S. 199–220. DOI: 10.1006/knac.1993.1008 (zitiert auf S. 34).

[GZS18] A. Gyrard, A. Zimmermann, A. Sheth. „Building IoT-Based Applications for Smart Cities: How Can Ontology Catalogs Help?“ In: *IEEE Internet of Things Journal* 5.5 (Okt. 2018), S. 3978–3990. ISSN: 2327-4662. DOI: 10.1109/jiot.2018.2854278 (zitiert auf S. 17).

[Har03] R. Harper. *Inside the Smart Home*. Berlin, Heidelberg: Springer-Verlag, 15. Aug. 2003. 280 S. ISBN: 1852336889 (zitiert auf S. 17).

[HBEV04] P. Haase, J. Broekstra, A. Eberhart, R. Volz. „A Comparison of RDF Query Languages“. In: *The Semantic Web – ISWC 2004*. Springer Berlin Heidelberg, 2004, S. 502–517. ISBN: 978-3-540-30475-3. DOI: 10.1007/978-3-540-30475-3_35 (zitiert auf S. 37, 53).

[HCH17] H.-H. Hsu, C.-Y. Chang, C.-H. Hsu. *Big Data Analytics for Sensor-Network Collected Intelligence*. 1. Aufl. USA: Academic Press, Inc., 2017. ISBN: 0128093935 (zitiert auf S. 28).

[HELD11] J. Han, H. E, G. Le, J. Du. „Survey on NoSQL database“. In: *2011 6th International Conference on Pervasive Computing and Applications*. IEEE, Okt. 2011, S. 363–366. DOI: 10.1109/icpca.2011.6106531 (zitiert auf S. 75).

[Hes02] W. Hesse. „Ontologie(n)“. In: *Informatik-Spektrum* 25.6 (Dez. 2002), S. 477–480. ISSN: 1432-122X. DOI: 10.1007/s002870200265 (zitiert auf S. 34).

[Hit08] P. Hitzler. *Semantic Web. Grundlagen*. Berlin: Springer Berlin Heidelberg, 2008. ISBN: 9783540339946. DOI: 10.1007/978-3-540-33994-6 (zitiert auf S. 36, 54, 68).

[HKF15] V. C. Hu, D. R. Kuhn, D. F. Ferraiolo. „Attribute-Based Access Control“. In: *Computer* 48.2 (Feb. 2015), S. 85–88. ISSN: 1558-0814. DOI: 10.1109/mc.2015.33 (zitiert auf S. 51).

[HYM14] S. Hiremath, G. Yang, K. Mankodiya. „Wearable Internet of Things: Concept, architectural components and promises for person-centered healthcare“. In: *2014 4th International Conference on Wireless Mobile Communication and Healthcare - Transforming Healthcare Through Innovations in Mobile and Wireless Technologies (MOBIHEALTH)*. Nov. 2014, S. 304–307. DOI: 10.1109/mobihealth.2014.7015971 (zitiert auf S. 17, 22).

[IA16] R. Isele, N. Arndt. „Mit semantischer Datenverwaltung Big Data in den Griff bekommen“. In: *Wirtschaftsinformatik & Management* 8.4 (Aug. 2016), S. 56–63. ISSN: 1867-5913. DOI: 10.1007/s35764-016-0065-z (zitiert auf S. 40).

[IKK+15] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, K.-S. Kwak. „The Internet of Things for Health Care: A Comprehensive Survey“. In: *IEEE Access* 3 (2015), S. 678–708. ISSN: 2169-3536. DOI: 10.1109/access.2015.2437951 (zitiert auf S. 17).

[it-] it-novum GmbH. *Data Lake*. URL: https://it-novum.com/big-data-analytics/data-lake/ (besucht am 25. 09. 2020) (zitiert auf S. 29, 31).

[JLY04] L. Jiang, D.-Y. Liu, B. Yang. „Smart home research“. In: *Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826)*. Bd. 2. IEEE, Aug. 2004, S. 659–663. DOI: 10.1109/icmlc.2004.1382266 (zitiert auf S. 17).

[Jur] JuraforumWiki-Redaktion. *Schutz der Privatsphäre - Definition, Rechte & im Internet*. URL: https://www.juraforum.de/lexikon/schutz-der-privatsphaere (besucht am 26. 09. 2020) (zitiert auf S. 18).

[JXC+14] L. Jiang, L. D. Xu, H. Cai, Z. Jiang, F. Bu, B. Xu. „An IoT-Oriented Data Storage Framework in Cloud Computing Platform“. In: *IEEE Transactions on Industrial Informatics* 10.2 (Mai 2014), S. 1443–1451. ISSN: 1941-0050. DOI: 10.1109/tii.2014.2306384 (zitiert auf S. 39).

[KKSF10] G. Kortuem, F. Kawsar, V. Sundramoorthy, D. Fitton. „Smart objects as building blocks for the Internet of things“. In: *IEEE Internet Computing* 14.1 (Jan. 2010), S. 44–51. ISSN: 1941-0131. DOI: 10.1109/mic.2009.143 (zitiert auf S. 17).

[KKTM18] H. Kondylakis, L. Koumakis, M. Tsiknakis, K. Marias. „Implementing a data management infrastructure for big healthcare data“. In: *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*. IEEE, März 2018, S. 361–364. DOI: 10.1109/bhi.2018.8333443 (zitiert auf S. 40).

[Kle16] S. Kleuker. „NoSQL mit MongoDB und Java“. In: *Grundkurs Datenbankentwicklung: Von der Anforderungsanalyse zur komplexen Datenbankanfrage*. Wiesbaden: Springer Fachmedien Wiesbaden, 2016, S. 299–326. ISBN: 978-3-658-12338-3. DOI: 10.1007/978-3-658-12338-3_15 (zitiert auf S. 65).

[KMJ+05] Y.-G. Kim, C.-J. Mon, D. Jeong, J.-O. Lee, C.-Y. Song, D.-K. Baik. „Context-Aware Access Control Mechanism for Ubiquitous Applications“. In: *Advances in Web Intelligence*. Hrsg. von P. S. Szczepaniak, J. Kacprzyk, A. Niewiadomski. Berlin, Heidelberg: Springer Berlin Heidelberg, 24. Mai 2005, S. 236–242. ISBN: 978-3-540-31900-9 (zitiert auf S. 41).

[KNL+18] F. Karim, O. A. Naameh, I. Lytra, C. Mader, M.-E. Vidal, S. Auer. „Semantic Enrichment of IoT Stream Data On-demand“. In: *2018 IEEE 12th International Conference on Semantic Computing (ICSC)*. IEEE, Jan. 2018, S. 33–40. DOI: 10.1109/icsc.2018.00014 (zitiert auf S. 17).

[KP11] H. Kondylakis, D. Plexousakis. „Exelixis: Evolving Ontology-Based Data Integration System“. In: *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*. SIGMOD '11. Athens, Greece: Association for Computing Machinery, 2011, S. 1283–1286. ISBN: 9781450306614. DOI: 10.1145/1989323.1989477 (zitiert auf S. 41).

[Krü19] L. Krüger. *Big Data im Gesundheitswesen Lehrmaterialien für den Schulunterricht : Jahrgangsstufen 10 bis 13 : Tutzinger Diskurs*. Tutzing: Akademie für Politische Bildung, 2019. ISBN: 9783981411188 (zitiert auf S. 18).

[KSS13] R. Kuhlen, W. Semar, D. Strauch, Hrsg. *Grundlagen der praktischen Information und Dokumentation*. DE GRUYTER SAUR, Jan. 2013. DOI: 10.1515/9783110258264 (zitiert auf S. 35, 36).

[Lan01] D. Laney. „3D Data Management: Controlling Data Volume, Velocity, and Variety“. In: *META group research note* 6 (Feb. 2001), S. 70 (zitiert auf S. 28).

[LL17] S. Luber, N. Litzel. *Was sind unstrukturierte Daten?* 28. Nov. 2017. URL: https://www.bigdata-insider.de/was-sind-unstrukturierte-daten-a-666378/ (besucht am 27. 09. 2020) (zitiert auf S. 28).

[LL19] S. Luber, N. Litzel. *Was ist Stream Processing?* 11. Sep. 2019. URL: https://www.bigdata-insider.de/was-ist-stream-processing-a-863076/ (besucht am 21. 10. 2020) (zitiert auf S. 83).

[Mad15] M. Madsen. „How to Build an Enterprise Data Lake: Important Considerations before Jumping In“. In: *Third Nature Inc* (2015), S. 13–17 (zitiert auf S. 29, 40).

[Mat17] C. Mathis. „Data Lakes“. In: *Datenbank-Spektrum* 17.3 (1. Nov. 2017), S. 289–293. ISSN: 1610-1995. DOI: 10.1007/s13222-017-0272-7 (zitiert auf S. 28, 39).

[McK18] W. McKinney. *Datenanalyse mit Python*. 2. Aufl. Archivierung/Langzeitarchivierung gewährleistet. Heidelberg: O'Reilly, 1. Nov. 2018. 542 S. ISBN: 978-3-96009-080-9 (zitiert auf S. 62).

[Mey18] S. Meyer. *Smart Objects Intelligente Objekte für IoT und IIoT-Anwendungen*. Techn. Ber. Fraunhofer-Institut für Integrierte Schaltungen IIS, Apr. 2018 (zitiert auf S. 17).

[MH13] A. B. M. Moniruzzaman, S. A. Hossain. „NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison“. In: *CoRR* abs/1307.0191 (30. Juni 2013). arXiv: 1307.0191 [cs.DB] (zitiert auf S. 29, 30).

[MK16] A. Meier, M. Kaufmann. *SQL- & NoSQL-Datenbanken*. Berlin, Heidelberg: Springer Vieweg Berlin Heidelberg, 2016. ISBN: 9783662476642. DOI: 10.1007/978-3-662-47664-2 (zitiert auf S. 27, 30–32, 34, 54, 57).

[MT16] N. Miloslavskaya, A. Tolstoy. „Big Data, Fast Data and Data Lake Concepts“. In: *Procedia Computer Science* 88 (2016), S. 300–305. DOI: 10.1016/j.procs.2016.07.439 (zitiert auf S. 75).

[MW15] N. Marz, J. Warren. *Big Data: Principles and Best Practices of Scalable Realtime Data Systems*. 1. Aufl. USA: Manning Publications Co., 7. Mai 2015. 328 S. ISBN: 1617290343 (zitiert auf S. 83).

[Ngu18] D. H. Nguyen. *Abstrakt zum Vortrag im Oberseminar Graphdatenbanken und Graph-Frameworks*. 7. Juni 2018 (zitiert auf S. 37).

[OBLB17] A. Oussous, F. Z. Benjelloun, A. A. Lahcen, S. Belfkih. „Comparison and Classification of NoSQL Databases for Big Data“. In: *International Journal of Big Data Intelligence* 4.3 (2017), S. 171. ISSN: 2053-1389. DOI: 10.1504/ijbdi.2017.085537 (zitiert auf S. 31, 56, 77).

[PZLN18] Z. Pan, T. Zhu, H. Liu, H. Ning. „A survey of RDF management technologies and benchmark datasets“. In: *Journal of Ambient Intelligence and Humanized Computing* 9.5 (Okt. 2018), S. 1693–1704. ISSN: 1868-5145. DOI: 10.1007/s12652-018-0876-2 (zitiert auf S. 55).

[RE10] A. Rahien, O. Eini. *RavenDB Mythology Documentation*. 29. Nov. 2010 (zitiert auf S. 56).

[Row07] J. Rowley. „The wisdom hierarchy: representations of the DIKW hierarchy“. In: *Journal of Information Science* 33.2 (Feb. 2007), S. 163–180. DOI: 10.1177/0165551506070706 (zitiert auf S. 45).

[Rus01] J. Rusher. *Triple Store*. Radar Networks. 2001. URL: https://www.w3.org/2001/sw/Europe/events/20031113-storage/positions/rusher.html (besucht am 21. 08. 2020) (zitiert auf S. 37).

[Rus17] P. Russom. *Data Lakes: Purposes, Practices, Patterns, and Platforms*. Techn. Ber. 2017 (zitiert auf S. 28).

[RV06] R. Rajagopalan, P. Varshney. „Data-aggregation techniques in sensor networks: a survey“. In: *IEEE Communications Surveys & Tutorials* 8.4 (2006), S. 48–63. DOI: 10.1109/comst.2006.283821 (zitiert auf S. 17).

[Sch07] M. Schubert. „Die Integrität einer Datenbank und Schlüssel aller Art“. In: *Datenbanken: Theorie, Entwurf und Programmierung relationaler Datenbanken*. Wiesbaden: Vieweg+Teubner Verlag, 8. Okt. 2007, S. 165–178. ISBN: 978-3-8351-9108-2. DOI: 10.1007/978-3-8351-9108-2_8 (zitiert auf S. 57).

[SDM+18] C. Stach, F. Durr, K. Mindermann, S. M. Palanisamy, S. Wagner. „How a Pattern-based Privacy System Contributes to Improve Context Recognition“. In: *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, März 2018, S. 355–360. DOI: 10.1109/percomw.2018.8480227 (zitiert auf S. 42).

[SGG] C. Stach, C. Giebler, C. Gritti. „How to Enable Privacy-aware Data Annotations“ (zitiert auf S. 18, 42–46, 50, 51).

[SGPM20] C. Stach, C. Gritti, D. Przytarski, B. Mitschang. „Trustworthy, Secure, and Privacy-aware Food Monitoring Enabled by Blockchains and the IoT“. In: *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, März 2020, S. 1–4. DOI: 10.1109/percomworkshops48775.2020.9156150 (zitiert auf S. 77).

[Sha18] B. Sharma. *Architecting data lakes: data management architectures for advanced business use cases*. OCLC: 1082143725. 2018 (zitiert auf S. 28, 29, 39, 47).

[Sit02] D. F. Sittig. „Personal health records on the internet: a snapshot of the pioneers at the end of the 20th Century“. In: *International Journal of Medical Informatics* 65.1 (Apr. 2002), S. 1–6. ISSN: 1386-5056. DOI: 10.1016/s1386-5056(01)00215-5 (zitiert auf S. 21).

[SM19] C. Stach, B. Mitschang. „Elicitation of Privacy Requirements for the Internet of Things Using ACCESSORS“. In: *Information Systems Security and Privacy*. Hrsg. von P. Mori, S. Furnell, O. Camp. Cham: Springer International Publishing, 4. Juli 2019, S. 40–65. ISBN: 978-3-030-25109-3 (zitiert auf S. 41).

[sol20a] solid IT gmbh. *DB-Engines Ranking von Document Stores*. Aug. 2020. URL: https://db-engines.com/de/ranking/document%20store (besucht am 16. 08. 2020) (zitiert auf S. 65).

[sol20b] solid IT gmbh. *DB-Engines Ranking von RDF Stores*. Sep. 2020. URL: https://db-engines.com/de/ranking/rdf%20store (besucht am 19. 09. 2020) (zitiert auf S. 55).

[sol20c] solid IT gmbh. *DB-Engines Ranking von Relational DBMS*. Sep. 2020. URL: https://db-engines.com/de/ranking/relational%20dbms (besucht am 23. 09. 2020) (zitiert auf S. 72).

[SP10] E. Sahafizadeh, S. Parsa. „Survey on access control models“. In: *2010 2nd International Conference on Future Computer and Communication*. Bd. 1. IEEE, Mai 2010, S. V1-1-V1–3. DOI: 10.1109/icfcc.2010.5497850 (zitiert auf S. 22).

[SR14] G. Schreiber, Y. Raimond. *RDF 1.1 Primer. W3C Working Group Note 24 June 2014*. W3C. 24. Juni 2014. URL: http://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/ (besucht am 10. 07. 2020) (zitiert auf S. 36).

[SS13] S. Sagiroglu, D. Sinanc. „Big data: A review“. In: *2013 International Conference on Collaboration Technologies and Systems (CTS)*. IEEE, Mai 2013, S. 42–47. DOI: 10.1109/cts.2013.6567202 (zitiert auf S. 27).

[SSGM19] C. Stach, F. Steimle, C. Gritti, B. Mitschang. „PSSST! The Privacy System for Smart Service Platforms: An Enabler for Confidable Smart Environments“. In: *Proceedings of the 4th International Conference on Internet of Things, Big Data and Security - Volume 1: IoTBDS,* INSTICC. SCITEPRESS - Science und Technology Publications, 2019, S. 57–68. ISBN: 978-989-758-369-8. DOI: 10.5220/0007672900570068 (zitiert auf S. 41).

[Sta19a] C. Stach. „Datenschutzkonzepte für Zeitreihendaten“. In: *Datenschutz und Datensicherheit - DuD* 43.12 (Dez. 2019), S. 753–759. ISSN: 1862-2607. DOI: 10.1007/s11623-019-1201-8 (zitiert auf S. 42).

[Sta19b] C. Stach. „VAULT: A Privacy Approach Towards High-Utility Time Series Data“. In: Okt. 2019, S. 41–46. ISBN: 978-1-61208-746-7 (zitiert auf S. 42).

[Ste18] A. Steinbach. *Daten sind das Öl des 21. Jahrhunderts*. Interview. Apr. 2018. URL: https://www.springerprofessional.de/transformation/industrie-4-0/daten-sind-das-oel-des-21--jahrhunderts/15708478 (besucht am 20. 10. 2020) (zitiert auf S. 77).

[SW16] I. Szilagyi, P. Wira. „Ontologies and Semantic Web for the Internet of Things - a survey“. In: *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*. IEEE, Okt. 2016, S. 6949–6954. DOI: 10.1109/iecon.2016.7793744 (zitiert auf S. 40).

[SWH+15] M. Steinebach, C. Winter, O. Halvani, M. Schäfer, Y. Yannikos. *Chancen durch Big Data und die Frage des Privatsphärenschutzes. Begleitpapier Bürgerdialog. Big Data und Privatheit*. Techn. Ber. Stuttgart, Juni 2015. 60 S. (zitiert auf S. 28).

[TA09] R. Trill, S. Arendt. „Die webbasierte Fitnessakte als Element der elektronischen Gesundheitsakte (eGA) in Deutschland“. In: *Telemedizinführer Deutschland* (2009). Hrsg. von Jäckel, S. 174–176 (zitiert auf S. 21, 22).

[The13] The W3C SPARQL Working Group. *SPARQL 1.1 Overview. W3C Recommendation 21 March 2013*. W3C. 21. März 2013. URL: http://www.w3.org/TR/2013/REC-sparql11-overview-20130321/ (besucht am 11. 06. 2020) (zitiert auf S. 37, 57, 68).

[Tre14] T. Trelle. *MongoDB*. 1. Aufl. Heidelberg: Dpunkt.Verlag GmbH, 1. Juni 2014. 290 S. ISBN: 3864901537 (zitiert auf S. 65).

[UM12] M. Unterstein, G. Matthiessen. *Relationale Datenbanken und SQL in Theorie und Praxis*. 5. Aufl. eXamen.press. Online-Ressource (XII, 307 Seiten 124 Abb, digital). Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. ISBN: 978-3-642-28986-6. DOI: 10.1007/978-3-642-28986-6 (zitiert auf S. 30).

[V14] M. V. „Comparative Study of NoSQL Document, Column Store Databases and Evaluation of Cassandra“. In: *International Journal of Database Management Systems (IJDMS)* 6.4 (Aug. 2014), S. 11–26. ISSN: 09755985. DOI: 10.5121/ijdms.2014.6402 (zitiert auf S. 33).

[W3C] W3C Semantic Sensor Network Incubator Group. *Semantic Sensor Network Ontology*. URL: https://www.w3.org/2005/Incubator/ssn/ssnx/ssn (besucht am 12. 10. 2020) (zitiert auf S. 40).

[WGFR16] W. Wingerath, F. Gessert, S. Friedrich, N. Ritter. „Real-time stream processing for Big Data“. In: *it - Information Technology* 58.4 (28. Aug. 2016), S. 186–194. DOI: 10.1515/itit-2016-0002 (zitiert auf S. 83).

[ZRL+17] Y. Zhang, J. Ren, J. Liu, C. Xu, H. Guo, Y. Liu. „A Survey on Emerging Computing Paradigms for Big Data“. In: *Chinese Journal of Electronics* 26.1 (Jan. 2017), S. 1–12. ISSN: 2075-5597. DOI: 10.1049/cje.2016.11.016 (zitiert auf S. 18).

Alle URLs wurden zuletzt am 05. 11. 2020 geprüft.

Erklärung

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

Ort, Datum, Unterschrift