Universität Stuttgart

Institut für Parallele und Verteilte Systeme

Universitätsstraße 38
70569 Stuttgart

Bachelorarbeit

Entwicklung von Metadatentemplates für unternehmensinterne Datenmarktplätze

Ganna Berezhna

Studiengang:	Medieninformatik
1. Prüfer:	Prof. Dr.-Ing. Bernhard Mitschang
2. Prüfer:	
Betreuer:	M.Sc. Rebecca Kay Eichler
begonnen am:	13.12.2021
beendet am:	13.07.2022

Kurzfassung

Die Menge der Daten in Unternehmen nimmt jedes Jahr zu. Diese Daten werden in verschiedenen Datenbanken des Unternehmens gespeichert und sind unterschiedlich strukturiert, wodurch es für die Mitarbeiter zunehmend herausfordernder wird, die Daten zu finden, zu verstehen, darauf zuzugreifen und diese für unterschiedliche Anwendungsfälle zu nutzen. Die Lösung dieses Problems liegt im Einsatz eines internen Datenmarktplatzes, d.h. einer Plattform, die es ermöglicht, die Informationen über im Unternehmen vorhandene Daten zu erfassen und die Daten zwischen den Mitarbeitern des Unternehmens auszutauschen. Damit die Nutzer des Datenmarktplatzes in der Lage sind, Daten zu suchen, auszuwählen und ihre Bedeutung zu verstehen, müssen die Daten mit Metadaten versehen werden. Metadaten oder „Daten über die Daten“ liefern z.B. Informationen über Datentyp, Owner, Struktur, Qualität, Nutzungsbedingungen und Anwendungsbereich. Wie aus diesen Beispielen ersichtlich wird, kann im Unternehmen eine Vielzahl von Metadaten erfasst und gespeichert werden. Dabei unterscheiden sich die Metadaten auch basierend auf den Daten, beispielsweise werden unterschiedliche Metadaten für verschiedene Datentypen wie Bilder, Text oder Videos benötigt. Um einen Überblick darüber zu gewinnen, welche Metadaten im Rahmen eines Datenmarktplatzes relevant sind, um die Daten auffindbar, verständlich und zugreifbar zu machen, wird eine strukturierte Übersicht benötigt, welche Metadaten für die unterschiedlichen Datentypen von Bedeutung sind, sowie ein Konzept, das die dynamische Darstellung der verschiedenen Metadaten auf dem Datenmarktplatz unterstützt.

Im Rahmen dieser Arbeit werden Metadatentemplates für den internen Datenmarktplatz entwickelt und sieben Gruppen von solchen Templates festgelegt. Darüber hinaus werden ein Konzept zur Darstellung von Templates und Template-Gruppen im Frontend des Datenmarktplatzes erarbeitet sowie der Einsatz der Templates prototypisch demonstriert. Zudem wird ein Metamodell für die Metadaten des Datenmarktplatzes konzipiert und es werden Metadatenmanagement-Tools vorgestellt, mit denen diese Metadaten extrahiert werden können. Metadatentemplates legen nicht nur fest, welche Metadaten auf Datenmarktplatz angezeigt werden können, sondern ermöglichen auch eine dynamische Darstellung verschiedener Metadaten zu unterschiedlichen Daten. Basierend auf den angezeigten Metadaten können die Datenmarktplatznutzer die Daten verstehen, verschiedene Datensätze miteinander vergleichen und diejenigen auswählen, die für ihren Anwendungsfall am besten geeignet sind.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AWS Amazon Web Services
BVA Bundesverwaltungsamt
CRM Customer Relationship Management
CSV Comma Separated Values
DBMS Datenbankmanagementsystem
DCAT Data Catalog Vocabulary
DERI Digital Enterprise Research Institute
DIKW Data, Information, Knowledge, Wisdom
DMAP Data Map
DMP Datenmarktplatz
DSGVO Datenschutz-Grundverordnung
ERP Enterprise Resource Planning
ETL Extract, Transform, Load
EXIF Exchangeable Image File Format
GLD Government Linked Data
GPS Global Positioning System
GUID Globally Unique Identifier
HTTP Hypertext Transfer Protocol
IBM International Business Machines
IDC International Data Corporation
IoT Internet of Things
ISO International Organization for Standardization
IT Informationstechnik
JSON JavaScript Object Notation
LOM Learning Object Metadata
MARC Machine-Readable Cataloging
MES Manufacturing Execution System
MLOps Machine Learning Operations
MPEG Moving Picture Experts Group
MSE Mean Squared Error
NoSQL Not only SQL
PSNR Peak Signal to Noise Ratio
RDBMS Relationales Datenbankmanagementsystem
RDF Resource Description Framework
RPC Remote Procedure Call
SNR Signal to Noise Ratio
SOAP Simple Object Access Protocol
SQL Structured Query Language
THD Total Harmonic Distortion
VRA Visual Resources Association
XML Extensible Markup Language

1 Einleitung

Die Ausbreitung des Internets, das Aufkommen von Cloud-Diensten und sozialen Medien hat zu einer Datenexplosion geführt und dazu, dass fast jedes Unternehmen „digital“ geworden ist [1]. Wenn es sich um große Unternehmen handelt, können diese Hunderte oder sogar Tausende von verschiedenen Informations- und Datenbanksystemen nutzen [2]. In [2] wird beispielsweise angeführt, dass der Volkswagen-Konzern über rund 5000 und Daimler über 3000 Informationssysteme verfügt. Mit der wachsenden Anzahl von Datenquellen im Unternehmen steigt auch die Anzahl der Herausforderungen bei der Integration, Speicherung und Verarbeitung von Daten. Die Hauptprobleme bei der Datenverwaltung im Unternehmen bestehen in der großen Menge und der Heterogenität der Daten. Darüber hinaus existieren in der Unternehmensumgebung oft mehrere Datenmodelle und Speichersysteme, was das Auffinden, Verstehen, Zugreifen und folglich die effektive Nutzung von Daten erschwert. Werden Daten richtig verwaltet, so kann ein Unternehmen aus diesen Wert schöpfen und beispielsweise neue Geschäftsmodelle entwickeln, Prozesse verbessern oder neue Kundengruppen erschließen. [1]

Eine Lösung für das Problem ist die Nutzung eines internen Datenmarktplatzes. *Interner Datenmarktplatz (DMP)* bezeichnet eine digitale Plattform, auf der Informationsprodukte für Mitarbeiter und Geschäftspartner des Unternehmens zur Verfügung gestellt werden [3]. D.h., die Daten aus verschiedenen Unternehmenssystemen und Datenbanken können an einem Ort, auf dem Datenmarktplatz, registriert werden und so das Auffinden und Verständnis der Daten erleichtern. In Hinblick auf das Verständnis von Daten spielen Metadaten eine wichtige Rolle in der Datenverwaltung im Allgemeinen und unter anderem auf dem Datenmarktplatz. *Metadaten* sind Daten, die andere Daten beschreiben, also „Daten über die Daten“ [4]. Zu den Metadaten gehören z.B. Datentyp, Größe, Format, Qualität, Nutzungsbedingungen oder Informationen zum Data Owner. Somit spielen die Metadaten im Prozess der Datensuche und Auswahl eine zentrale Rolle auf dem Datenmarktplatz. Diese Arbeit, deren Aufgaben weiter präzisiert werden, hat zum Ziel, die für den internen Datenmarktplatz möglichen Metadaten zu sammeln und zu strukturieren.

1.1 Ziele der Arbeit

Ausgehend von den vorangegangenen Ausführungen lässt sich feststellen, dass das wichtigste Ziel dieser Arbeit darin besteht, eine Liste von relevanten Metadaten zu erstellen sowie ein Konzept ihrer Darstellung auf dem Datenmarktplatz zu entwickeln. Um die gesammelten Metadaten je nach Typ und Verwendungszweck der Daten zu differenzieren und zu strukturieren, werden Metadatentemplates erstellt. Es ist somit das Ziel dieser Arbeit die folgenden Themen zu bearbeiten:

1. Die im Unternehmen vorhandenen Datentypen und -speicher ermitteln.

2. Anwendungsszenarien für die Nutzung des Datenmarktplatzes erstellen sowie Anforderungen für die benötigten Metadaten auf dem Datenmarktplatz basierend auf den Anwendungsszenarien ableiten.

3. Metadatentemplates für verschiedene Datentypen entwickeln.

4. Alle in Templates gesammelten Metadaten für den Datenmarktplatz mit einem Metadatenmodell beschreiben.

5. Einen Überblick über die Metadatenmanagement-Tools erstellen, mit denen die Metadaten für die entwickelten Templates extrahiert werden können.

6. Ein Konzept für eine dynamische Darstellung unterschiedlicher Metadaten aus den Templates auf dem Datenmarktplatz ausarbeiten.

7. Alle entwickelten Konzepte basierend auf einer prototypischen Implementierung evaluieren.

1.2 Aufbau der Arbeit

Die Arbeit ist wie folgt aufgebaut: in Kapitel 2 werden die Funktionen und Rollen des internen Datenmarktplatzes, Datentypen, -quellen und -speicher im Unternehmen erläutert sowie Metadatentypen, -funktionen und -standards vorgestellt. Kapitel 3 befasst sich mit der Darstellung von Anwendungsszenarien für zwei unterschiedliche Datenmarktplatznutzer, auf deren Grundlage in Kapitel 4 die Anforderungen an den Datenmarktplatz hinsichtlich der Nutzung von Metadaten formuliert werden. In Kapitel 5 werden grundlegende Konzepte im Zusammenhang mit der Erfassung und Strukturierung von Metadaten vorgestellt. Die für die verschiedenen Datentypen entwickelten Templates werden in Kapitel 6 präsentiert, und Kapitel 7 beschreibt ein Metamodell für die in den Templates erfassten Metadaten. Die verschiedenen Metadatenmanagement-Tool-Typen, aus denen der Datenmarktplatz Metadaten extrahiert, werden in Kapitel 8 beschrieben. Kapitel 9 und 10 stellen ein Konzept zur Darstellung sowie die Umsetzung der Integration von Metadaten aus den Templates in den Datenmarktplatz-Prototyp dar. Die entwickelten Templates und der implementierte Prototyp werden in Kapitel 11 evaluiert. Zuletzt werden in Kapitel 12 die Ergebnisse der Arbeit zusammengefasst sowie ein Ausblick auf zukünftige Arbeiten gegeben.

2 Grundlagen

Um Metadatentemplates für den internen Datenmarktplatz zu erstellen, ist es zunächst notwendig, einige theoretische Grundlagen zu erfassen. Als erstes wird erläutert, was der interne Datenmarktplatz ist, welche Rollen und Komponenten er umfasst und welche Funktionen er erfüllt. Als nächstes wird untersucht, welche Typen von Produkten auf dem internen Datenmarktplatz angeboten werden können. Dies erfordert ein Verständnis dafür, welche Daten im Unternehmen vorhanden sind, wo sie gespeichert und wie sie verarbeitet werden. Schließlich ist es wichtig, im Detail zu verstehen, was Metadaten sind, welche verschiedenen Typen es gibt, wozu sie gebraucht werden und welche Rolle sie auf dem Datenmarktplatz spielen.

2.1 Interner Datenmarktplatz

Wie bereits in der Einleitung erwähnt, ist der interne Datenmarktplatz eine digitale Plattform, die den Austausch von Daten zwischen Mitarbeitern und ausgewählten Geschäftspartnern des Unternehmens ermöglicht [3]. An dieser Stelle sei darauf hingewiesen, dass es auch externe Datenmarktplätze gibt, auf denen einzelne Unternehmen und Privatpersonen Daten bereitstellen können und deren Hauptmotivation die Monetarisierung ist, d.h. der Erhalt einer Vergütung für die bereitgestellten Daten [3, 5]. Beispiele für solche Datenmarktplätze sind z.B. *Advaneo*[1], das sowohl kostenlose als auch kostenpflichtige Datensätze enthält, oder *AWS Marketplace*[2], der Datensätze für verschiedene Kategorien umfasst, darunter Datenverarbeitungs- und Softwareimplementierungsdienste. Darüber hinaus gibt es spezialisierte Datenmarktplätze, die Daten aus einem bestimmten Anwendungsbereich anbieten, wie z.B. *Otonomo*[3], das sich auf den Verkauf von Daten für die Automobilindustrie spezialisiert ist.

Das Ziel eines internen Datenmarktplatzes ist es, Daten im Unternehmen zu demokratisieren, d.h. für möglichst viele Mitarbeiter im Unternehmen bereitzustellen. Manchmal werden Daten in einzelnen Abteilungen des Unternehmens in verschiedenen lokalen Speichern abgelegt, was den Zugang für Mitarbeiter anderer Abteilungen erschwert. Außerdem können Daten in Datenbanken gespeichert werden, die aus technischer Sicht schwer zugänglich sein können. Das Hauptproblem dabei ist, dass die Mitarbeiter möglicherweise nicht wissen, welche Daten überhaupt im Unternehmen vorhanden sind, und daher nicht in der Lage sind, sie effektiv zu nutzen. In einem späteren Abschnitt wird ausführlich darauf eingegangen, welche Datenbanken und Datenspeicher im Unternehmen existieren. [3, 5, 6]

Im Folgenden wird ein Blick auf die wichtigsten Rollen und Komponenten des Datenmarktplatzes geworfen.

2.1.1 Komponenten und Rollen auf dem Datenmarktplatz

Um die Funktionen des Datenmarktplatzes betrachten zu können, ist es zuerst notwendig zu wissen, welche Rollen und Komponenten er hat. Alle diese Bestandteile und ihre Beziehungen

[1] Advaneo. Access to the world of data: https://www.advaneo.de
[2] AWS Marketplace: https://aws.amazon.com/marketplace
[3] Otonomo. Multi-Layered Vehicle Data. Fleet Data. Mobility Intelligence: https://otonomo.io

zueinander sind in Abbildung 1 dargestellt, welche aus [7] entnommen und abgeändert wurde.

Die zentrale Komponente des Datenmarktplatzes stellen die Daten selbst dar, d.h. die *Informationsprodukte*, die auf dem Datenmarktplatz zum Austausch angeboten werden. Es kann sich dabei nicht nur um Daten, sondern auch um datennahe Dienstleistungen handeln. Der zweite wichtige Bestandteil des Datenmarktplatzes ist der *Datenkatalog*, in dem die Informationen über die Datensätze auf dem Datenmarktplatz gespeichert sind. Der Katalog unterstützt in der Regel das Suchen und Klassifizieren von Daten. In Kapitel 8 werden die Kataloge, die mit dem Datenmarktplatz verbunden werden können, näher betrachtet. Ein weiterer notwendiger Baustein des Datenmarktplatzes ist das *Metadaten-Repository*, d.h. der Speicherort, an dem alle Metadaten der Datensätze gesammelt werden. Auf die Typen und Funktionen von Metadaten wird in Abschnitt 2.6 noch näher eingegangen. [3, 8, 9]

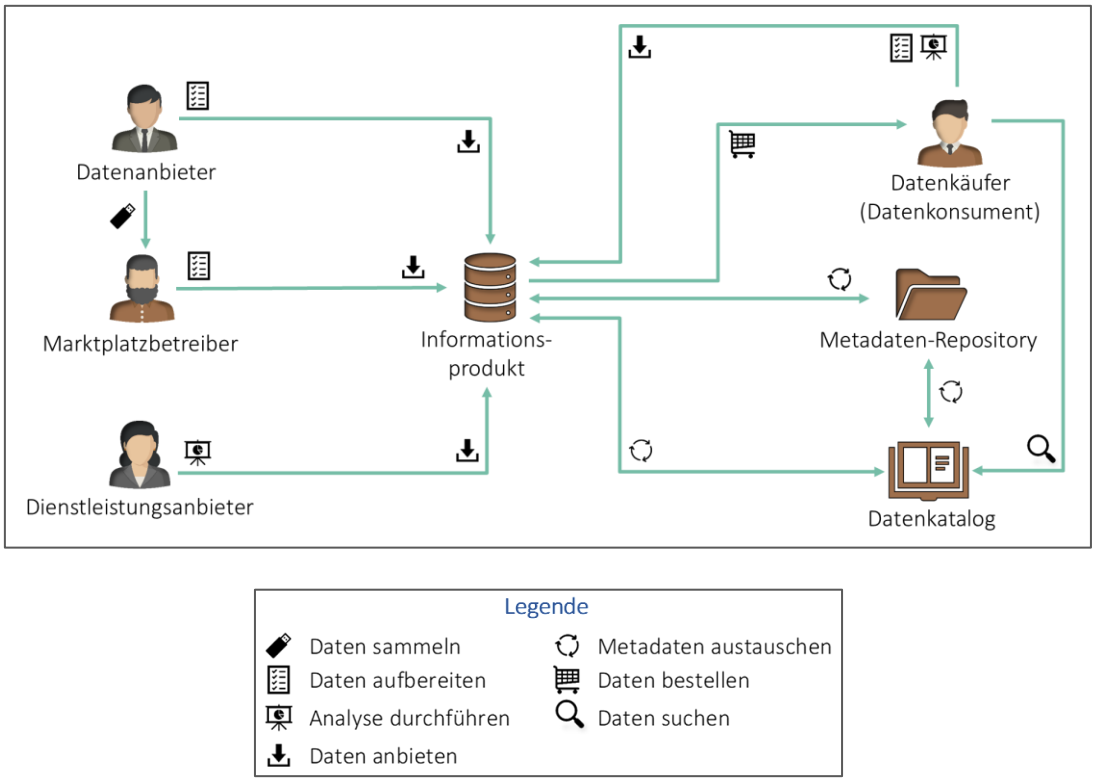


Abbildung 1. Hauptrollen und -komponenten auf dem Datenmarktplatz [7]

Auf dem Datenmarktplatz treten auch Akteure auf, die mit seinen Komponenten interagieren. In [3, 10] werden vier Hauptrollen genannt, nämlich Datenanbieter, Datenkäufer, Dienstleistungsanbieter und Marktplatzbetreiber, die im Folgenden ausführlicher beschrieben werden. Der *Datenanbieter* ist eine Person (im Falle des internen Datenmarktplatzes ein Mitarbeiter des Unternehmens), die die Daten besitzt und sie anderen im Unternehmen zur Verfügung stellen möchte. Der Datenanbieter muss sowohl die Daten selbst als auch die Nutzungsbedingungen beschreiben. Der *Datenkäufer* ist ein Mitarbeiter, der Daten des Unternehmens finden und darauf zugreifen möchte, um sie weiterzuverwenden, zu analysieren oder zu verändern und wieder auf dem Datenmarktplatz anzubieten. Das bedeutet, dass der Datenkäufer auch ein Datenanbieter sein kann und umgekehrt. Es sollte angemerkt werden, dass die Bezeichnung „Datenkäufer“ einen gewissen Bezug zu Geld

herstellt, was im Falle eines internen Datenmarktplatzes nicht angebracht ist, da sein Ziel vor allem in der Datendemokratisierung und nicht in der Monetarisierung besteht, weshalb im Folgenden die Bezeichnung *Datenkonsument* verwendet wird. Der *Dienstleistungsanbieter* vertreibt seine Dienstleistungen zur Datenanalyse, -bearbeitung oder -anreicherung auf dem Datenmarktplatz. Es kann auch einen Vermittler – den *Marktplatzbetreiber* – geben, der als Vermittler fungiert und die Daten der Datenanbieter sammelt, diese aufbereitet und auf dem Datenmarktplatz zur Verfügung stellt. Der Dienstleistungsanbieter und der Marktplatzbetreiber sind nicht zwingend auf dem Datenmarktplatz vorgesehen. [3, 10]

2.1.2 Funktionen des Datenmarktplatzes

Um besser zu verstehen, wie der Datenmarktplatz den Endnutzern hilft, Daten zu finden und letztendlich zu demokratisieren, ist es notwendig, seine Funktionen zu betrachten. Es ist zu bemerken, dass die Funktionen des Datenmarktplatzes in der Literatur nicht eindeutig genannt werden und auch von der Art des Datenmarktplatzes abhängen können. Da der Schwerpunkt dieser Arbeit auf dem internen Datenmarktplatz liegt, werden die wichtigsten Funktionen betrachtet, die für diesen relevant sind und auch in [3, 5, 9] erwähnt werden. Dazu gehören Datenaustausch, Datenintegration und -verwaltung, Gewährleistung der Datensicherheit sowie Verwaltung von Metadaten, Crowdsourcing und Administration.

Der *Datenaustausch* setzt die Möglichkeit voraus, Daten anzubieten und zu akquirieren. Neben der direkten Möglichkeit, Daten anzubieten und anzufragen, sollte der Datenmarktplatz eine erweiterte Suche mit der Sortierung und Filterung von Datensätzen unterstützen sowie die Möglichkeiten bieten, die Datensätze aufzulisten, die der Nutzer später erwerben möchte. Die Historie der bereits erworbenen Datensätze sollte den Nutzern ebenfalls zur Verfügung gestellt werden. [3, 5, 9]

Datenintegration und -verwaltung umfassen das Laden und das Zusammenführen von Daten aus verschiedenen Quellen, die Beschreibung, Kategorisierung und Katalogisierung dieser Daten, sowie ihre Analyse und Visualisierung. Dazu gehören auch die Überprüfung und Darstellung der Datenqualität, die Verfolgung von Datenänderungen, die Verwaltung der Historie dieser Änderungen und die Aktualisierung von Daten sowie die Bereitstellung von Daten z.B. über APIs oder Downloads, das Hinzufügen der Datenvorschau und Informationen darüber, wo die Daten gespeichert sind. Es sei darauf hingewiesen, dass die Darstellung der Datenqualität und die Verfolgung von Datenänderungen in Abschnitt 2.6.2 ausführlicher behandelt werden. [3, 9]

Aufgabe des Datenmarktplatzes ist auch die *Gewährleistung der Datensicherheit*, was sowohl die Überprüfung der Vertraulichkeit der Daten selbst als auch die Kontrolle und den Schutz der Nutzerkonten und ggf. die Anonymisierung der Daten umfasst [3].

Zu der *Verwaltung von Metadaten* auf dem Datenmarktplatz gehören der Import und die Integration von Metadaten aus unterschiedlichen Metadatenmanagement-Tools sowie die Festlegung obligatorischer Metadaten [3]. Die Rolle der Metadaten auf dem Datenmarktplatz wird in Abschnitt 2.7 detaillierter erläutert.

Eine weitere Funktion des Datenmarktplatzes besteht im *Crowdsourcing*, also dem Austausch von Erfahrungen über die Nutzung von Datensätzen zwischen den Mitarbeitern des

Unternehmens. Neben dem Schreiben von Feedback zu den Datensätzen kann Crowdsourcing auch darin bestehen, dass die Datenmarktplatznutzer an der Katalogisierung der Daten und dem Hinzufügen weiterer Informationen zu den Datensätzen sowie an der Analyse und Anreicherung der Daten mitwirken. [5]

Die *Administration* des Datenmarktplatzes umfasst sowohl die Erstellung und Pflege einer graphischen Oberfläche mit benutzerfreundlicher Navigation als auch die Verwaltung von Benutzerkonten, einschließlich der Registrierung neuer Benutzer und der Kontrolle des Zugangs zu Produkten des Datenmarktplatzes [3].

Als Weiteres ist es sinnvoll zu erörtern, welche Produkte auf dem Datenmarktplatz präsentiert werden können, was das Thema des nächsten Abschnittes sein soll.

2.2 Daten und Datentypen auf dem internen Datenmarktplatz

Das Hauptprodukt des Datenmarktplatzes, wie aus dem Namen bereits hervorgeht, sind Daten. Daher ist zunächst zu überlegen, was Daten sind und welche Typen von Daten es gibt. Dies ist notwendig, um die im Unternehmen vorhandenen Datentypen zu verstehen und dementsprechend möglichst viele dieser Typen auf dem internen Datenmarktplatz mit den Metadatentemplates, mit denen diese Arbeit verbunden ist, unterstützen zu können.

Der Begriff *Daten* bezieht sich auf eine Reihe von qualitativen oder quantitativen Werten [11], die interpretierbare Symbole oder Signale umfassen [12]. Daten, die in einem bestimmten Kontext analysiert wurden und einen bestimmten Wert für eine oder mehrere Personen haben, werden als *Informationen* bezeichnet. Wenn es möglich wird, Schlussfolgerungen aus diesen Informationen zu ziehen und auf dieser Grundlage zu handeln, ist das Ergebnis der Verarbeitung von Informationen *Wissen*. Wenn das Wissen durch ein Verständnis für seine mögliche Anwendung ergänzt wird, handelt es sich um *Weisheit*. Die Beziehung zwischen Daten, Informationen, Wissen und Weisheit ist in der DIKW-Pyramide (Data, Information, Knowledge, Wisdom) in Abbildung 2 dargestellt. Ganz unten stehen Daten als Grundlage für alle folgenden Stufen, gefolgt von Informationen und Wissen. Die Weisheit vervollständigt die Pyramide in ihrer Spitze. [12]

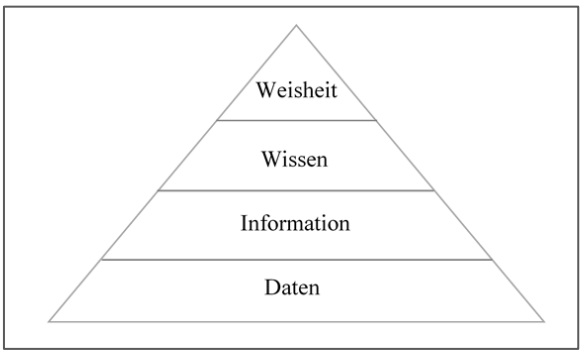


Abbildung 2. DIKW Pyramide [12]

Ein Beispiel für diese unterschiedlichen Stufen könnte wie folgt aussehen (Abbildung 3): es gibt ein Dokument, in dem Buchstaben, Zahlen und Symbole aufgezeichnet sind. Diese werden als Daten bezeichnet. Wenn diese Daten als Tabelle mit Spaltennamen dargestellt werden, lassen sich beispielsweise Informationen über die Lufttemperatur im Sommer und im

Winter gewinnen. Aus diesen Informationen kann darauf geschlossen werden, dass die Lufttemperatur im Winter niedriger ist als im Sommer – dies ist ein Wissen, das aus den Informationen gewonnen wurde. Wenn dieses Wissen genutzt wird, lässt sich daraus schließen, dass Häuser im Winter geheizt werden sollten, weil die Lufttemperatur niedriger ist – das ist die daraus resultierende Weisheit.

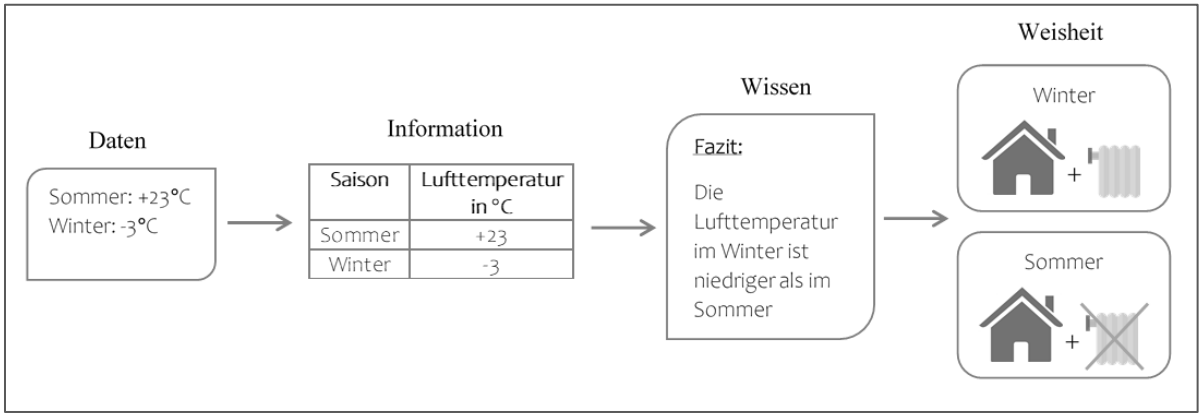


Abbildung 3. Beispiel: Daten, Information, Wissen und Weisheit

Es sei darauf hingewiesen, dass die Daten wiederum zu einem Typ gehören können. Der *Datentyp* bezieht sich in dieser Arbeit auf ein Merkmal von Datensätzen, das es ermöglicht, eine ungefähre Darstellung ihres Inhalts zu verstehen. Das kann z.B. eine Tabelle, ein Text, ein Bild, ein Audio oder ein Video sein. Auf alle diese Typen, die Unterschiede in ihrer Struktur und weitere Eigenschaften wird im Folgenden näher eingegangen.

2.2.1 Struktur der Daten

Hinsichtlich der Struktur lassen sich Daten in drei Typen unterteilen und zwar strukturierte, semi-strukturierte und unstrukturierte Daten [13].

In den Bereich *strukturiert* werden Daten eingeordnet, die eine streng definierte Struktur aufweisen, d.h. sie sind nach einem vorgegebenen Muster formatiert [13, 14]. Am häufigsten werden strukturierte Daten in Form von Tabellen dargestellt, z.B. als Excel-Tabellen oder Google Sheets-Dateien. Die zweite Art von Daten umfasst *unstrukturierte* Daten, also Daten, die keine klare Struktur haben und sich in absolut beliebiger Form darstellen lassen [15]. Unstrukturierte Daten werden von Menschen erstellt oder von Maschinen generiert, das können beispielsweise Logfiles, Videos, Audios, Texte oder Bilder sein [14, 16]. *Semi-strukturierte* Daten enthalten sowohl strukturierte als auch unstrukturierte Komponenten [1, 16]. Beispiele für semi-strukturierte Daten können CSV-Dateien, XML, JSON oder E-Mails sein. Wenn es sich z.B. um eine E-Mail handelt, stellt das E-Mail-Markup den strukturierten Teil davon dar, d.h. die Felder *Betreff*, *Text*, *Empfänger* und *Absender*. Der Text der E-Mail selbst ist hingegen unstrukturiert.

Alle Daten lassen sich laut [11, 16] ebenfalls in zwei weitere Kategorien einteilen: Roh- und bereinigte Daten. Dabei werden als *Rohdaten* solche bezeichnet, die keiner Bearbeitung, Auswertung oder Filterung unterzogen wurden und in der Form vorliegen, in der sie ursprünglich erzeugt wurden. *Bereinigte Daten* beziehen sich hingegen auf Daten, die in

irgendeiner Weise bearbeitet und geändert wurden, beispielsweise um ihre Qualität zu verbessern.

Hier soll deutlich gemacht werden, dass der Datenmarktplatz alle diese Daten abdecken soll, das bedeutet, er soll Daten unterschiedlicher Strukturen und Verarbeitungsgrade anbieten.

2.2.2 Unternehmensdaten

Da sich diese Arbeit mit einem internen Datenmarktplatz befasst, ist es wichtig die Definition von Unternehmensdaten zu verdeutlichen. Definitionsgemäß sind *Unternehmensdaten* die Daten, die von Mitarbeitern und Partnern eines Unternehmens genutzt werden und für dieses von Wert sind [1]. In [11, 17] werden drei Kategorien von Unternehmensdaten genannt und ihre Definitionen angegeben, die im Folgenden kurz erläutert werden. Dabei handelt es sich um Stamm-, Transaktions- und analytische Daten. *Stammdaten* beschreiben die Hauptobjekte des Unternehmens, d.h. alle Kunden-, Mitarbeiter- oder produktbezogenen Daten. *Transaktionsdaten* werden von verschiedenen Anwendungen und Geschäftsprozessen in einem Unternehmen generiert und sind immer mit einem Ereignis zu einem bestimmten Zeitpunkt verbunden. Dabei kann es sich beispielsweise um den Kauf eines Produkts oder um den Eingang eines Reklamationsschreibens handeln. *Analytische Daten* sind Daten, die durch Validierung und Analyse aus Stamm- und Transaktionsdaten gewonnen werden, z.B. verschiedene Berichte oder Kalkulationen. Ein interner Datenmarktplatz ist dafür konzipiert, den Mitarbeitern den Austausch all dieser Typen von Unternehmensdaten zu ermöglichen.

2.2.3 Menge an Daten

Mit der wachsenden Zahl von Datenquellen, dem Aufkommen des Internets der Dinge (IoT), Datenstreaming sowie verschiedenen Cloud-Technologien sehen sich Unternehmen zunehmend mit der Notwendigkeit konfrontiert, sogenannte Big Data zu speichern und zu verarbeiten [1, 13]. *Big Data* sind Daten, deren Größe oder Struktur es nicht erlaubt, sie mit Standardtechnologien zu extrahieren, zu verarbeiten und zu speichern [1]. Daten können als Big Data bezeichnet werden, wenn sie die sogenannten *4V* erfüllen: Variety, Velocity, Volume und Veracity. *Variety* bedeutet, dass alle Daten in unterschiedlichen Formaten vorliegen, unterschiedliche Strukturen und Verarbeitungsgrade aufweisen und aus heterogenen Quellen stammen. *Velocity* kann auf zwei Arten betrachtet werden: die Geschwindigkeit, mit der Daten generiert werden, ist groß, und die Möglichkeit, große Datenmengen in Echtzeit zu analysieren, kann begrenzt sein. *Volume* beschreibt die Datenmenge, die verarbeitet und analysiert werden muss, um daraus wertvolle Informationen zu extrahieren. Volume ist im Prinzip kein quantitativer Indikator, sondern vielmehr ein Konzept der Datenskala, das die Fähigkeit gängiger Anwendungen zur Verarbeitung von Daten charakterisiert. *Veracity* drückt sich darin aus, dass Richtigkeit, Authentizität, Vollständigkeit, Einzigartigkeit und Wertigkeit der Daten überprüft und garantiert werden können. Big Data kann jede Kombination aus strukturierten, semi-strukturierten und unstrukturierten Daten enthalten. [1, 11, 13, 15, 18]

Da Big Data im Unternehmen oft vorliegen, können sie auch auf dem Datenmarktplatz vorkommen. Dies können z.B. IoT-Daten sein, die aus Datensätzen mit unterschiedlichen Sensorwerten bestehen. Das bedeutet, dass der Datenmarktplatz auch für die Darstellung solcher Daten konzipiert werden soll.

2.2.4 Datenanfall und –verarbeitung

Daten im Unternehmen werden kontinuierlich und in großen Mengen aus heterogenen Quellen generiert, haben einen unterschiedlichen Verarbeitungs- und Bereinigungsgrad und liegen in verschiedenen Formaten vor [14, 16]. Die Datenmenge steigt dabei exponentiell an. Hier sei hervorgehoben, dass es sich bei den meisten Daten im Unternehmen, etwa 80%, um semi-strukturierte und unstrukturierte Daten handelt [13, 14, 16]. Neben neuen Daten speichern Unternehmen oft große Mengen veralteter Daten, falls diese für Analysen benötigt werden, die normalerweise archiviert werden [11].

Es ist zudem erwähnenswert, dass Batch- und Streaming-Daten existieren, die ebenfalls in das Unternehmen gelangen. *Batch-Daten* umfassen Dateneinheiten, die über einen bestimmten Zeitraum anfallen. Alle oder fast alle Daten in einem solchen Batch werden gleichzeitig analysiert und verarbeitet, was aufgrund der Größe der Daten einige Herausforderungen mit sich bringen kann. *Streaming-Daten* werden kontinuierlich in Echtzeit generiert. Dies sind Daten aus verschiedenen Geschäftsanwendungen, Sensoren, Webdienstprotokollen oder Daten über das Nutzerverhalten auf Webseiten (z.B. Informationen zu Klicks). Flugzeuge, medizinische Geräte, Aufzüge, Ölpipelines, Handys und Tablets erzeugen ständig Protokolldateien. Ein Merkmal von Streaming-Daten ist, dass sich ihr Wert nur manifestiert, wenn sie nacheinander in der Reihenfolge ihres Entstehens verarbeitet werden. [11, 18, 19]

Der Datenmarktplatz soll Informationen über all diese vielfältigen Daten, die auf unterschiedliche Weise, in unterschiedlichen Mengen und mit unterschiedlicher Geschwindigkeit im Unternehmen entstehen, denjenigen Mitarbeitern zur Verfügung stellen, die daran interessiert sind. Der Datenmarktplatz kann z.B. Protokolldaten von einigen Geräten in Form eines Abonnements anbieten. Das bedeutet, dass der Datenkonsument Zugang zu einem Datenstream für einen bestimmten Zeitraum erhält und diese Daten analysieren kann. Der Datenmarktplatz kann auch Informationen über veraltete und archivierte Daten zur Verfügung stellen, wenn deren Analyse dem Unternehmen in Zukunft von Nutzen sein kann. So können beispielsweise verschiedene Spezifikationen fehlerhafter Produkte verwendet werden, um die Ursachen von Produktfehlern herauszufinden und sie in Zukunft zu vermeiden.

2.3 Datenquellen im Unternehmen

Die Hauptmotivation des internen Datenmarkplatzes, wie bereits erwähnt, besteht in der Demokratisierung von Daten. Damit alle verfügbaren Unternehmensdaten katalogisiert und auf dem Datenmarktplatz präsentiert werden können, ist es zunächst notwendig zu verstehen, woher diese Daten stammen. Daten kommen aus internen und externen Quellen in das Unternehmen [1, 14]. *Intern* werden die Daten von den Mitarbeitern und unterschiedlichsten Unternehmenssystemen generiert [14, 15]. Auf diese Systeme wird in einem der nächsten Abschnitte näher eingegangen. Eine große Menge an Daten gelangt auch aus *externen* Quellen in das Unternehmen. Dies können Veröffentlichungen aus sozialen Netzwerken, Ergebnisse von Online-Umfragen, E-Mail-Nachrichten, verschiedene Dokumente [14], Daten aus Geolokalisierungssystemen [13] oder Regierungsdaten aus kostenlosen Portalen sein [15].

Es ist wichtig zu beachten, dass die gleichen Daten aus verschiedenen Quellen in das Unternehmen gelangen können [11, 14]. Beispielsweise kann derselbe Datensatz von Mitarbeitern verschiedener Abteilungen des Unternehmens gekauft werden, wenn diese nicht nachvollziehen können, welche Datensätze bereits vom Unternehmen erworben wurden [18]. All dies bestätigt einmal mehr, wie wichtig es ist, möglichst viele zusätzliche Informationen zu den Daten bereitzustellen und Daten im Unternehmen zu demokratisieren, um solche Szenarien zu vermeiden. Wie bereits erwähnt, kann dieses Problem durch die Nutzung des internen Datenmarktplatzes gelöst werden, der detaillierte Auskünfte über die im Unternehmen gespeicherten Daten liefert.

2.4 Datenspeicher im Unternehmen

Wie bereits aus dem vorherigen Abschnitt deutlich wird, werden Daten im Unternehmen in unterschiedlichen Formaten und großen Mengen gespeichert. Während mit Datenquellen alle Systeme und Technologien gemeint sind, die Daten produzieren, wird in diesem Unterkapitel über verschiedene Datenspeicher diskutiert, die in einem typischen Unternehmen vorhanden sind. Die Daten aus diesen Speichern müssen schließlich auf dem Datenmarktplatz dargestellt werden können.

2.4.1 Datenspeichertypen

Genauso wie bei Datenquellen gibt es zwei Haupttypen von Datenspeichern: intern und extern [11]. *Interne Speicher* existieren nur im Unternehmensnetzwerk und nur Mitarbeiter des Unternehmens haben Zugriff darauf. *Externer Speicher* umfasst alle Arten von Datenspeichern, die sich außerhalb des Unternehmensnetzwerks befinden.

Je nach Format und Verarbeitungsgrad können Daten in einem Unternehmen in relationalen und nicht relationalen Datenbanken sowie in Dateisystemen gespeichert werden [11, 20]. *Relationale Datenbanken* (*SQL*) enthalten nur strukturierte Daten, die in Form von Tabellen dargestellt werden [11, 13]. Dazu gehören z.B. PostgreSQL[4], SQLite[5] und MySQL[6]. Unterschiedliche Datensätze können über externe eindeutige Schlüssel miteinander verknüpft werden [1, 11]. *Nicht-relationale Datenbanken* (*NoSQL*) speichern in der Regel semi-strukturierte und unstrukturierte Daten und unterscheiden sich von relationalen in der Speicherstruktur und den Prinzipien der Arbeit mit Daten [1, 13]. Dies kann z.B. Key-Value-, spaltenweise, oder Indexspeicherung sowie Speicherung in Form von Dokumentenschlüsseln oder Graphen sein [11]. Nicht-relationale Datenbanken sind beispielsweise MongoDB[7] und Apache Cassandra[8]. *Dateisysteme* enthalten Daten jeglichen Formats und Verarbeitungsgrades; beispielsweise das verteilte Dateisystem Hadoop[9], in dem Dateien beliebiger Größe und Typen gespeichert werden können.

Der Datenmarktplatz sollte die Bereitstellung von Informationen über Daten aus all diesen Typen von Datenspeichern unterstützen sowie Angaben zu den Datenspeichern selbst

[4] PostgreSQL: The World's Most Advanced Open Source Relational Database: https://www.postgresql.org
[5] SQLite: https://www.sqlite.org
[6] MySQL: https://www.mysql.com
[7] MongoDB: https://www.mongodb.com
[8] Apache Cassandra. Open Source NoSQL Database: https://cassandra.apache.org
[9] Apache Hadoop: https://hadoop.apache.org

enthalten, z.B. ob es sich um einen internen oder externen Datenspeicher handelt, wer darauf Zugriff hat sowie wann und von wem die Daten in den Datenspeicher hochgeladen wurden.

2.4.2 Systemtypen

Relationale und nicht relationale Datenbanken sowie verschiedene Dateisysteme können in unterschiedlichen Unternehmenssystemen verwendet werden. Dabei kann es sich um analytische und operative Systeme handeln [21].

Operative Systeme

Zu den operativen Systemen gehören z.B.: ERP (Enterprise Resource Planning), CRM (Customer Relationship Management), MES (Manufacturing Execution System) und andere Systeme. Das *ERP-System* befasst sich mit der Verwaltung von Finanz-, Produktions- und Handelsaktivitäten des Unternehmens. Die Hauptaufgabe des *CRM-Systems* besteht darin, einen Kundenstamm zu bilden, Transaktionen mit Kunden zu steuern und Kundenbedürfnisse zu ermitteln. *MES* ist ein System zur Optimierung der Aktivitäten eines Unternehmens mit Hilfe der Dokumentation aller Produktionsprozesse. Dazu gehören die Visualisierung von Produktionsprozessen, die Erhebung quantitativer Produktions-indikatoren sowie Integration verschiedener Geräte. [17, 21, 22]

Analytische Systeme

Analytische Systeme können z.B. ein Data Warehouse und einen Data Lake umfassen [17, 21].

Das *Data Warehouse* enthält nur strukturierte, bereinigte und aktualisierte Daten, die in der Regel nach einem bestimmten Schema gespeichert werden und aus verschiedenen externen und internen Unternehmensdatenquellen stammen. Ein klassisches Data Warehouse besteht aus drei Bereichen: Staging-, Normalized- und Datamart-Bereich. Die Rohdaten gelangen in den *Staging-Bereich*, von wo sie in den *Normalized-Bereich* verschoben werden und erst nur dort den Mitarbeitern des Unternehmens zur Verfügung stehen. Im Normalized-Bereich können die Daten analysiert und anschließend in den *Datamart-Bereich* übertragen werden, wo verschiedene Datenausschnitte den Mitarbeitern einzelner Abteilungen bereitgestellt werden. Jeder der Bereiche hat sein eigenes Datenmodell. [11, 13, 19, 23]

Ein *Data Lake* ist ein interner Datenspeicher, der eine große Menge nicht nur verarbeiteter, sondern auch Rohdaten in verschiedenen Formaten und aus verschiedenen Quellen enthält, die für Analysen verwendet werden können. Je nach Verarbeitungsgrad der gespeicherten Daten wird der Data Lake in Zonen eingeteilt. [11, 13, 17, 19, 22, 24]

Nachfolgend wird eines der existierenden Zonenmodelle von Giebler et al. [22] betrachtet.

Die *Raw Zone* enthält normalerweise die Daten in ihrer ursprünglichen Form. Solche Daten wurden noch nicht verarbeitet und sind nicht für die Verwendung in Unternehmensanalyseberichten geeignet, können aber für Data Scientists oder für maschinelles Lernen nützlich sein. In dieser Zone befinden sich in der Regel die meisten Daten im Vergleich zu anderen Zonen. Daten in der *Harmonized Zone* werden aus der Raw Zone übertragen und stehen für Analyse und Verarbeitung zur Verfügung, sie können hier aus

mehreren Quellen kombiniert und angepasst werden. Auch die Datenintegrität und -qualität kann hier überprüft werden. Je nach Anwendungsfall werden die Daten in der *Distilled Zone* zusätzlich aufbereitet, ergänzt und ggf. in Semantik und Detaillierungsgrad verändert. Dieselben Daten können auf unterschiedliche Weise für verschiedene Zwecke verarbeitet werden. Die *Explorative Zone* wird verwendet, um Daten zu analysieren und Experimente mit Daten durchzuführen. Die Daten können hier beliebig und mit jeglichen Modellierungsmethoden modifiziert und verarbeitet werden, falls dies für die Analyse erforderlich ist. Für die Bearbeitung sensibler Daten gibt es einen separaten verschlüsselten Teilbereich, auf den Data Scientists nur mit besonderer Genehmigung zugreifen können. Die *Delivery Zone* enthält vorgefertigte analysierte Daten, die für die Verwendung in Berichten und Anwendungen geeignet sind, die einer breiten Palette von Mitarbeitern des Unternehmens zur Verfügung stehen und in verschiedene Analysetools importiert werden können. Diese Zone ähnelt dem Datamart-Bereich von Data Warehouse. [22]

Es ist wichtig zu betonen, dass sich Data Warehouse und Data Lake im Unternehmen keineswegs ausschließen, da sie voneinander abweichende Ziele verfolgen und ihre Konzepte unterschiedlich sind. Das Data Warehouse ist eher für Manager und Geschäftsangestellte gedacht, die fertige strukturierte Daten für spezielle Anwendungsfälle benötigen. Der Data Lake ist ein flexibleres Werkzeug zur Verarbeitung, Analyse und Verfeinerung einer riesigen Datenmenge, unabhängig von ihrer Struktur, und kann für Wissenschaftler, Forscher und Data Scientists praktisch sein. [11, 13]

Der Datenmarktplatz soll Daten sowohl aus operativen als auch aus analytischen Systemen unterstützen. Wie bei den einzelnen Datenbanken oder Dateisystemen, die im vorigen Abschnitt vorgestellt wurden, soll der Datenmarktplatz auch Informationen über die Unternehmenssysteme bereitstellen, in denen sich die Daten befinden, denn diese Informationen charakterisieren auch die Daten selbst. So gibt beispielsweise die Angabe der Data Lake Zone Auskunft über den Verarbeitungsgrad der Daten.

2.5 Datenlebenszyklus

Somit wird deutlich, dass die Daten, die in das Unternehmen gelangen, oft einen ziemlich komplizierten Weg durchlaufen, vom Laden bis zur abschließenden analytischen Verarbeitung. Unterschiedliche Grade der Datenverarbeitung sind auch für die Beschreibung und Einordnung der Daten auf dem Datenmarktplatz wichtig, weil das Verständnis des Verarbeitungsstadiums bestimmter Daten in manchen Fällen helfen kann, ihren Wert oder ihre Relevanz für bestimmte Aufgaben von Datenkonsumenten einzuschätzen. Wichtig ist hier auch zu verstehen, dass Daten nicht immer ein statisches Produkt sind. Wie bereits erwähnt, durchlaufen Daten einen bestimmten Lebenszyklus, entwickeln sich dynamisch, verändern sich und können in jeder Phase eine andere Darstellung und sogar einen anderen Zweck haben. Dieser Zyklus kann im Allgemeinen die folgenden Schritte umfassen: Datenerhebung, Bereinigen, Integrieren, Transformieren, Kombinieren von Daten aus verschiedenen Quellen, Datenanalyse, Aufbau eines Datenmodells und Daten-visualisierung [1]. Das Verschieben von Daten zwischen verschiedenen Systemen erfolgt mit Hilfe von *ETL-Tools (Extract, Transform, Load)*. Mit ETL werden Daten aus der Quelle extrahiert, vorverarbeitet und bereinigt, transformiert und in den endgültigen Speicher geladen [13].

Alle Phasen des Datenlebenszyklus sind auf dem Datenmarktplatz anzugeben, damit die Nutzer diese Informationen nachverfolgen können. D.h., wenn die auf dem Datenmarktplatz angebotenen Daten aus anderen Daten hergeleitet wurden, z.B. durch das Hinzufügen einer Spalte zu einer Tabelle, sollte dies in ihrer Beschreibung auf dem Datenmarktplatz angegeben werden. Eine solche Beschreibung könnte z.B. Informationen darüber enthalten, welche Spalte von wem und wann hinzugefügt wurde.

2.6 Metadaten

Da die Hauptaufgabe dieser Arbeit darin besteht, die verschiedenen Datentypen auf dem Datenmarktplatz mit Hilfe von Metadaten zu beschreiben, ist es notwendig, das Thema Metadaten ausführlicher zu diskutieren.

Wie in der Einleitung angegeben, ist die einfachste Definition von Metadaten „Daten über die Daten“ [4]. Der Begriff der Metadaten ist jedoch umfassender, denn die Funktionen von Metadaten, auf die später noch eingegangen wird, sind umfangreich [4, 25]. Eine ausführlichere Definition von Metadaten kann beispielsweise aus [26] abgeleitet werden; diese lautet wie folgt: *Metadaten* sind strukturierte Informationen über ein Produkt, die dem Benutzer helfen, die Merkmale dieses Produkts zu verstehen, ohne das Produkt selbst sehen oder analysieren zu müssen. Zu den Quellen von Metadaten im Unternehmen können alle in Abschnitte 2.2 – 2.4 beschriebenen Systeme und Datentypen gehören. Berichte, Tabellen, Dokumente, Datenbanksysteme, Data Lakes und Data Warehouses – all diese Daten und Systeme sind in der Regel mit Metadaten versehen. [4, 25, 26]

2.6.1 Metadatenfunktionen

Um zu klären, wofür Daten mit Metadaten versehen werden, ist es angebracht, die Hauptfunktionen von Metadaten zu betrachten. Die Hauptfunktionen von Metadaten für die verschiedenen Datentypen sind also die folgenden:

1. *Beschreibung der Informationsprodukte.* Das Hauptziel von Metadaten besteht darin, ein Informationsprodukt so zu beschreiben, dass seine Struktur, Herkunft, Qualität, Bedeutung, Anwendungsbereich und weitere Eigenschaften für die Person, die es nutzen wird, verständlich sind [25].

2. *Identifizierung und Auffindung der Informationsprodukte.* Metadaten werden verwendet, um das Auffinden von Informationsprodukten zu gewährleisten und ein Produkt von einem anderen zu unterscheiden. Dabei kann es sich sowohl um eine eindeutige Identifizierung handeln, z.B. durch eine GUID (Globally Unique Identifier), als auch um eine Kategorisierung der Produkte nach verschiedenen Eigenschaften. [4, 25]

3. *Vergleich verschiedener Informationsprodukte.* Sobald mehrere Produkte gefunden und identifiziert wurden, können sie auch, wiederum anhand von Metadaten, miteinander verglichen werden. Z.B., welche der beiden Dateien größer ist oder welche früher erstellt wurde. [25]

4. *Verfolgung des Datenschutzes und der Urheberrechte.* Neben der Beschreibung und Identifizierung von Produkten sollen Metadaten auch Informationen darüber liefern, ob

ein Informationsprodukt vertrauliche bzw. personenbezogene Daten enthält, sowie Informationen darüber, wer die Daten erstellt hat und wem sie gehören. [4]

5. *Beschreibung des Zugriffs auf die Produkte.* Eine weitere Funktion von Metadaten ist die Bereitstellung von Informationen über die Möglichkeiten des Zugriffs auf das Produkt, z.B. wer das Recht hat, darauf zuzugreifen, und wie genau der Zugriff erfolgen kann. [25]

Die Betrachtung der Funktionen von Metadaten lässt den Schluss zu, dass sie unterschiedlichen Zielen dienen, was bedeutet, dass einige Typen von Metadaten unterschieden werden können. Auf diese Typen wird im Folgenden ausführlicher eingegangen.

2.6.2 Metadatentypen

In der Literatur gibt es keine eindeutige Klassifizierung von Metadaten, aber einige Metadatentypen werden häufiger erwähnt als andere. Nach einem Überblick über einige der in [26–31] vorgestellten Klassifizierungen und unter Berücksichtigung der Besonderheiten der Datenprovisionierung auf dem Datenmarktplatz werden in dieser Arbeit vier Typen von Metadaten vorgestellt, die auch für die Erstellung von Templates relevant sind. Dabei handelt es sich um beschreibende, administrative, technische und Business-Metadaten, die im Folgenden näher erläutert werden.

Beschreibende Metadaten sind Metadaten, die dazu dienen, das Informationsprodukt, das sie betreffen, zu identifizieren und so zu beschreiben, dass die Benutzer es finden können. Z.B. den Namen, die Identifikationsnummer, die textliche Beschreibung, die Schlüsselwörter oder das Datum der Erstellung. Beschreibende Metadaten sollten für alle Informationsprodukte bereitgestellt werden, unabhängig von deren Typ. Somit erfüllen beschreibende Metadaten die ersten beiden Metadatenfunktionen. [26–28, 30]

Administrative Metadaten sind Metadaten, die Informationen über den Zugang zu den Daten und die Nutzungsbedingungen enthalten, z.B. wer der Data Owner ist und wer darauf zugreifen kann. Es handelt sich um Informationen darüber, wo die Daten gespeichert werden, ob sie vertraulich sind, wer sie besitzt und unter welchen Bedingungen er bereit ist, sie zur Verfügung zu stellen, und wie der Zugriff auf die Daten erfolgen kann. Daraus geht hervor, dass die administrativen Metadaten die Funktionen Nummer 4 und 5 übernehmen. [27, 28, 30]

Technische Metadaten zeigen die Darstellung von Daten bzw. erfassen die Form und Struktur der Daten, z.B. Größe, Format oder Typ. Metadaten von Tabellen, Indizes, Views sowie Beziehungen zwischen Tabellen sind ebenfalls technisch. Es ist möglich, mehrere Datensätze anhand von technischen Metadaten zu vergleichen, was der dritten Metadatenfunktion entspricht. [27–29]

Business-Metadaten sind Metadaten, die die Bedeutung von Daten erklären und Business-Begriffe definieren. In [27] wird auch darauf eingegangen, dass Business-Metadaten verschiedene Daten so ergänzen können, dass aus Daten Informationen werden, und dass Business-Analysten bereits in der Lage sein werden, diese Informationen in Wissen umzuwandeln. Daher decken Business-Metadaten auch die erste Funktion von Metadaten ab,

nämlich den Teil, der sich mit der Bedeutung der Daten befasst. Zu den Business-Metadaten gehören auch die Datenqualität und die Daten Lineage, die für den Datenauswahlprozess auf dem Datenmarktplatz besonders wichtig sind und im Folgenden näher vorgestellt werden. [27, 29, 31]

Wie aus Abschnitt 2.4 ersichtlich, können Daten aus verschiedenen Quellen extrahiert, in die operativen und analytischen Systeme des Unternehmens verschoben, dort verändert und verarbeitet, analysiert und ergänzt werden. In dem Abschnitt 2.5 wurde auch kurz auf den Lebenszyklus von Daten im Unternehmen eingegangen. Der Datenlebenszyklus stellt den Verlauf der Datenverarbeitung dar und wird als *Daten Lineage* bezeichnet [32]. Es sollte nun hinzugefügt werden, dass die Überwachung der Lineage eine wichtige Aufgabe im Unternehmen ist [19]. In [19] wird beispielsweise erwähnt, dass Informationen über jede beabsichtigte oder unbeabsichtigte Änderung von Daten wichtige Informationen im Unternehmen darstellen, da sie zur Überprüfung der Authentizität und der Integrität der Daten verwendet werden können. Und in [18] bestimmt die Verfügbarkeit von Auskünften über die Lineage der Daten das Ausmaß, in dem der Analytiker den Daten vertrauen kann.

Die *Datenqualität* bezieht sich, wie in [33] erwähnt, auf das Ausmaß, in dem die Daten der Realität entsprechen und die an sie gestellten Anforderungen erfüllen. Das Datenqualitätsmanagement in Unternehmen ist in der Regel eine wichtige Managementaufgabe, da es viele der im Unternehmen ablaufenden Prozesse beeinflusst [33]. So wird in [34] festgestellt, dass schlechte Datenqualität die Ursache für Fehlentscheidungen sein könnte. Die Datenqualität wird anhand sogenannter *Dimensionen* oder *Metriken* gemessen, wie z.B. Genauigkeit, Vollständigkeit oder Eindeutigkeit [33]. Die Qualität unstrukturierter Daten kann auch durch andere Parameter ausgedrückt werden, abhängig von dem Datentyp. Diese Parameter sowie die Metriken für die Datenqualität werden in den Abschnitten 6.4 – 6.6 für die verschiedenen Datentypen ausführlicher erörtert.

Es ist wichtig zu beachten, dass alle Typen von Metadaten wichtig sind, um eine bessere Verwaltung und ein besseres Verständnis der Daten zu erreichen. Diese Klassifizierung ist jedoch sehr grob und einige Metadatentypen können sich überschneiden und ähnliche Informationen über das Informationsprodukt enthalten. So lassen sich beispielsweise administrative, technische und Business-Metadaten auch als beschreibend bezeichnen, während administrative Metadaten auch über eine technische Komponente verfügen. [26, 27, 30, 35]

2.6.3 Metadatenstandards

Um Metadatentemplates für den Datenmarktplatz zu erstellen, wäre es hilfreich zu wissen, wie Metadaten dargestellt und wie sie beschrieben werden können, wozu die Metadatenstandards betrachtet werden können.

Metadatenstandards sind spezielle Dokumente, die Hinweise darauf geben, welche Metadaten für welche Datentypen geeignet sind. Sie dienen dazu, die Konsistenz von Metadaten zu gewährleisten und den Austausch von Metadaten zwischen verschiedenen Anwendungen und Organisationen zu ermöglichen, um das Auffinden der Daten zu erleichtern. Diese Standards können je nach Datentyp und Zweck variieren. Es lässt sich zwischen allgemeinen und spezialisierten Standards unterscheiden. [4]

Allgemeine Standards

Zu den *allgemeinen Metadatenstandards* gehören solche, die eine Reihe von Metadaten enthalten, die für jeden Datentyp anwendbar sind. Dazu gehören z.B. MARC21[10] (Machine-Readable Cataloging), DCAT[11] (Data Catalog Vocabulary) und Dublin Core[12].

MARC-Standard wurde 1969 von der Kongressbibliothek[13] in den USA entwickelt. Die *MARC21*-Version erschien 1999 und umfasst eine Reihe von Formaten, die einen Standard für die Katalogisierung von Metadaten von Text-, Software-, Kartographie-, Audio- und wissenschaftlichen Materialien darstellen. Solche Metadaten beinhalten in der Regel Titel, Kategorie, Anmerkungen, Datum der Veröffentlichung und Beschreibung. MARC21 umfasst fünf Formate, von denen jedes je nach seinem Zweck Metadatenstandards für einen bestimmten Typ von Daten enthält. Dies können Behörden-, bibliographische, Klassifikations- oder Bestandsdaten sowie Community Informationen sein. [26]

Der zweite erwägenswerte Standard ist *DCAT*. DCAT wurde vom Digital Enterprise Research Institute (DERI) erarbeitet, von der eGov Interest Group verbessert und im Jahr 2014 von der Government Linked Data (GLD) Working Group standardisiert. Der DCAT-Standard basiert auf dem Resource Description Framework (RDF), unterstützt die Beschreibung von Daten sowie ermöglicht die Kompatibilität zwischen verschiedenen Datenkatalogen (Datenkataloge als Metadatenmanagement-Tools werden auch in Kapitel 8 näher erläutert). Eine der im DCAT-Standard enthaltenen Entitäten, die für diese Arbeit von Interesse sein könnte, nennt sich *dataset*. Einige der Metadaten dieser Entität sind in Abbildung 4 dargestellt. Es ist auch möglich, zusätzliche Entitäten zu definieren sowie Daten nach Typen und Kategorien zu klassifizieren.

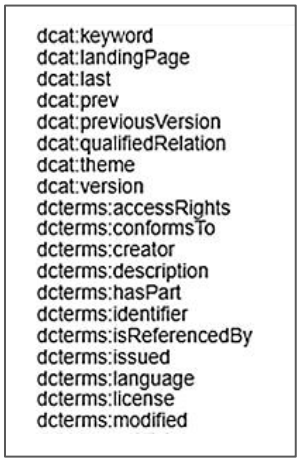


Abbildung 4. Teil des DCAT-Modells für eine *dcat:Dataset*-Entität

Ein weiterer Metadatenstandard ist *Dublin Core*, der von Mitgliedern einer Bibliotheksgemeinschaft in Dublin im Jahr 1995 entwickelt wurde. Dublin Core enthält 15 Kernelemente, die optional sind und sowohl mehrfach vorkommen als auch durch andere

[10] MARC Standards: https://www.loc.gov/marc/
[11] Data Catalog Vocabulary, Version 3: https://www.w3.org/TR/vocab-dcat-3/
[12] Dublin Core Standard: https://www.dublincore.org
[13] Library of Congress: https://www.loc.gov

Metadaten ergänzt werden können. Es wurden auch sogenannte *terms* definiert, die die Bedeutung der Kernelemente verdeutlichen. Für die Zwecke dieser Arbeit lohnt es sich, den Dublin-Core-Satz ein wenig genauer zu betrachten. Die Elemente dieses Satzes sind zusammen mit ihrer Beschreibung und Bezeichnung in Tabelle 1 aufgeführt.

Tabelle 1. Dublin Core Elemente

Dublin Core Element	Beschreibung des Elements	Bezeichnung (evtl. mit *term*)
Title	Name des Datensatzes	dc: title
Creator	Person, die für den Inhalt des Datensatzes verantwortlich ist.	dc: creator
Subject	Der Inhalt des Datensatzes, kann z.B. durch Schlüsselwörter oder Kategorien beschrieben werden.	dc: subject
Description	Beschreibung des Datensatzes	dc: description
Publisher	Person, die einen Datensatz zur Verfügung stellt, indem sie ihn veröffentlicht.	dc: publisher
Contributor	Person, die zur Erstellung des Datensatzes beigetragen hat.	dc: contributor
Date	Datum, an dem der Datensatz erstellt oder geändert wurde.	dc: date (dc: date.created dc: date.modified)
Type	Datentyp, der angibt, zu welcher Gruppe von strukturierten, semi-strukturierten oder unstrukturierten Daten der Datensatz gehört. Z.B. Text, Tabelle, Bild, Audio, Video	dc: type
Format	Datei-Format, in dem der Datensatz präsentiert wird.	dc: format (dc: format.extent)
Identifier	Eine eindeutige Nummer oder ein Link zum Datensatz, mit der/dem er eindeutig identifiziert werden kann.	dc: identifier
Source	Ein Informationsprodukt oder -system, von dem der aktuelle Datensatz abgeleitet wurde.	dc: source
Language	Sprache, in der die Daten im Datensatz vorliegen.	dc: language
Relation	Andere Datensätze, die in irgendeiner Weise mit dem aktuellen Datensatz in Verbindung stehen.	dc: relation
Coverage	Informationen über den Zeitraum oder den geografischen Ort, auf den sich die Daten beziehen.	dc: coverage
Rights	Festlegung der Urheberrechte, z.B. Lizenz	dc: rights (dc: rights.accessRights dc: rights.license)

Die dritte Spalte in Klammern enthält auch die terms, die den einen oder anderen Parameter spezifizieren. Z.B. kann *date* sowohl das Erstellungsdatum (*dc:date.created*) als auch das

Änderungsdatum (*dc:date.modified*) eines Datensatzes bezeichnen. Die Dateigröße wird durch *dc:format* mit *dc:format.extent* angegeben, und *dc:rights.accessRights* und *dc:rights.license* bezeichnen Zugriffsregeln bzw. Lizenzinformationen. Weitere Betrachtungen können auch in spezialisierten Standards bestehen.

Spezialisierte Standards

Spezialisierte Metadatenstandards unterscheiden sich von allgemeinen Standards dadurch, dass sie für die Beschreibung bestimmter Datentypen konzipiert sind und nur Metadaten enthalten, die für diese Typen relevant sind [4, 36]. Z.B. Standards zur Beschreibung von Kunstwerken und Kulturgütern wie der VRA[14] (Visual Resources Association) Core, Learning Object Metadata (LOM) Standards [37] oder Audio- und Filmstandards wie EXIF[15] (Exchangeable Image File Format) oder MPEG-7 (von der Moving Picture Experts Group) [38]. Auf die letztgenannten beiden Standards für unstrukturierte Daten wird in dieser Abhandlung noch näher eingegangen werden.

Einer der bekanntesten Standards für Bildmetadaten, der *EXIF-Standard*, ermöglicht das Hinzufügen von Metadaten zu verschiedenen Dateiformaten, wie z.B. JPG, TIFF oder PNG. Diese Metadaten können z.B. Informationen über den Autor, die Nutzungsbedingungen, GPS-Koordinaten (Global Positioning System), Schlüsselwörter, Kamera- und Bildspezifikationen enthalten. Metadaten können entweder von der Kamera zum Zeitpunkt der Aufnahme erzeugt oder manuell eingegeben werden.

Ein weiterer Standard, *MPEG-7,* umfasst einen Satz von Regeln zur Beschreibung von Multimedia-Inhalten, um diese zu standardisieren. Er kann Bild-, Audio- und Video-Metadaten enthalten. Berücksichtigt werden den Datentyp, ihre Nutzungs- und Zugangsbedingungen, ihr Inhalt, mögliche Klassifizierungen sowie ihr Verhältnis zu anderen Informationsressourcen. Beispiele für Felder sind Farbe, Textur, Form, Bewegung, Lokalisierung, Signalparameter und spektrale Eigenschaften. Einige dieser Metadaten werden auch in Abschnitt 6.6 ausführlicher behandelt. [39]

Da die beschriebenen Metadatenstandards eine große Anzahl verschiedener Metadaten enthalten, sowohl allgemeine als auch spezialisierte, besteht es die Möglichkeit, Metadaten für verschiedene Datentypen für Metadatentemplates aus diesen Standards abzuleiten.

2.6.4 Anforderungen an Metadaten

Damit Metadaten ihre Funktionen erfüllen können, müssen sie bestimmte Anforderungen einhalten. Die wichtigsten Parameter, die bestimmen, ob Metadaten für die Datenverwaltung verwendbar und nützlich sind, werden in Abbildung 5 dargestellt. Diese umfassen Qualität, Reichhaltigkeit und Kompatibilität. Alle diese Parameter sind es ebenfalls wert, etwas genauer untersucht zu werden. [25–27]

Die *Qualität der Metadaten* hängt immer von dem konkreten Anwendungsfall ab, vom Nutzer und von dem, was er sucht. Dazu können einige grundlegende Merkmale wie Aktualität,

[14] VRA CORE Schemas & Documentation: https://www.loc.gov/standards/vracore/schemas.html
[15] Exchangeable Image File Format: https://www.loc.gov/preservation/digital/formats/fdd/fdd000146.shtml

Genauigkeit, Eindeutigkeit, Zugänglichkeit, Herkunft, Konsistenz und Vollständigkeit gehören. [25]

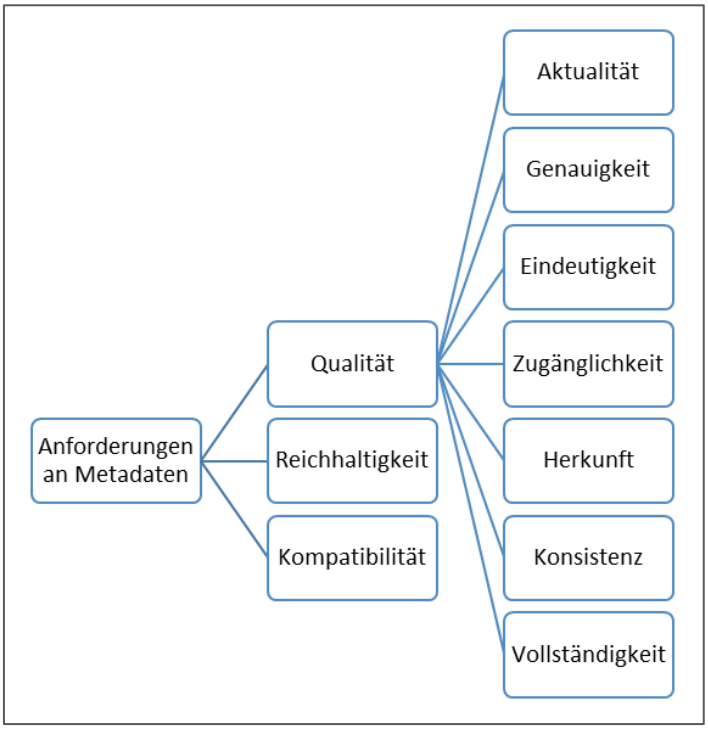


Abbildung 5. Anforderungen an Metadaten

Aktualität bedeutet, dass die Metadaten dem aktuellen Stand des Informationsprodukts, das sie beschreiben, entsprechen [25, 27]. Wenn z.B. eine Tabelle aktualisiert wurde und sich die Anzahl der Zeilen erhöht hat, die Metadaten aber nicht geändert wurden und eine veraltete Anzahl von Zeilen anzeigen, sind die Metadaten nicht aktuell. *Genauigkeit* der Metadaten bedeutet, dass die Metadaten mit dem Inhalt des Informationsprodukts übereinstimmen [25, 27]. Wenn ein Datensatz, der z.B. „Gewinnbericht 2020" heißt und vom Typ *text* ist, die Bilder von Produkten aus der Produktionsabteilung des Unternehmens enthält, dann sind die Metadaten nicht genau. Der nächste Parameter der Metadatenqualität ist die *Eindeutigkeit*. Dies bedeutet, dass die Metadaten keine inkonsistenten Informationen enthalten sollten [25]. Wenn das Feld *owner* beispielsweise *Anny Smile* lautet, kann das Auftauchen eines zusätzlichen Feldes *data holder* mit dem Wert *Stefanie Schulze* den Benutzer nur verwirren. Die *Zugänglichkeit* von Metadaten setzt voraus, dass sie vom Nutzer leicht gelesen und verstanden werden können [25, 26]. Wenn der Benutzer nach Informationen über z.B. die Dateigröße zunächst lange suchen muss und dann den Wert *5591736* ohne irgendwelche Einheiten oder Erklärungen findet, dann können solche Metadaten nicht als zugänglich bezeichnet werden. Auch die *Herkunft der Metadaten* ist wichtig, d.h. wer und wann sie hinzugefügt hat oder wie sie automatisch gewonnen wurden. Diese Informationen sind wichtig, um die Zuverlässigkeit der Metadaten beurteilen zu können [27]. Als Beispiel können hier zwei Szenarien angeführt werden. Bei dem einen sieht der Datenmarktplatznutzer, dass die Metadaten automatisch aus der Daten selbst extrahiert wurden (z.B. die bereits erwähnten Metadaten im EXIF-Format). Im zweiten Szenario wird die Quelle der Metadaten nicht angegeben. Das Vertrauen in die Metadaten und damit in die Daten selbst wäre im ersten Szenario höher. Die *Konsistenz* der Metadaten besteht darin, dass sie für alle verfügbaren Datensätze in der gleichen Form dargestellt werden und die gleiche

Bedeutung haben [25]. Wenn ein Datensatz z.B. das Feld *keywords* und ein anderer das Feld *tags* enthält, ist es für den Benutzer möglicherweise nicht klar, ob diese Felder dasselbe bedeuten oder unterschiedlich sind. *Vollständigkeit* der Metadaten setzt voraus, dass die Informationsressource so vollständig wie möglich beschrieben wird [25]. Das bedeutet, dass nach Möglichkeit keine vorgesehenen Metadatenfelder leer gelassen werden sollten. Natürlich ist es nicht immer möglich, alle potenziellen Metadaten zu erhalten, aber im Idealfall sollte der Nutzer keine Fragen zu den wichtigsten Merkmalen des Informationsprodukts haben. Metadaten sollten auch für alle durchsuchbaren und auswählbaren Produkte verfügbar sein [25].

Die *Reichhaltigkeit der Metadaten* bezeichnet die Menge der nützlichen Informationen, die sie enthalten und wird durch den Detaillierungsgrad der Datenbeschreibung charakterisiert [25]. An dieser Stelle sei darauf hingewiesen, dass der Detaillierungsgrad der Datenbeschreibung von dem jeweiligen Anwendungsfall abhängt und ein abstrakter Parameter ist. Zur besseren Verdeutlichung des Konzepts der Reichhaltigkeit der Metadaten kann das folgende Szenario als Beispiel angeführt werden: der Benutzer sucht eine Tabelle mit Bewegungssensordaten und findet einen Datensatz mit den Namen "Sensor_data_cleaned", dessen Beschreibung nur grundlegende Informationen wie Data Owner, Erstellungsdatum, Format und Dateigröße enthält. Der Name des Datensatzes lässt vermuten, dass die Daten in irgendeiner Weise bereinigt wurden, aber es gibt keine Informationen darüber, was genau gemacht wurde und von wem. Es ist ebenfalls nicht bekannt, wo die Daten im Unternehmen gespeichert sind, mit welchem Sensor und wann sie erhoben wurden, wie viele Zeilen und Spalten die Tabelle enthält usw. Solche Metadaten können offensichtlich nicht als reichhaltig bezeichnet werden, da sie keine detaillierten Informationen über den Datensatz liefern. Zu beachten ist zudem, dass sich die Reichhaltigkeit und Vollständigkeit der Metadaten auf unterschiedliche Merkmale beziehen, obwohl sie in einem Zusammenhang stehen. Bei der Erfassung von Metadaten sollte zunächst bestimmt werden, welche Metadaten für welche Datentypen relevant sind, wobei alle Besonderheiten der Daten zu berücksichtigen sind. Die gesammelten potenziellen Metadaten (Metadatenfelder) stellen die Reichhaltigkeit der Metadaten dar. Vollständigkeit bedeutet, dass möglichst viele dieser gesammelten Metadaten (Metadatenfelder) mit einem Wert versehen sind.

Die letzte Anforderung an Metadaten, die *Kompatibilität*, bezieht sich auf die Fähigkeit, Metadaten sowohl innerhalb des Systems als auch mit anderen Systemen auszutauschen. Im Sinne dieser Arbeit ist der Datenmarktplatz als ein System zu verstehen. Die Kompatibilität innerhalb des Systems kann durch die Erstellung von Metadatentemplates erzielt werden, um die es in dieser Arbeit geht. Eine externe Kompatibilität kann durch die Verwendung der bereits erwähnten Metadatenstandards gewährleistet werden. [25]

2.7 Die Bedeutung von Metadaten für den Datenmarktplatz

Nachdem nun die Komponenten, Rollen und Funktionen des Datenmarktplatzes sowie die Funktionen und Typen von Metadaten betrachtet wurden, können Schlussfolgerungen in Hinblick auf die Bedeutung von Metadaten für den Datenmarktplatz gezogen werden. Obwohl die Verwaltung von Metadaten als eigenständige Funktion des Datenmarktplatzes hervorgehoben wird, sind in Wirklichkeit viele Funktionen mit der Verwendung von Metadaten verbunden.

Mit beschreibenden Metadaten können die Mitarbeiter des Unternehmens beispielsweise Informationen über die im Unternehmen gespeicherten Datensätze finden und sie einsehen, was den Datenaustausch verbessert. Mit Hilfe von Business- und technischen Metadaten können Informationen über die Bedeutung und Eigenschaften von Daten abgerufen werden. Des Weiteren können die Qualität und der Lebenszyklus von Daten überwacht werden, woraus sich wiederum Rückschlüsse darauf ziehen lassen, ob die Daten vertrauenswürdig sind. Business-Metadaten tragen somit zur Datenverwaltung bei. Informationen über die Sicherheit der Daten, den Speicherort und die Zugriffsmöglichkeiten zählen zu den administrativen Metadaten.

Somit ermöglicht die Einbeziehung von Metadaten auf dem Datenmarktplatz die Wiederverwendung von Daten. Das Abrufen von Daten und die Auswahl von Daten erfolgen mit Hilfe von Metadaten. Dies umfasst die Extraktion von Metadaten aus Daten und deren Speicherung in Metadaten-Repository sowie die anschließende Präsentation dieser Metadaten für die Nutzer des Datenmarktplatzes. Wenn die Daten katalogisiert und mit Metadaten versehen sind, ist es für die Mitarbeiter des Unternehmens möglich, Datensätze zu finden, die sich auf verschiedene Anwendungsfälle und Probleme beziehen. [26, 27]

Die Vorteile der Verwendung von Metadaten auf dem Datenmarktplatz sind daher folgende:

- es werden mehr Unternehmensdaten verwendet, was bedeutet, dass mehr Wert aus den Daten gezogen und somit mehr Wissen erworben wird;
- dieselben Daten werden nicht mehrmals von verschiedenen Mitarbeitern gekauft, was die Anzahl der Datenduplikate im Unternehmen reduziert und somit den Speicherplatz freigibt;
- es werden keine Analysen und Berichte zweimal erstellt, was den Zeitaufwand der Mitarbeiter verringert und zu einer schnelleren Erledigung der Aufgaben beiträgt.

3 Anwendungsszenarien zur Nutzung des Datenmarktplatzes

Um Anforderungen an den Datenmarktplatz hinsichtlich der produktiven Nutzung der Metadaten zu formulieren und die erstellten Metadatentemplates bewerten zu können, werden zwei Szenarien zur Nutzung des Datenmarktplatzes geschildert. Die Hauptrolle im ersten Szenario spielt ein *technischer Nutzer* des Datenmarktplatzes, der sich hauptsächlich für strukturierte Daten, die in den relationalen Datenbanken gespeichert sind, sowie eine statistische Datenanalyse interessiert. Das zweite Szenario richtet sich an einen *nicht technischen Nutzer*, der Berichte über bestimmte Ereignisse sucht und keine genauen numerischen Daten benötigt. Der technische Nutzer ist ein *Datenanalyst*, der nach [40] an der Visualisierung und Analyse von Daten in einem Unternehmen beteiligt und für Entscheidungen auf Basis der analysierten Daten verantwortlich ist. Nicht-technische Aufgaben können von einem *Business-Nutzer* (in [41] auch *Business-Analyst* genannt) ausgeführt werden, dessen Interessen sich häufiger auf die Lösung von Business-Problemen konzentrieren, beispielsweise solche, die mit der Interaktion eines Unternehmens mit Kunden zusammenhängen [40]. Solche Aufgaben erfordern keine technischen Fähigkeiten, sondern eine Businesserfahrung [41]. In den folgenden Szenarien werden Anwendungsfälle für die Produktionsabteilungen (Szenario 1) sowie den Kundenservice (Szenario 2) eines Unternehmens beschrieben, das Holzleisten für Fußböden herstellt.

3.1 Szenario 1. Analyse von Produktmängeln

Es ist bekannt, dass einige Produkte, die in der Vergangenheit von dem Unternehmen hergestellt wurden, eine große Anzahl von Mängeln während der Lagerphase aufwiesen, woraufhin die Produktion dieses Produkts eingestellt wurde. Um die Produktion wieder aufnehmen zu können, aber dennoch zukünftig das Auftreten fehlerhafter Produkte zu verhindern, möchte ein Datenanalyst Ursachen von Produktmängeln herausfinden, einschließlich der Untersuchung der Abhängigkeit der Qualität der Ware von der Temperatur und Luftfeuchtigkeit des Raums, in dem diese Produkte gelagert wurden. Der Datenanalyst weiß mit Sicherheit, dass diese Produkte im Laufe von zwei Jahren produziert und in drei verschiedenen Lagerhäusern gelagert wurden, in denen jeweils Temperatur- und Luftfeuchtigkeitssensoren installiert waren (gezeigt in Abbildung 6). Alle relevanten Daten sind über den Datenmarktplatz zugänglich.

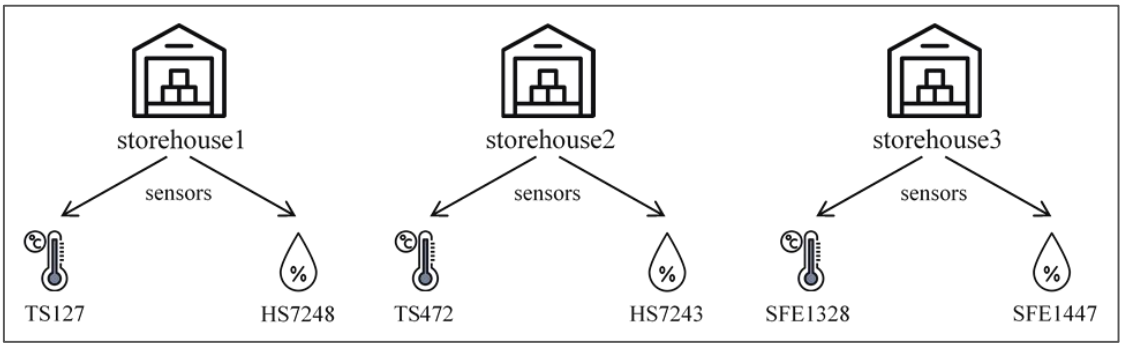


Abbildung 6. Lagerhäuser, in denen Holzleisten zwischengelagert werden und welche jeweils mit Temperatur- und Luftfeuchtigkeitssensoren ausgestattet sind

Um die Analyse durchzuführen, möchte der Datenanalyst die folgenden Datensätze über den Datenmarktplatz finden:

1. Daten aller Temperatur- und Feuchtigkeitssensoren, die in den Warenlagern installiert wurden
2. Informationen darüber, wie viele fehlerhafte Produkte für jedes der Lager vorhanden waren
3. Informationen über die technischen Eigenschaften der Sensoren
4. Technische Eigenschaften der Lager, in denen die Produkte gelagert wurden
5. Fotos von fehlerhaften Produkten

Als Suchabfrage wird der Begriff *wood plank* verwendet. Der Analyst hat die folgenden Anforderungen an die erforderlichen Daten:

S1A1: Die Sensordaten sollten für die Analyse geeignet sein, indem sie in Form einer Tabelle vorliegen.

S1A2: Die Sensordaten sollten vorverarbeitet/bereinigt vorliegen. Dazu gehört beispielsweise, dass Ausreißer entfernt oder Zeilen mit fehlenden Temperatur- oder Luftfeuchtigkeitswerten gelöscht wurden.

S1A3: Die Sensordaten sollten Informationen über die Modelle der verwendeten Sensoren und deren Software enthalten, da einige Sensormodelle bekanntermaßen anfänglich falsche Messungen geliefert haben, sodass die Ergebnisse, die durch Messungen mit den Sensoren des Sens123XX-Modells erhalten wurden, nicht berücksichtigt werden sollten.

S1A4: Alle Aktionen (einschließlich Bereinigung), die an den Daten durchgeführt wurden, sollten nachverfolgt werden können, damit genau beurteilt werden kann, was getan wurde.

S1A5: Für die Spalten mit der Temperatur und Feuchtigkeit sind detaillierte Beschreibungen wünschenswert, einschließlich der minimalen und maximalen Werte. Dies erklärt sich dadurch, dass bestimmte Maximal- und Minimalwerte von Temperatur und Luftfeuchtigkeit das Auftreten von Produktfehlern ausschließen und ein Lagerfach mit einer solchen Temperatur oder Luftfeuchtigkeit nicht in Betracht gezogen werden kann.

S1A6: Wichtig ist, dass die Daten genau, vollständig und konsistent sind.

S1A7: Beispieldaten sollten vor dem Zugriff einsehbar sein.

S1A8: Der Data Owner soll bekannt sein. Es ist in diesem Fall wünschenswert, dass die Daten von einem Mitarbeiter der Produktionsabteilung angeboten werden, da diese Abteilung die Holzleisten produziert hat.

S1A9: Die Daten sollten öffentlich lizenziert sein, damit sie für einen eigenen Anwendungsfall genutzt werden dürfen.

S1A10: Die Provisionierung von Daten ist in Form eines Zugriffs auf die interne Datenbank gewünscht, um nicht alle Daten auf einmal, sondern auch Teile davon anfragen zu können.

S1A11: Fotos von fehlerhaften Produkten sollten mit mindestens 0,3 Megapixeln, drei Farbkomponenten sowie mit wenigstens 400 x 400 Pixeln Auflösung und möglichst ohne Blitzlicht aufgenommen werden.

3.2 Szenario 2. Analyse der Kundenabwanderung

Aufgrund der zunehmenden Kundenabwanderung möchte sich der Business-Analyst eine Reihe von Berichten sichten, die sich auf die Interaktion des Unternehmens mit seinen Kunden beziehen, um Möglichkeiten zu finden, die Kundenabwanderung zu reduzieren. Dazu gehören E-Mails mit Beschwerden und Anregungen von Kunden, Berichte über Aktionen und Mailings sowie Kundenbewertungen zu den Produkten des Unternehmens.

Die Ausdrücke *customer support* oder *customer care* werden als Suchanfrage dienen. Der Business-Analyst möchte die folgenden Datensätze finden:

1. E-Mails von Käufern an die Support- und Garantieservices
2. Berichte über die vom Unternehmen durchgeführten Aktionen und Mailings
3. Audioaufnahmen von Telefongesprächen mit Kunden
4. Kundenbewertungen in Foren und sozialen Netzwerken

Die Anforderungen des Business-Analysten an die erforderlichen Datensätze sind folgende:

S2A1: Die Daten sollten zum Download bereitgestellt sein, da die Ausarbeitung von Daten in einem speziellen Programm auf dem Computer des Business-Analysten durchgeführt werden soll.

S2A2: Alle Datensätze sollten anonymisiert oder alle personenbezogenen Daten auf andere Weise verborgen werden. Informationen darüber, wie und von wem die Daten anonymisiert wurden, sollten ebenfalls verfügbar sein.

S2A3: Jahres- und Quartalsberichte über durchgeführte Aktionen und Mailings sind in Textform gewünscht, nicht als gescanntes Dokument, um darin suchen zu können.

S2A4: Die Dauer von Audioaufnahmen von Gesprächen mit Kunden sollte 3 – 5 Minuten nicht überschreiten. In solchen Aufzeichnungen geht es darum zu verstehen, wie die Mitarbeiter des Unternehmens den Kunden grüßen, wie sie sich vorstellen sowie wie freundlich und hilfsbereit sie zu Beginn des Gesprächs sind. Längere Tonaufnahmen, die ebenfalls auf dem Datenmarktplatz angeboten werden können, dürfen daher im Rahmen dieses Anwendungsfalles nicht berücksichtigt werden.

S2A5: Eine Übersicht von Kundenbewertungen in Foren und sozialen Netzwerken sollten sich auf den Kundendienst beziehen und folgende Schlüsselwörter (keywords) enthalten: *service, support, warranty service, customer support* oder *customer care.*

4 Anforderungen an den Datenmarktplatz

Wie in Abschnitt 2.1.2 bereits erwähnt, ist die Verwaltung von Metadaten eine der Hauptfunktionen des Datenmarktplatzes, was bedeutet, dass der Datenmarktplatz eine produktive Nutzung der Metadaten gewährleisten sollte, damit die Nutzer Datensätze provisionieren, suchen sowie die Bedeutung und den Verwendungszweck der Daten besser verstehen können. Für die Erstellung von Metadatentemplates ist es daher notwendig, Anforderungen an den Datenmarktplatz in Hinblick auf die Nutzung der Metadaten zu erheben. Diese Anforderungen werden auf Basis der Szenarien in Kapitel 3 abgeleitet und lassen sich in funktionale und nicht-funktionale Anforderungen unterteilen. Die funktionalen Anforderungen beziehen sich auf die Metadaten, die auf der Detailseite zu einem Datenprodukt hinzugefügt werden müssen, während die nicht-funktionalen die technischen Eigenschaften der Darstellung dieser Metadaten auf dem internen Datenmarktplatz enthalten.

Zu den *funktionalen* Anforderungen gehören:

A1: Datensätze sollten mit *beschreibenden Metadaten* versehen werden, die unter anderem eine eindeutige Identifizierung ermöglichen und zur Suche auf dem Datenmarktplatz genutzt werden können. Dies ist z.B. der Name des Datensatzes, sein Typ oder seine Beschreibung.

A2: Die Detailseite soll spezifische *administrative Metadaten* enthalten, wie z.B. Informationen zum Data Owner, Nutzungsbedingungen und Daten-Provisionierungsoptionen. Außerdem sollte der Benutzer nachvollziehen können, mit welchem Tool die Metadaten extrahiert wurden.

A3: Je nach Struktur der im Datensatz enthaltenen Daten sollten deren *technische Metadaten* verfügbar sein, die Informationen über Format, Struktur, Typ und Eigenschaften der Daten enthalten.

A4: *Business-Metadaten*, die dazu dienen, die Daten für nicht-technische Nutzer zu erklären, sollten dem Benutzer ebenso zur Verfügung stehen. Zu solchen Metadaten gehört z.B. die Bedeutung einer Tabellenspalte, d.h. die Informationen, die den Inhalt der Daten beschreiben.

A5: Der *Datenspeicherort* soll bekannt sein. Falls die Daten in einem Unternehmenssystem vorliegen (z.B. in einem Data Lake oder Warehouse), dann sollten für dieses System zusätzliche system-spezifische Metadaten vorhanden sein. Dabei sollte der Datenverarbeitungsgrad nachvollziehbar sein; beispielsweise durch Angabe der Data Lake oder Warehouse Zone, in der sich die Daten befinden.

A6: Die *Daten Lineage*, durch die der Lebenszyklus der Daten verfolgt wird, sollte auf der Detailseite dargestellt werden.

A7: Es sollte auch möglich sein, Datensätze, die aus Teilmengen bestehen, auf dem Datenmarktplatz darzustellen, z.B. als *Kollektionen*. Dies ist der Fall, wenn ein Datensatz z.B. mehrere Tabellen oder mehrere Bilder enthält.

A8: Die Detailseite sollte über *Datenqualitätsmetriken* verfügen.

A9: Wenn es nicht gegen das Urheberrecht oder das Gesetz zum Schutz personenbezogener Daten[16] verstößt, sollte eine *Vorschau der Daten* möglich sein.

Wie aus [21, 42–44] ersichtlich, werden IoT-Daten häufig im Unternehmen gesammelt und gespeichert und können dementsprechend auf dem Datenmarktplatz provisioniert werden. Bei der Auswahl der IoT-Daten sind auch Informationen über die Geräte, die die Daten liefern, von Bedeutung, da die Messergebnisse direkt von den technischen Daten des Sensors abhängen. Um die Suche nach Sensordaten am Datenmarktplatz zu erleichtern, wird die folgende Anforderung eingeführt:

A10: Wenn es sich bei den Daten um Sensordaten handelt, sollten dem Benutzer auch *IoT-spezifische Metadaten* zur Verfügung stehen.

A11: Als Option für Crowdsourcing ist eine Funktion zum *Bewerten und Kommentieren* von Datensätzen wünschenswert.

Zu den *nicht-funktionalen* Anforderungen gehören die folgenden Eigenschaften und Merkmale der Darstellung von Metadatentemplates auf dem internen Datenmarktplatz:

A12: *Benutzerfreundlichkeit* und *Einheitliche Darstellung*. Der Benutzer sollte nachvollziehen können, wo und welche Metadaten er zu den Datensätzen finden kann. Die Metadaten für verschiedene Datensätze sollten sich an derselben Stelle/Registerkarte befinden und im selben Stil gehalten werden.

A13: *Erweiterbarkeit*. Es sollte möglich sein, die Detailseite um zusätzliche Metadaten zu erweitern.

[16] Bundesdatenschutzgesetz: https://www.buzer.de/s1.htm?g=BDSG+2018&f=1

5 Verwandte Arbeiten

Wie in den vorangegangenen Kapiteln erläutert, besteht das Hauptziel der Erfassung von Metadaten für den Datenmarktplatz darin, den Nutzern möglichst vollständige Informationen über die Datensätze zu liefern. Um mögliche Metadaten für den internen Datenmarktplatz zu sammeln und Templates dafür zu entwickeln, ist es notwendig zu verstehen, wie solche Templates generiert werden und aussehen können. Aus diesem Grund ist es sinnvoll zu betrachten, wie ähnliche Aufgaben in der Literatur behandelt werden.

So schlagen Gebru et al. [45] vor, spezielle Tabellen für Datensätze zum maschinellen Lernen zu verwenden, die *Datasheets* heißen. Solche Tabellen sollten Informationen über den Prozess der Datenerhebung, Datenstruktur und den jeweiligen Einsatzbereich enthalten. Für die Erstellung von Datasheets wird in dem Artikel empfohlen, einen Satz von 57 Fragen zu verwenden, die in Abschnitte eingeteilt sind. Diese Abschnitte entsprechen den wichtigsten Phasen des Lebenszyklus der Daten für das maschinelle Lernen. Dazu gehören Motivation, Struktur, Erhebungsprozess, Vorverarbeitung, Nutzung, Verbreitung und Pflege der Daten. Die drei Fragen für den Abschnitt „Nutzung“ lauten z.B.: „Wurde der Datensatz bereits verwendet und wenn ja, für welche Aufgaben?“, „Für welche anderen Aufgaben kann der Datensatz verwendet werden?“, „Für welche Aufgaben sollte der Datensatz nicht verwendet werden?“. Fragen, die auf den Datensatz nicht anwendbar sind, können übersprungen werden. Es wird jedoch angeraten, dass so viele Fragen wie möglich beantwortet werden. In [45] wird darauf hingewiesen, dass die Verwendung solcher Datasheets einerseits die Anbieter von Datensätzen dazu anregt, bei der Sammlung, Verbreitung und Pflege der Daten bewusster vorzugehen, und andererseits das Vertrauen der Konsumenten in die Daten erhöht, das Risiko des Datenmissbrauchs verringert sowie eine bessere Auswahl von Datensätzen je nach anstehenden Aufgaben ermöglicht. Obwohl die in [45] vorgeschlagene Methode für Daten des maschinellen Lernens entwickelt wurde, ist sie auch für diese Arbeit relevant. Es ist nicht möglich, alle in dem Artikel vorgestellten Fragen auf alle Datensätze auf dem Datenmarktplatz anzuwenden, da einige von ihnen spezifisch und nur für Daten des maschinellen Lernens geeignet sind. Interessant ist jedoch, wie die Fragen in Abschnitte unterteilt sind, was dazu beiträgt, die Informationen über die Datensätze strukturiert darzustellen und auch für die Erstellung von Templates verwendet werden kann.

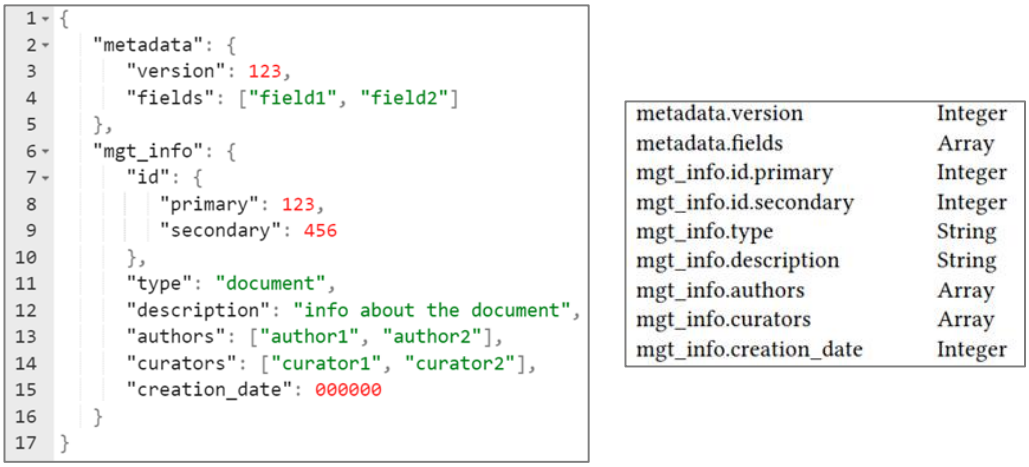
Ein ähnliches Konzept wird auch von Mitchell et al. [46] beschrieben, wobei die Verwendung sogenannter *Model Cards* zur Beschreibung von Modellen des maschinellen Lernens vorgeschlagen wird, um den Kontext der Modellverwendung zu verdeutlichen und Missbrauch zu verhindern. Model Cards sind Dokumente mit Informationen über Modelle des maschinellen Lernens, die Einzelheiten über das Modell (z.B. Version, Typ, Lizenz) und den vorgesehenen Einsatz, Informationen über die Performance und Effizienz des Modells sowie Vorbehalte und Empfehlungen zur Verwendung des Modells umfassen. Es ist auch möglich, Model Cards zu verwenden, um Modelle für verschiedene Personengruppen und Bedingungen zu vergleichen, für die die Modelle bestimmt sind (z.B. Rasse, Alter, geografische Lage). Wie im Falle der Datasheets aus [45] sind die Model Cards nur für den Einsatz im maschinellen Lernen gedacht, was ihre Verwendung für alle Datensätze auf dem Datenmarktplatz nicht zulässt.

Richards et al. [47] diskutieren eine Methodik zur Dokumentation von Modellen der künstlichen Intelligenz, *FactSheets* genannt. Einer der Schritte in dieser Methodik besteht in der Entwicklung von Templates für das Modell. Ein solches Template enthält Fragen, die sich auf die Beschreibung des Modells beziehen. Dies sind z.B. Informationen darüber, wie das Modell trainiert wurde, welche Datentypen es verwendet, welche Eingabedaten akzeptiert und welche Ausgabedaten produziert werden sowie wie es getestet wurde. In [47] wird angemerkt, dass all diese Informationen es ermöglichen, das Modell besser zu verstehen und zu entscheiden, ob es für den Anwendungsfall des Entwicklers geeignet ist. In diesem Fall beschränken sich die Fragen zu den Templates wiederum auf einen Anwendungsbereich, nämlich die künstliche Intelligenz, aber es ist zu erkennen, dass auf der Grundlage der Quellen [45–47] eine Strategie für die Erfassung von Metadaten skizziert werden kann, die darin besteht, Fragen (Metadatenfelder) für Datensätze zu finden, die von ihrem Ersteller beantwortet (ausgefüllt) werden können. Diese Strategie kann auch zur Erstellung von Metadatentemplates verwendet werden.

Abbildung 7. Einige Metadaten, die in den von Huawei verwendeten Data Maps für Daten vom Typ *Business Asset* angezeigt werden [48]

Ma und Du [48] präsentieren die von Huawei entwickelten *Data Maps* (*DMAP*), die das Auffinden und Verstehen von Daten im Unternehmen erleichtern sollen. Bei diesen Data Maps handelt es sich im Wesentlichen um Sätze von Metadaten, die die Daten beschreiben. Dieses Konzept ist der Bereitstellung von Metadaten auf dem Datenmarktplatz sehr ähnlich. Außerdem wurden die Data Maps für unterschiedliche Datentypen und unterschiedliche Nutzer konzipiert. Vier Benutzergruppen werden von Data Maps bedient: Business- und Datenanalysten sowie Datenverwalter und IT-Entwickler. Das bedeutet, dass für jede dieser Gruppen die Metadaten der Data Maps auf ihre Interessen und Aufgaben im Unternehmen zugeschnitten werden. Darüber hinaus können die Daten auf der Grundlage der Metadaten der Data Maps gesucht, sortiert, gefiltert und klassifiziert werden. In Abbildung 7 sind einige der Metadaten aufgelistet, die den Nutzern für die Datensätze vom Typ *Business Asset* angezeigt werden. Dabei handelt es sich um verschiedene Metadaten, die z.B. Informationen über den Asset-Typ, -Quelle (*Data Source*), -Verantwortlichen (*Data steward*) und -Vertraulichkeitsstufe (*Confidentiality level*) enthalten. *Sample data* sind Beispieldaten, die es den Nutzern ermöglichen, sich mit den Daten vertraut zu machen, d.h. zu verstehen, wie diese

aufgebaut sind. Es sei darauf hingewiesen, dass in [48] nur grundlegende Metadaten bei der Beschreibung von Data Maps dargestellt werden, z.B. fehlen hier die Datenqualität und Lineage, die auch für die Mitarbeiter des Unternehmens von Interesse sein könnten. Die vorgestellten Metadaten sind jedoch auch für den internen Datenmarktplatz relevant, und einige von ihnen können bei der Erstellung von Templates verwendet werden. Z.B. solche Attribute, wie der Daten Verantwortlicher, der Datenspeicherort und die Vertraulichkeitsstufe. Es kann auch angebracht sein, die Beispieldaten den Nutzern zur Verfügung zu stellen. Die Aufteilung der Data Maps nach Nutzertypen wäre jedoch für den Datenmarktplatz wenig sinnvoll. Wenn z.B. ein Business-Analyst keinen Zugang zu den Metadaten hat, der z.B. einem Datenanalysten möglich ist, würde dies der Motivation des Datenmarktplatzes widersprechen, nämlich der Datendemokratisierung, bei der Informationen über die im Unternehmen vorhandenen Daten für möglichst viele Mitarbeiter einsehbar sein sollen. Es sei darauf hingewiesen, dass es im Falle des Datenmarktplatzes nicht darum geht, dass jeder Nutzer Zugang zu allen Daten des Unternehmens hat, sondern nur darum, dass er die Daten finden, Informationen über sie einsehen und nur dann Zugang beantragen kann, wenn sie von Interesse sind.



```
{
  "metadata": {
    "version": 123,
    "fields": ["field1", "field2"]
  },
  "mgt_info": {
    "id": {
      "primary": 123,
      "secondary": 456
    },
    "type": "document",
    "description": "info about the document",
    "authors": ["author1", "author2"],
    "curators": ["curator1", "curator2"],
    "creation_date": 000000
  }
}
```

metadata.version	Integer
metadata.fields	Array
mgt_info.id.primary	Integer
mgt_info.id.secondary	Integer
mgt_info.type	String
mgt_info.description	String
mgt_info.authors	Array
mgt_info.curators	Array
mgt_info.creation_date	Integer

Abbildung 8. JSON-Metadatenbeschreibung und JSON-Felder [49]

Drakopoulos et al. [49] stellen eine Metadatenstruktur zur Beschreibung verschiedener Kulturgüter im JSON-Format vor. Abbildung 8 zeigt ein Beispiel für Metadatenfelder im JSON-Format auf der linken Seite und eine Beschreibung dieser Feldtypen auf der rechten Seite. Wie zu sehen ist, umfasst der Metadatensatz die Version und eine Reihe von Feldern sowie Informationen über die Ersteller des Kulturguts, das Erstellungsdatum, den Typ und die Beschreibung. Auch hier ist klar, dass das Schema nur auf Kulturgüter anwendbar ist. Aber das Prinzip seines Aufbaus und die Verwendung des JSON-Formats zur Beschreibung von Metadaten ist für diese Arbeit von Belang.

Es gibt auch einige Projekte, die Metadatentemplates in ihre Produkte integrieren, wie z.B. *Egeria*[17]. Hier wird angeboten, bei der Eingabe von Informationen über eine neue Ressource in den Katalog Templates zu verwenden. Jede dieser Templates enthält eindeutige Elemente für das digitale Asset, wie z.B. den „qualifiedName“ sowie detailliertere Informationen über

[17] Templated cataloging: https://egeria-project.org/features/templated-cataloguing/overview/

das Asset. Solche Templates können für verschiedene Typen von digitalen Ressourcen entwickelt werden, deren Klassifizierung sie ebenfalls unterstützen. Abbildung 9 zeigt z.B. den von Coco Pharmaceuticals verwendeten Templatesatz, der Templates für verschiedene Datentypen enthält, die in der Abbildung durch unterschiedliche Formen dargestellt sind. Wie aus der Abbildung hervorgeht, kann es sich bei diesen Datentypen bei Coco Pharmaceuticals um klinische Patientendaten, verschiedene Datensätze oder APIs handeln. Mit diesem Ansatz können Metadaten aufgegliedert werden, was die Benutzerfreundlichkeit von Templates erhöht und auch für die Entwicklung von Templates für den Datenmarktplatz von praktischem Interesse ist.

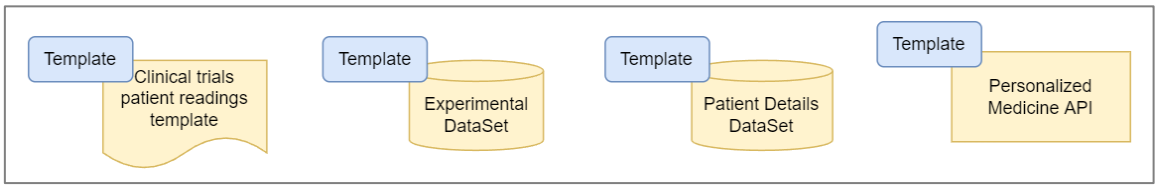


Abbildung 9. Set von Templates für die Katalogisierung von digitalen Ressourcen, die von Egeria für Coco Pharmaceuticals bereitgestellt werden[18]

Somit lässt sich aus den vorgestellten Informationen schließen, dass Metadatentemplates denjenigen, die sie bereitstellen, die Beschreibung der Daten erleichtern und aufgrund der darin enthaltenen Informationen zu einem besseren Verständnis der Daten beitragen können. Darüber hinaus ist festzustellen, dass bei der Verwendung von Templates eine Gruppierung nach Datentyp oder Anwendungsbereich erfolgen kann. Diese Gruppierung kann auch einen Vorteil für den Datenmarktplatz haben, wenn Metadatentemplates verwendet werden. Interessant ist auch die Verwendung des JSON-Formats für die Beschreibung der Metadaten. Alle diese Aspekte werden bei der Entwicklung der Templates berücksichtigt. Metadatentemplates sind das Thema des nächsten Kapitels und in Kapitel 10 wird die Implementierung eines Datenmarktplatz-Prototyps unter Verwendung der JSON-Metadatendarstellung vorgestellt.

[18] Das Bild wurde der Webseite https://egeria-project.org/features/templated-cataloguing/ entnommen

6 Metadatentemplates

In Abschnitt 2.2 wurden die grundlegenden Datentypen, die im Unternehmen vorhanden sein können, berücksichtigt, d.h. die einfachste Unterteilung bei der Entwicklung von Templates für den internen Datenmarktplatz könnte in Templates für strukturierte, semi-strukturierte und unstrukturierte Daten erfolgen. Ein Blick auf die Anforderungen an den Datenmarktplatz aus Kapitel 4 zeigt jedoch, dass es auch sinnvoll ist, Metadaten zusätzlich auf andere Weise zu gruppieren, z.B. durch die Trennung von Metadaten für Datenspeicherung, Nutzungsbedingungen, Provisionierungsoptionen, Lineage und Qualität. Daher ist es notwendig, verschiedene Metadaten zu erfassen, die zusätzliche Informationen über die Datensätze liefern, um die in Kapitel 4 beschriebenen Anforderungen zu erfüllen. Die Metadaten werden es ermöglichen, dass die Nutzer des Datenmarktplatzes die Datensätze leichter finden sowie einordnen können, ob die Daten ihren Ansprüchen genügen. Des Weiteren können sie ebenfalls verstehen, wofür sie die Daten verwenden dürfen. Dafür soll der Datenmarktplatz dynamisch unterschiedliche Metadaten zu verschiedenen Datensätzen anzeigen können. Um eine dynamische Anzeige zu gewährleisten und alle gesammelten Metadaten zu systematisieren und zu strukturieren, wurden 69 Templates entwickelt, die die Metadaten für einen internen Datenmarktplatz enthalten. Alle Zusammenhänge zwischen den Templates sind in Abbildung 10 schematisch dargestellt und die Templates selbst befinden sich im Anhang 1 der Arbeit. Die in rosa hervorgehobenen Templates werden in diesem Kapitel näher vorgestellt.

Zur Erfassung von Metadaten und zur Erstellung von Templates wurden folgende Hilfsmittel verwendet:

- Datenmarktplätze Advaneo und AWS Datamarketplace;
- Metadatenstandards (Dublin Core, EXIF, MPEG-7);
- Tools und Websites zur Extraktion von Metadaten;
- Informationen aus der Literatur.

Je nach Datentyp, für den die Templates entwickelt wurden, werden in den entsprechenden Abschnitten Namen von Tools und Webseiten sowie Referenzen auf die verwendete Literatur angegeben.

6.1 Framework von Metadatentemplates

Um das Framework besser zu verstehen, sollten die in Abbildung 10 verwendeten Bezeichnungen der einzelnen Elemente genauer betrachtet werden. Schwarzer Text bedeutet hier entweder einen allgemeinen Begriff, der hilft, die Templates zu strukturieren, wie z.B. *structured* oder ein Attributname, dessen Werte andere Attribute sind, die durch Templates dargestellt werden, wie z.B. das Attribut *data_access_method*, das durch die Templates <*api*>, <*download*> oder <*access*> dargestellt werden kann. Blauer Text in dreieckigen Klammern bedeutet, dass für dieses Element ein Metadatentemplate entwickelt wurde. Pfeile mit schwarzer Spitze zeigen Subtypen oder Varianten von Begriffen an, und die mit weißer Spitze weisen auf die Vererbung hin.

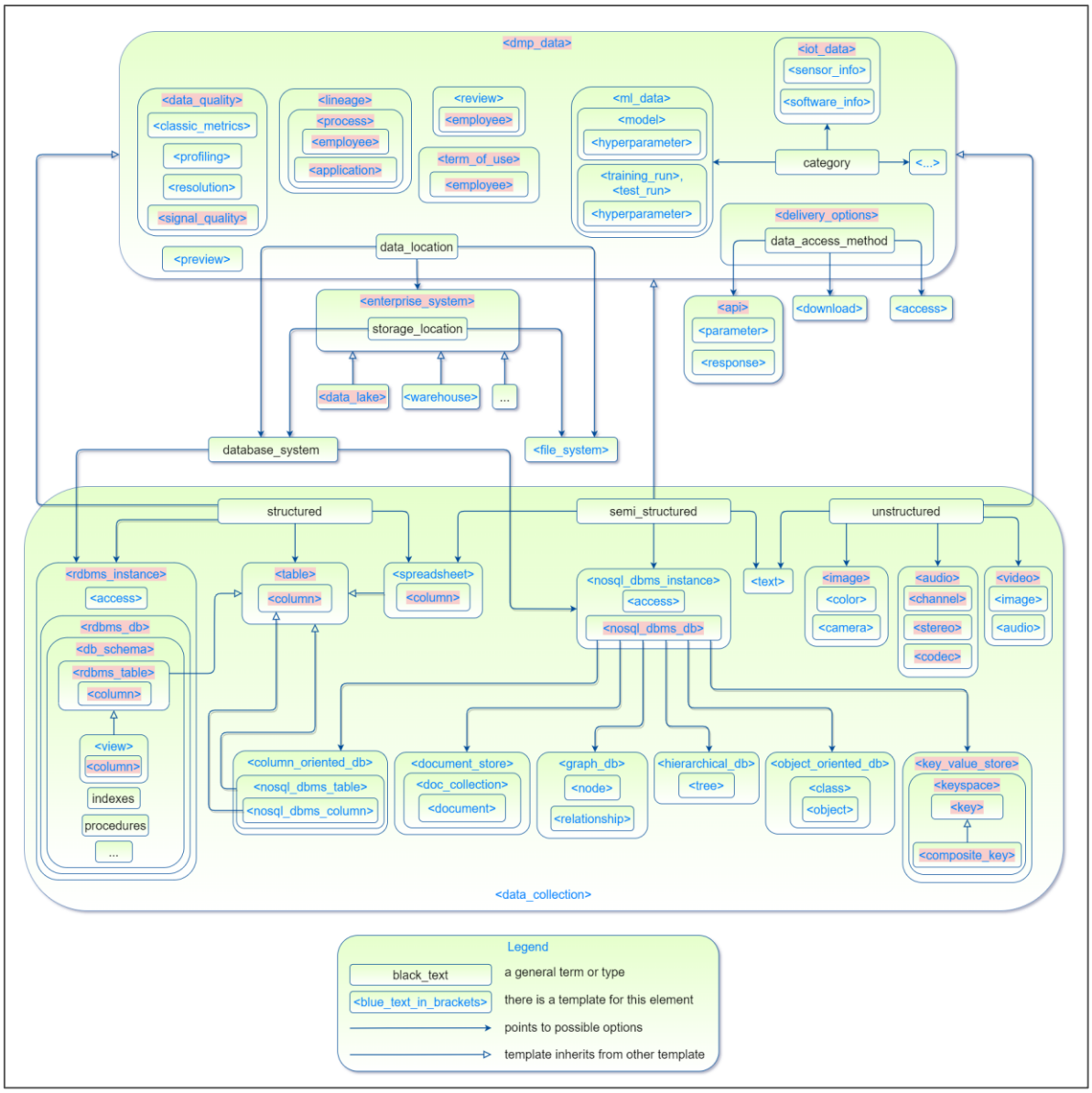


Abbildung 10. Framework von Metadatentemplates

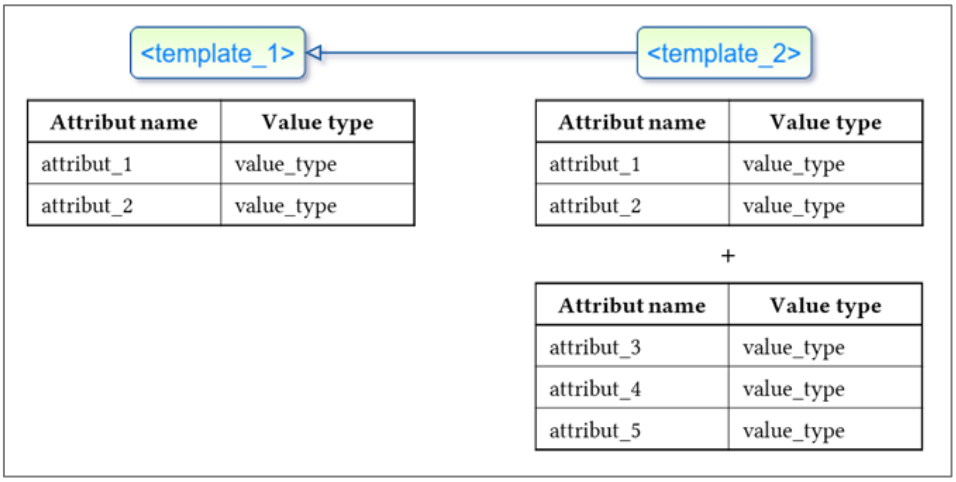


Abbildung 11. Vererbung von Metadaten

Um die Datentypen in den Templates und einige der Bezeichnungen zu erläutern, wurde auch das in Abbildung 12 dargestellte Beispiel erstellt, das drei vereinfachte Templates für *<database>*, *<table>* und *<column>* enthält. So werden mit einem Stern die obligatorischen Metadaten markiert, wie im Fall des Attributs *database* in Template *<table>*. In den Klammern können auch zulässige Werte stehen, wenn es Beschränkungen für die möglichen Werte gibt, wie beim Attribut *type* des Templates *<database>*. Bei den Attributwerttypen kann es sich um einfache Werte wie *string*, *number*, *date* oder einzelne Templates handeln. D.h., einige der Templates werden dann Teil anderer Templates sein. In Abbildung 12 ist dies z.B. das Template *<column>*, das Teil des Templates *<table>* ist. Wenn es mehrere solche Template-Objekte gibt, handelt es sich um ein Array von Templates, im angeführten Beispiel ist dies *array<column>*. Im Gegensatz zur Vererbung verfügen solche eingebauten Templates nur über ihre eigenen Metadaten und können nur innerhalb eines anderen Templates existieren, von dem sie ein Teil sind. Das bedeutet, dass die Spalte (*column*) in diesem Beispiel nicht als eigenständiges Produkt angeboten werden kann.

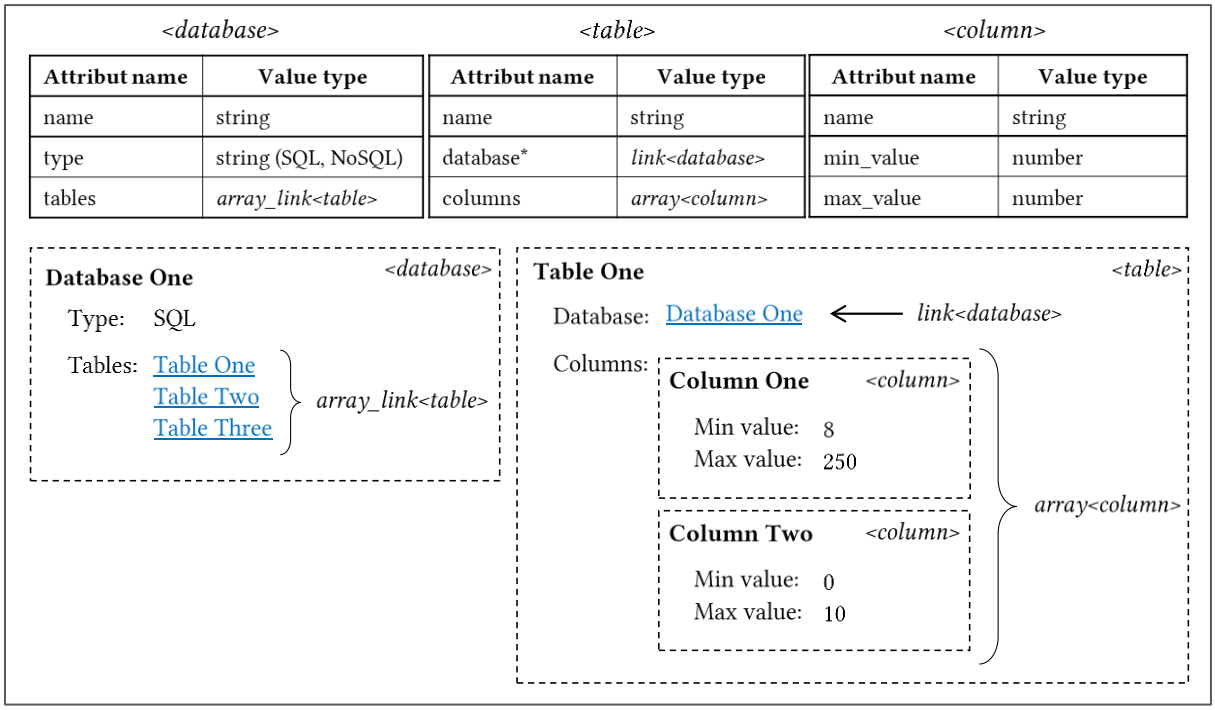


Abbildung 12. Beispiel für Templates und ihre Repräsentation

Zusätzlich zu den einfachen Typen und Templates gibt es auch Links zu Daten, die durch Templates dargestellt werden, wie z.B. *link<database>*, d.h. wenn der Link *Database One* angeklickt wird, erscheint eine detailliertere Beschreibung der Datenbank, deren Metadaten durch das Template *<database>* abgebildet werden. Wenn mehrere solche Links zu mehreren Templates-Objekten vorhanden sind, dann wird dieser Typ als *array_link* bezeichnet, wie z.B. *array_link<table>* in Abbildung 12.

6.2 Prozess der Entstehung von separaten Templates

Um die Templates zu erstellen, wurden zunächst alle Daten des Unternehmens wie in Abbildung 10 dargestellt strukturiert, wobei die Anforderungen an den Datenmarktplatz und die in Kapitel 3 entwickelten Szenarien berücksichtigt wurden. Für alle Elemente des Frame-

works wurden alle möglichen Metadaten unter Verwendung der am Anfang dieses Kapitels genannten Hilfsmittel erfasst. In zwei Fällen wurden eigene Templates angelegt. Beide Fälle sind in Abbildung 13 dargestellt. Der erste Fall ist, wenn der Metadatensatz Metadaten aus einem anderen Template enthält, dann kann er diese Metadaten, wie im vorherigen Abschnitt beschrieben, erben. Die Abbildung zeigt ein Beispiel für einen vereinfachten Metadatensatz für die Daten von Typ *image*. Wie ersichtlich, sind die ersten sechs Elemente identisch mit den Elementen des Templates *<dataset>*, so dass diese Elemente geerbt werden können. Das Template enthält jedoch auch weitere Metadaten, die sich von den Metadaten des übergeordneten Templates unterscheiden, sodass daraus ein separates Template gebildet werden kann. Der zweite Fall liegt vor, wenn der Metadatensatz zwei oder mehr Elemente enthält, die sich auf dasselbe Attribut beziehen. In der Abbildung 13 ist zu erkennen, dass das *<image>*-Template drei Attribute für das Element *camera* sowie drei für das Element *color* hat. In diesem Fall können *<camera>* und *<color>* zu separaten Templates innerhalb der *<image>*-Template zusammengefasst werden.

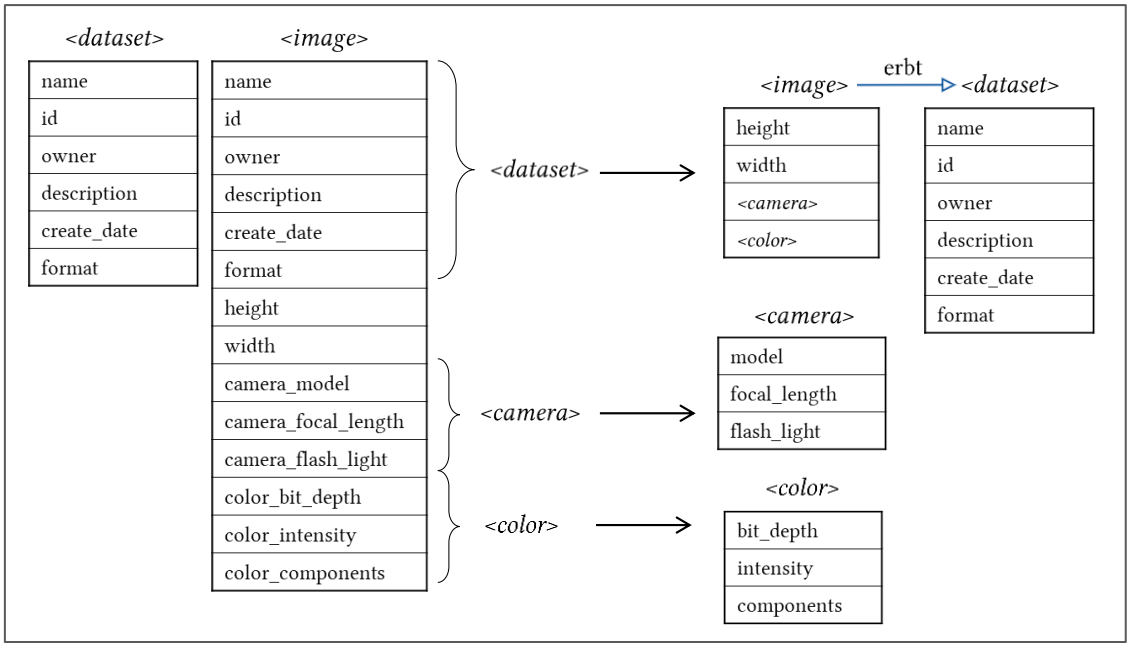


Abbildung 13. Prozess der Entstehung von separaten Templates

Anhand der Templates lässt sich der Prozess der Datensatzauswahl durch den Nutzer auf dem Datenmarktplatz nachvollziehen, wozu die Metadaten beitragen sollen. Um das Konzept der Templates genauer zu erklären, ist es sinnvoll, sowohl das allgemeine *<dmp_data>*-Template, zu dem auch weitere Templates gehören, als auch Templates für strukturierte, semi-strukturierte und unstrukturierte Daten zu betrachten.

6.3 Allgemeines Template

Das allgemeine Template *<dmp_data>* (Tabelle 2, Template 1), dessen Teil des Frameworks in Abbildung 14 dargestellt ist, enthält die wichtigsten obligatorischen Metadaten zu jedem Datensatz, wie z.B. *name*, *guid*, *description*, *type*, *dmp_type* und *location* (die Bedeutungen dieser Attribute werden im Folgenden näher erklärt). Mit diesen Metadaten kann der Datenmarktplatznutzer eine Vorauswahl treffen und mehrere Datensätze wählen, die thematisch und vom Typ für seinen Anwendungsfall passen. Das Attribut *type* bezieht sich

hier allgemein auf einen Typ von Daten, wie z.B. eine Tabelle, einen Text, ein Bild oder einen Audio-Datensatz. Bei Szenario 1 möchte der Datenanalyst z.B. Bilder von defekten Holzleisten finden, wofür er den Begriff „wood planks“ eingeben und den Datentyp „image“ auswählen kann. Da er sich auch für Sensordatentabellen interessiert, was in Anforderung *S1A1* festgelegt ist, könnte der Datentyp auch eine Tabelle sein. Und der Business-Analyst aus Szenario 2, der an customer support interessiert ist, könnte Text- und Audiodateien auf dem Datenmarktplatz auswählen.

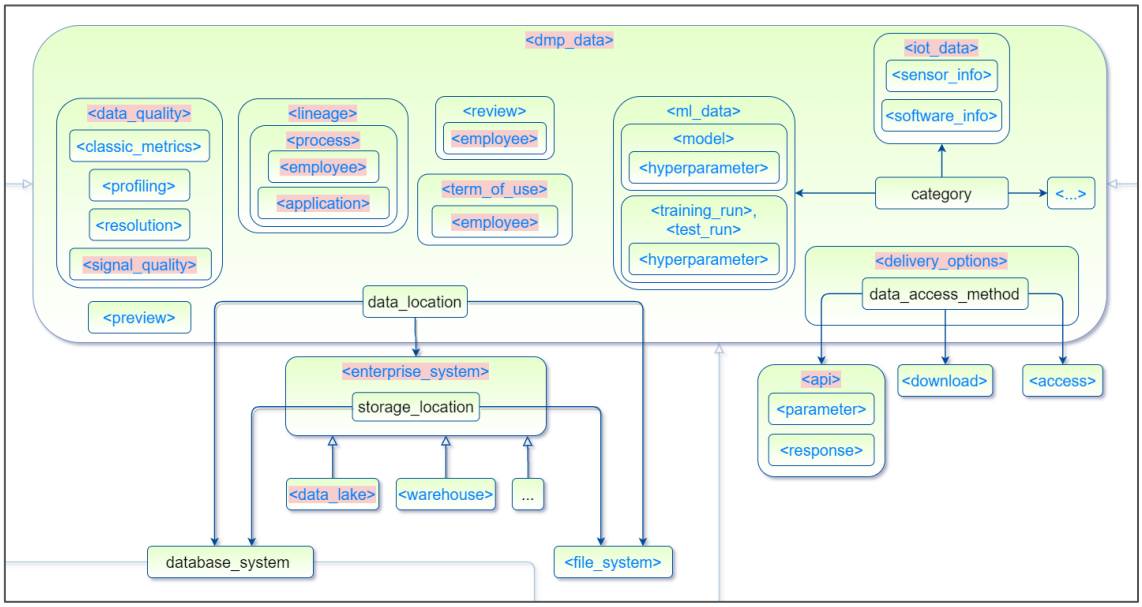


Abbildung 14. Teil des Frameworks, der das allgemeine *<dmp_data>*-Template darstellt

Der *dmp_type*[19] ist spezifisch und kann die Werte *asset* und *product* annehmen. *Asset* wird verwendet, wenn ein Datensatz im Unternehmen in einem Datenkatalog verzeichnet ist, aber noch nicht auf dem Datenmarktplatz registriert wurde. Ein Nutzer des Datenmarktplatzes kann eine Anfrage an den Data Owner senden, aber in diesem Fall muss der Owner die Daten zunächst als Produkt registrieren, bevor der Zugang gewährt werden kann. *Product* ist ein Datensatz, der bereits auf dem Datenmarktplatz registriert wurde und zusätzlich administrative Metadaten wie Nutzungsbedingungen und Provisionierungsoptionen enthält, d.h. diese Attribute sind nur für das Produkt obligatorisch (ausgenommen ist das Attribut *owner*, das im Falle eines *assets* ebenfalls erforderlich ist).

Alle anderen Attribute des *<dmp_data>*-Templates, z.B. *language*, *create_date*, *modify_date*, *format*, *size*, *coverage* oder *version* sind optional und können je nach Datentyp erscheinen. Von Interesse sind hier jedoch die Attribute, für die separate Templates entwickelt wurden. Diese sind z.B. *category*, *lineage*, *quality*, *preview* und *reviews*. Wie bereits erwähnt, sind auch separate Templates für obligatorische Attribute wie *location*, *term_of_use* und *delivery_options* vorhanden.

Zusätzlich zu den Spalten mit dem Attributnamen und dem Wertetyp enthalten sowohl das allgemeine Template als auch das Template für die Nutzungsbedingungen (Template 2) eine

[19] *dmp_type*-Attribut und seine Werte wurden im Rahmen eines Studentenprojekts zur Entwicklung eines Datenmarktplatz-Prototyps definiert

zusätzliche Spalte mit der Abbildung des jeweiligen Attributs auf den Dublin Core Standard, welcher in Abschnitt 2.6.3 näher erläutert wurde. Diese Abbildung kann in Zukunft die Integration von Metadaten aus Tools oder Systemen, die den Dublin Core Standard unterstützen, erleichtern. Als nächstes sind die Attribute eines allgemeinen *<dmp_data>*-Templates, die ebenfalls Metadatentemplates sind, genauer zu betrachten.

Tabelle 2. *<dmp_data>*-Template

Attribut Name	Dublin Core Attribut	Value Type
name*	dc: title	string
guid*	dc: identifier	string
description*	dc: description	string
type*	dc: type	string
dmp_type*		string (asset, product)
location*		*<rdbms_db>*, *<nosql_dbms_db>*, *<filesystem>*, *<enterprise_system>*, string
term_of_use* (for product)	dc: rights	*<term_of_use>*
delivery_options* (for product)		*<delivery_options>*
language	dc: language	string
create_date	dc: date.created	date, string
modify_date	dc: date.modified	date, string
mime_type		string
format	dc: format	string
size	dc: format.extent	number
coverage	dc: coverage	string
category	dc: subject	*<ml_data>*, *<iot_data>*, string
keywords		array<string>
version		number, string
lineage		*<lineage>*
quality		*<quality>*
preview		*<preview>*
reviews		*array<review>*
metadata_tool		string, link

6.3.1 Templates für Nutzungsbedingungen

Sobald der Benutzer einen oder mehrere Datensätze, die er näher untersuchen möchte, gefunden und ausgewählt hat, kann er damit beginnen, die Informationen über die Nutzungsbedingungen zu erkunden. Dafür wird das Template *<term_of_use>* bereitgestellt (Template 2). Die obligatorischen Elemente dieses Templates sind die Attribute *owner*, *security_class*, *contain_personal_data* und *license*. Diese Attribute sind zudem in Tabelle 3 zusammengefasst.

Neben der Umsetzung der EU-Richtlinie über das Urheberrecht [51] hat die Verfügbarkeit von Informationen über den Data Owner auf dem Datenmarktplatz auch administrative und

technische Bedeutung, da nur der Data Owner Zugang zu Daten gewähren kann. Wenn der Typ des Wertes des Attributes *owner* beachtet wird, lässt sich feststellen, dass es sich neben dem Typ *string* auch um den Typ *<employee>* handeln könnte, der ein Template für die Bereitstellung von Informationen über die Mitarbeiter des Unternehmens ist (Tabelle 4, Template 3). Diese Informationen müssen den Vor- und Nachnamen des Data Owners sowie seine E-Mail-Adresse enthalten, damit die Nutzer ihn kontaktieren können, wenn sie Fragen zum Datensatz haben. Andere Metadaten in dem Template sind optional und ergänzen die Angaben über den Data Owner. Um wieder auf Szenario 1 zurückzukommen: Anforderung *S1A8* lautet, dass der Datenanalyst wissen möchte, in welcher Abteilung der Data Owner arbeitet, da diese Informationen bei der Auswahl eines Datensatzes nützlich sein können.

Tabelle 3. *<term_of_use>*-Template

Attribut Name	Dublin Core Attribut	Value Type
owner*	dc: creator	*<employee>*, string
security_class*	dc: rights.accessRights	number (0, 1, 2, 3, 4)
contain_personal_data*		boolean
license*	dc: rights.license	*<license>*, string
permitted_usage		string
conditions_of_use		string

Tabelle 4. *<employee>*-Template

Attribut name	Value type
first_name*	string
last_name*	string
department	string
position	string
phone_number	number
email*	string

Auch das Attribut *security_class*[20] hat eine administrative und technische Bedeutung für den Datenmarktplatz. Je nach Security Class kann z.B. ein direkter Zugang zu öffentlichen Daten gewährt werden oder ein Freigabeprozess für den Zugriff erforderlich sein, falls es sich um vertrauenswürdigere Daten handelt. Die Security Class kann fünf verschiedene Werte annehmen und repräsentiert die Vertraulichkeitsstufe des Datensatzes. Die folgenden Werte sind für *security_class* möglich: *0* – keine Klasse zugewiesen; *1* – öffentlich; *2* – intern; *3* – vertraulich; *4* – streng vertraulich.

Das Attribut *contain_personal_data* gibt an, ob der Datensatz personenbezogene Daten enthält. Laut der Datenschutz-Grundverordnung[21] (DSGVO) sind *personenbezogene Daten* Informationen, die eine Person eindeutig identifizieren können, wie z.B. Name, Adresse oder Passnummer. Die Verwendung personenbezogener Daten kann eingeschränkt sein. Daher ist es wichtig, dass es möglich ist, festzustellen, welche Datensätze personenbezogen sind, und gegebenenfalls die personenbezogenen Daten zu verschlüsseln oder die entsprechende

[20] *security_class*-Attribut und seine Werte wurden im Rahmen eines Studentenprojekts zur Entwicklung eines Datenmarktplatz-Prototyps definiert
[21] DSGVO: https://dsgvo-gesetz.de

Erlaubnis für ihre Verwendung einzuholen [28]. All dies kann dazu beitragen, Probleme mit der Sicherheit personenbezogener Daten zu umgehen. Um solche Probleme zu vermeiden, besteht z.B. die Anforderung des Business-Analysten in Szenario 2 darin, alle personenbezogenen Daten zu anonymisieren (*S2A2*).

Das Attribut *license* liefert zusätzliche Informationen darüber, wie die Daten zu behandeln sind, ob sie geändert oder für kommerzielle Zwecke verwendet werden können, ob es erlaubt ist, sie unter anderen Lizenzen wieder anzustellen sowie ob der Data Owner bei der nächsten Verwendung immer angegeben werden muss [52]. Für den Datenmarktplatz kann die Verwendung von Creative-Commons-Lizenzen[22] vorgeschlagen werden. Darüber hinaus ist es sinnvoll, den Benutzern die Möglichkeit zu geben, ihre eigenen Lizenzen hinzuzufügen. Die Lizenztypen können bei Bedarf erweitert werden. In Rahmen dieser Arbeit wurde ein separates <*license*>-Template für das *license*-Attribut erstellt, das Parameter wie *name*, *description* und *link* enthält (Template 4). Anforderung *S1A9* ist ein Beispiel für einen Nutzer, der auf dem Datenmarktplatz nach Daten sucht und nur an offen lizenzierten Daten interessiert ist, z.B. um sie ohne Einschränkungen ändern und wieder in den Datenmarktplatz einstellen zu können, aber unter seiner eigenen Lizenz.

Die Attribute *permitted_usage* und *conditions_of_use* sind optional und dienen dazu, weitere Details über den Verwendungszweck und besondere, nicht lizenzierte Bedingungen für die Nutzung des Datensatzes hinzuzufügen. Der Data Owner kann in den *conditions_of_use* z.B. festlegen, dass die Daten den Mitarbeitern der Abteilungen Buchhaltung und Finanzen zur Verfügung gestellt werden können. Der *permitted_usage* kann z.B. in der Durchführung von Gewinnberechnungen für das laufende Jahr oder einer Kostenanalyse bestehen.

6.3.2 Templates für Datenspeicherung

Sobald der Datenmarktplatznutzer die Nutzungsbedingungen zur Kenntnis genommen hat, kann er sich darüber informieren, wo sich die angebotenen Daten im Unternehmen befinden. Dies kann wichtig sein, wenn der Benutzer z.B. möchte, dass die Daten intern im Unternehmen gespeichert werden, wie in Anforderung *S1A10*. Je nach Datenstruktur wird der auf dem Datenmarktplatz angebotene Datensatz in einem der Datenspeicher im Unternehmen abgelegt. Wie in Abschnitt 2.4.1 beschrieben, kann es sich dabei um eine relationale oder nicht-relationale Datenbank sowie um ein Dateisystem handeln. Für jede Art der Speicherung wurde ein Metadatentemplate entwickelt. Templates für relationale und nicht-relationale Datenbanken werden in den Abschnitten 6.3 und 6.4 behandelt, die sich mit strukturierten bzw. semi-strukturierten Daten befassen.

Im Falle von Dateisystemen bestehen die Metadaten in der Regel aus Informationen über den Pfad zu den Daten und dem Datum, an dem die Daten dem System hinzugefügt wurden, sowie über den Zeitpunkt der Erstellung oder Änderung der Daten [53]. Da die letzten beiden Attribute bereits im allgemeinen Template enthalten sind, müssen sie nicht erneut berücksichtigt werden, weil sie ohnehin vererbt werden. Daher wurde beschlossen, dass das Template <*file_system*> nur zwei Attribute enthält, nämlich *file_path* und *timestamp*. Dieses Template kann in Zukunft bei Bedarf durch Metadaten ergänzt werden (Template 5). Es sollte

[22] About The Licenses: http://creativecommons.org/licenses

auch daran erinnert werden, dass Datenbanken und Dateisysteme innerhalb von Unternehmenssystemen existieren können, wie bereits in Abschnitt 2.4.2 erwähnt. Zu diesen Systemen gehören z.B. der Data Lake und das Data Warehouse. Die Templates für diese Systeme sollen im Folgenden näher betrachtet werden.

Templates für Unternehmenssysteme

Das Template *<enterprise_system>* (Template 6) enthält Metadaten, die Informationen über den Typ des Unternehmenssystems, das unmittelbare Daten-Repository und darüber, wie und wann die Daten in das System importiert wurden, enthalten. Dieses Template ist zusätzlich in Tabelle 5 wiedergegeben.

Tabelle 5. *<enterprise_system>*-Template

Attribut name	Value type
system_type	*<warehouse>*, *<data_lake>*, string
storage_location	*<rdbms_db>*, *<nosql_dbms_db>*, *<filesystem>*, string
import_from	*<rdbms_db>, <nosql_dbms_db>, <filesystem>, string*
import_timestamp	date, string

Das Attribut *<system_type>* kann die Werte *<data_lake>*, *<warehouse>* oder *string* annehmen. Der Typ *string* wird benutzt, wenn es sich um ein Unternehmenssystem handelt, das in dieser Arbeit nicht behandelt wird. Die Attribute *storage_location* und *import_from* enthalten Informationen über den aktuellen Speicherort von Daten und über den Datenspeicher, aus dem die Daten in das System importiert wurden. Der Datenspeicherort kann, wie schon mehrfach erwähnt, eine Datenbank oder ein Dateisystem sein, für die, wie bereits bekannt, ebenfalls eigene Templates zur Verfügung stehen. Das Attribut *import_timestamp* gibt Auskunft über das Datum und die Uhrzeit des Imports.

Tabelle 6. *<data_lake>*-Template

Attribut name	Value type
data_lake_zone	string
is_in_other_zones	string

Es sollte auch beachtet werden, dass *<data_lake>* und *<warehouse>* (Templates 7, 8) ebenfalls separate Templates sind und von dem *<enterprise_system>*-Template erben, d.h. neben ihren eigenen spezifischen Metadaten enthält jedes von ihnen auch Metadaten des übergeordneten Templates.

Die spezifischen Metadaten des *<data_lake>*-Templates umfassen beispielsweise den Data Lake Zone sowie Informationen darüber, in welchen anderen Data Lake Zonen sich der Datensatz befindet (Tabelle 6, Template 7). Durch die Angabe des Data Lake Zones könnte der Datenverarbeitungsgrad bestimmt werden. So kann der Datenanalyst aus Szenario 1 erkennen, ob die Daten seine Anforderung erfüllen und bereits verarbeitet sind, z.B. dass Zeilen mit fehlenden Temperatur- oder Luftfeuchtigkeitswerten gelöscht wurden (Anforderung *S1A2*). Ein zusätzliches Attribut des *<warehouse>*-Templates ist z.B. die Logik der Datentransformation durch ETL-Tools im Data Warehouse (Template 8).

6.3.3 Templates für Daten-Provisionierungsoptionen

Nachdem der Nutzer die Informationen über den Datenspeicherort bekommen hat, möchte er eventuell wissen, wie er auf die Daten zugreifen kann. Wie wichtig der Datenzugangstyp für den Nutzer ist, kann davon abhängen, welche Programme er nutzt und wie er die Daten bearbeiten will. Es kann mehrere Optionen für die Datenbereitstellung geben und die Daten können regelmäßig aktualisiert, z.B. mit neuen Werten ergänzt, werden. Informationen über die Aktualisierung der Daten können für den Datenmarktplatznutzer wichtig sein, wenn er z.B. jede Woche oder jeden Monat neue Daten auswerten möchte. Das Attribut *delivery_options* des Templates <*dmp_data*> ist wiederum ein Template, das die Attribute *update_cycle* und *data_access_method* enthält (Template 9). Das Template <*delivery_options*> ist auch in Tabelle 7 zu finden.

Tabelle 7. <*delivery_options*>-Template

Attribut name	Value type
update_cycle	string
data_access_method	<*download*>, <*access*>, <*api*>, string

Ein Blick auf die Anforderungen *S1A10* und *S2A1* zeigt, dass sie sich auf den Typ des Datenzugriffs beziehen. Der Typ des Datenzugriffs kann für den Benutzer aus technischer Sicht von Interesse sein, z.B. wenn der Benutzer auf die Datenbank zugreifen möchte, um nur den Teil der Daten abzufragen, der ihn interessiert (wie in Szenario 1). Alternativ kann der Nutzer die Daten auch herunterladen oder über eine API anfragen und auf seinem Computer in einem speziellen Programm verarbeiten (wie in Szenario 2). Der Typ der Bereitstellung der Daten ist auch für den Datenmarktplatz selbst von technischer Bedeutung, da der Bestellprozess für verschiedene Provisionierungstypen unterschiedlich ablaufen kann. Die geplanten Provisionierungsoptionen für einen Datenmarktplatz können Download, Access to Database und API sein, die ebenfalls als Templates fungieren (Templates 10 – 12). Die <*download*>- und <*access*>-Templates sind einfach und enthalten nur die grundlegenden Metadaten, die zu ihrer Beschreibung notwendig sind. Das <*api*>-Template ist eine nähere Betrachtung wert (Tabelle 8, Template 12).

Tabelle 8. <*api*>-Template

Attribut name	Value type
api_type	string
root_path	string
endpoint	sting
http_method	string
description	string
parameters	*array<parameter>*
responses	*array<response>*

Eine *API* ist ein Webdienst, der es mehreren Anwendungen ermöglicht, über Anfragen und Antworten miteinander zu kommunizieren [54]. Nach [54] besteht der Vorteil einer API darin, dass sie den Zugang zu den Datenbeständen des Unternehmens erleichtert, was wiederum dazu beitragen kann, neue Verwendungsmöglichkeiten für diese Bestände im Unternehmen

zu finden. Auf dem Datenmarktplatz würde die Möglichkeit, über eine API auf Daten zuzugreifen, bedeuten, dass der Datenmarktplatznutzer, sobald der Data Owner den Zugriff genehmigt hat, das erforderliche Attributset erhält, um die Daten von dem Server, auf dem die Daten gespeichert sind, anzufragen. Bei diesen Attributen handelt es sich um den *api-type* (z.B. REST (Representational State Transfer), RPC (Remote Procedure Call) oder SOAP (Simple Object Access Protocol)), den *root_path* und *endpoint*, die *http-method*, die Anfragebeschreibung (*description*) sowie die Anfrageparameter (*parameters*) und Beispiele für mögliche Serverantworten (*responses*). Dabei sind <*parameter*> und <*response*> ebenfalls separate Metadatentemplates und enthalten die grundlegenden Attribute, die zur korrekten Angabe der Parameter der Anfrage und zum besseren Verständnis der möglichen Server-Antworten erforderlich sind (Templates 13, 14). Alle Attribute des <*api*>-Templates wurden aus den Beispielen der Swagger-Dokumentation zusammengestellt, die in [54] beschrieben sind, sowie von der Website API Commons[23], die eine Möglichkeit zur Erstellung von API-Spezifikationen bietet.

6.3.4 Templates für Kategorien

Ist der Nutzer mit den bisherigen Metadaten vertraut, kann er den Anwendungsbereich der Daten genauer betrachten. Dies kann u.a. dadurch erleichtert werden, dass bekannt ist, zu welcher Kategorie die Daten gehören.

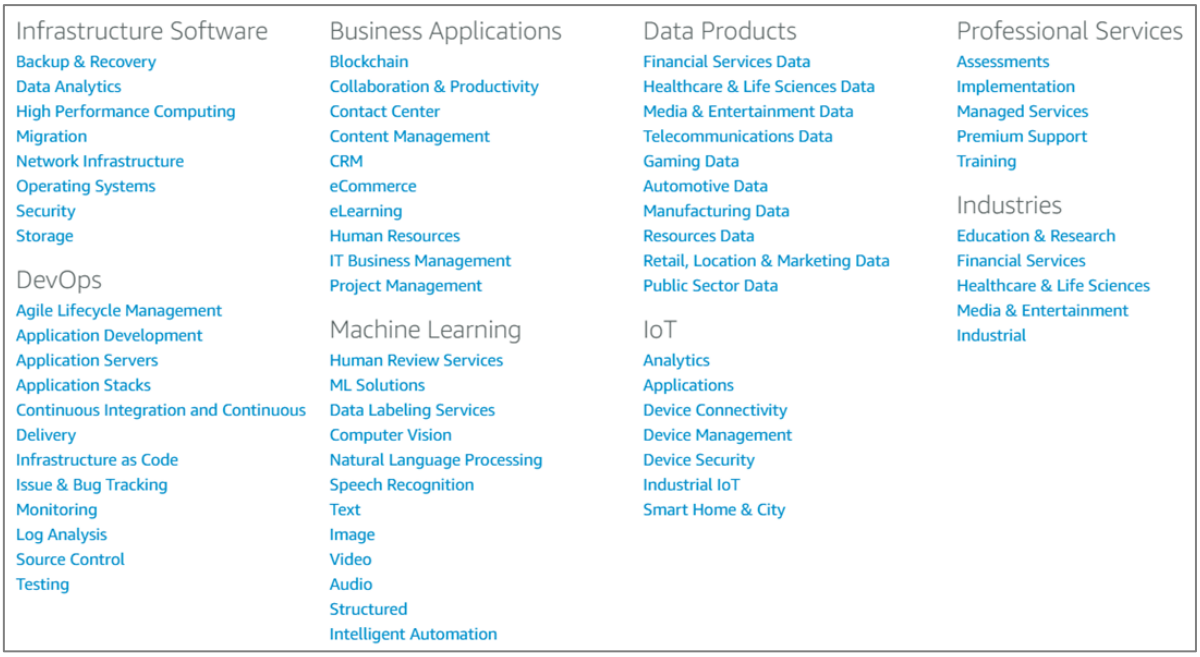


Abbildung 15. Datenkategorien des AWS Marketplaces

Zur besseren Veranschaulichung der Bedeutung von Datenkategorien lassen sich die Kategorien betrachten, die im AWS Marketplace präsentiert werden. Die Hauptkategorien sind im Screenshot (Abbildung 15) dargestellt. Wie aus der Abbildung hervorgeht, enthält der AWS Marketplace acht große Kategorien und 69 Unterkategorien. Hier ist zu sehen, dass die möglichen Produktkategorien *Finanzen*, *Gesundheit*, *Autos*, *Medien* und andere sind. Darüber

[23] API Commons: http://apicommons.org

hinaus könnten die möglichen Kategorien auch *Datendienstleistungen*, *Anwendungen* oder *Softwarecode* sein.

Auf der Grundlage der von Amazon angegebenen Kategorien und unter Berücksichtigung der in Kapitel 3 entwickelten Szenarien werden folgende Kategorien für den Datenmarktplatz vorgeschlagen:

- Finance
- Healthcare
- Insurance
- Education
- Entertainment
- Telecommunications
- Automotive
- Marketing
- IoT
- Machine Learning

Da die Kategorie auch mit dem Datentyp verbunden ist, kann ein separates Template für eine Kategorie zu einem besseren Verständnis der technischen Merkmale der Daten beitragen. So wurden beispielsweise für die Kategorien *IoT-Daten* und *Machine Learning* eigene Templates entwickelt, auf die im Folgenden etwas ausführlicher eingegangen wird. In Zukunft können je nach Bedarf auch weitere Templates für andere Kategorien erstellt werden, was in Abbildung 14 durch <…> angezeigt ist.

Templates für IoT-Daten

Nach der Definition handelt es sich bei *IoT-Daten* um Daten, die von IoT-Geräten erzeugt werden, d.h. Daten von verschiedenen Sensoren [42, 43]. Laut der IDC-Prognose[24] (International Data Corporation), die auch in [43] zitiert wird, wird die Menge der IoT-Daten im Jahr 2025 etwa 80 Zeta-Byte betragen. In [44] wird darauf hingewiesen, dass es wichtig ist, Sensordaten mit Metadaten zu versehen, da Metadaten zusätzliche Informationen über die Bedeutung der Daten und den Kontext ihrer Verwendung enthalten können. Dabei kann es sich beispielsweise um Informationen über den Sensortyp handeln, die die Daten des Temperatursensors von den Daten des Sicherheitssystems unterscheiden und so die Verwendungsmöglichkeiten der Daten differenzieren. In [55] werden auch grundlegende Metadaten von IoT-Geräten erörtert. In [42] wird sogar vorgeschlagen, einen Datenmarktplatz zu entwickeln, der ausschließlich auf die Bereitstellung von IoT-Daten spezialisiert ist.

Ein Blick auf Szenario 1 und die Anforderung *S1A3* z.B. zeigt, dass der Datenanalyst Informationen über die Sensormodelle benötigt, die zur Messung von Feuchtigkeit und Temperatur in den Lagerhäusern verwendet werden. So wurde auf der Grundlage der Informationen aus [42–44, 55] und unter Berücksichtigung der Anwendungsszenarien das Template *<iot_data>* (Tabelle 9, Template 15) für den Datenmarktplatz entwickelt. Dieses Template enthält solche Attribute wie technische Daten des Sensors, Maßeinheiten, Messfrequenz und -genauigkeit, minimal und maximal mögliche Sensorwerte, Typ der Sensorverbindung und Metadaten der Software. Für die Sensor- und Softwaredaten gibt es ebenfalls separate Templates (Templates 16, 17). Das Template *<sensor>* enthält die grundlegenden Spezifikationen des Sensors, wie Name, Typ, Modell, Seriennummer und

[24] IoT Device and Data Forecast: https://www.idc.com/research/viewtoc.jsp?containerId=US48087621

andere. Das <*software*>-Template besteht aus den Merkmalen der Software, mit der der Sensor verbunden ist, wie Name, Version und Konfiguration der Software.

Tabelle 9. <*iot_data*>-Template

Attribut name	Value type
sensor_data	<*sensor*>
measurement_units	string
sampling_frequency	string
measurement_accuracy	number
min_value	number
max_value	number
connection_type	string (Ethernet, Wi-Fi, Bluetooth)
software	<*software*>

Templates für Machine Learning Data

Die zweite Kategorie, auf die in dieser Arbeit ebenfalls eingegangen wird, umfasst die Daten des maschinellen Lernens. *Maschinelles Lernen* ist ein Gebiet, das sich mit der Entwicklung von Algorithmen unter Verwendung statistischer Methoden befasst, mit dem Ziel, den Computer in die Lage zu bringen, Probleme selbständig, d.h. ohne direkte Programmierung durch den Menschen, zu lösen [6].

In [56] wird ein Schema zur Speicherung von Metadaten von Experimenten für maschinelles Lernen vorgestellt. Auf der Grundlage der Metadaten aus diesem Schema und der in Kapitel 5 vorgestellten Quellen sowie unter Verwendung von Informationen von der Neptun-Website[25], welche das Metadaten-Repository für Machine Learning Operations (MLOps) ist, wurde ein Template <*ml_data*> für die Kategorie des maschinellen Lernens entwickelt (Template 18). Dieses Template enthält grundlegende Informationen über das Modell, das für das Training verwendet wurde, den Typ des Datensatzes, der angibt, ob der Datensatz als Trainings- oder Testsatz verwendet wurde, eine Beschreibung des Trainings- oder Test-Runs und Beispiele für Vorhersagen, die aus dem Experiment abgeleitet werden können, das mit den angebotenen Daten durchgeführt wurde.

Auch hier sei darauf hingewiesen, dass <*model*>, <*training_run*> und <*test_run*> ebenfalls separate Templates sind, die wiederum ein weiteres Template <*hyperparameters*> enthalten, das den Namen, den Typ und die Werte der Hyperparameter des Modells oder des Runs enthält (Templates 19 – 21).

6.3.5 Templates für Daten Lineage

In diesem Schritt kann davon ausgegangen werden, dass der Nutzer einige geeignete Datensätze in Bezug auf die Metadaten, die er zuvor in Betracht gezogen hat, ausgewählt hat. Er möchte nun möglicherweise wissen, wie zuverlässig diese Daten sind, d.h. ob sie vertrauenswürdig sind. Dafür möchte der Datenanalyst in Szenario 1 in der Lage sein, alle Datenänderungen zu verfolgen (*S1A4*). Wie aus Abschnitt 2.6.2 bereits bekannt, kann die

[25] Neptune Blog: https://neptune.ai/blog/ml-metadata-store

Lineage dem Benutzer helfen, alle Änderungen an den Daten zu verfolgen und die Zusammenhänge von Datensätzen mit anderen Daten zu erkennen. Daher wurde auch für die Lineage ein eigenes *<lineage>*-Template (Tabelle 10, Template 22) erstellt, das Attribute wie einen Link zur Eingabedatei, Informationen über den Prozess, durch den die Eingabedatei geändert wurde, sowie einen Link zur Ausgabedatei enthält .

Tabelle 10. *<lineage>*-Template

Attribut name	Value type
input	*link<dmp_data>*, string
output	*link<dmp_data>*, string
process	*<process>*, string

Das Attribut *prozess* hat ebenfalls ein separates Template (Tabelle 11, Template 23), das Informationen darüber enthält, von wem und wann die Änderungen vorgenommen wurden sowie ihre Beschreibung. Dabei ist zu beachten, dass Änderungen entweder von einem Mitarbeiter des Unternehmens (Template *<employee>*) oder von einer Anwendung durchgeführt werden können, für die zusätzlich ein Template *<application>* angelegt wurde, das den Namen und die Beschreibung dieser Anwendung enthält (Tabelle 12, Template 24).

Tabelle 11. *<process>*-Template

Attribut name	Value type
name	string
actor	*<employee>*, *<application>*, string
timestamp	date, string
description	string

Tabelle 12. *<application>*-Template

Attribut name	Value type
name	string
description	string

6.3.6 Templates für Data Quality

Nachdem der Nutzer nun die Daten Lineage gesehen und die Daten als vertrauenswürdig eingestuft hat, möchte er sich eventuell nach der Qualität der Daten erkundigen.

Tabelle 13. *<quality>*-Template

Attribut name	Value type	For data type
classic_metrics	*<classic_metrics>*	table, column, text
data_profilling	*<profiling>*	table, column
image_resolution	*<resolution>*	image, video
image_artefacts	text	image, video
signal_quality	*<signal_quality>*	image, audio, video
audio_total_harmonic_distortion	number	audio, video
audio_frequency_response	number	audio, video

Um den Nutzern des Datenmarktplatzes Informationen über die Datenqualität zur Verfügung zu stellen, wurde auch ein Template für solche Informationen entwickelt. Das Template *<quality>* enthält je nach Datentyp unterschiedliche Datenqualitätsparameter (Tabelle 13, Template 25). Dabei kann es sich um Qualitätsmetriken für Tabellen, Profiling oder andere Qualitätsparameter handeln, auf die in einigen der folgenden Abschnitte für jeden Datentyp näher eingegangen wird. Je nach Struktur der Daten, ihrem Inhalt und ihrem Anwendungsbereich kann es unterschiedlich sinnvoll sein, die eine oder andere Metrik zu berechnen bzw. anzugeben.

6.3.7 Templates für Preview und Reviews

Ist die Datenqualität erst bekannt und z.B. als hoch eingestuft, besteht beim Nutzer, der die Daten anfragen möchte, möglicherweise der Wunsch, zu erfahren, wie die Daten aussehen, und, wenn möglich, Bewertungen dazu zu lesen.

Damit der Benutzer herausfinden kann, wie die Daten aufgebaut sind, wird vorgeschlagen, eine Vorschau der Daten anzuzeigen. Eine solche Vorschau würde sich auch der Datenanalyst aus Szenario 1 in Anforderung *S1A7* wünschen. Es ist zu beachten, dass dies nicht immer möglich ist, insbesondere wenn die Daten vertrauliche Informationen enthalten und vor der Bereitstellung noch anonymisiert werden müssen oder von einem nicht standardisierten Typ sind. Das *<preview>*-Template enthält eine Liste der verschiedenen Datentypen, die auf dem Datenmarktplatz präsentiert werden, sowie die entsprechenden Vorschauoptionen, die zur Verfügung stehen können (Template 30). Das können z.B. die ersten 10 Zeilen einer Tabelle, die ersten 50 Wörter eines Textes, ein Thumbnail eines Bildes oder die ersten 30 Sekunden eines Audios sein.

Was die Bewertungen betrifft, so können sie es den Nutzern ermöglichen, Meinungen über Datensätze auszutauschen, und können als eine Art Crowdsourcing angesehen werden, das, wie bereits aus Abschnitt 2.1.2 bekannt, eine der Funktionen des Datenmarktplatzes ist. Das Template *<review>* enthält den Namen des Mitarbeiters, seine Bewertung (z.B. in Form von Sterne-Rating), den Kommentar und das Datum, an dem er geschrieben wurde (Template 31). Wie bei dem Template für Nutzungsbedingungen kann der Autor der Bewertung als *string* oder als *<employee>*-Template angegeben werden, das bereits in Abschnitt 6.3.1 beschrieben wurde.

6.4 Templates für strukturierte Daten

Die weitere Auswahl des Datensatzes hängt davon ab, welche Datentypen dem Benutzer zur Verfügung stehen. Der erste Typ umfasst strukturierte Daten, zu denen, wie bereits in Abschnitt 2.2.1 erwähnt, relationale Tabellen und Spreadsheets gehören. Ein Teil des Frameworks für Templates für strukturierte Daten ist in Abbildung 16 zu sehen. Zunächst wird mit einer allgemeinen Tabellen- und Spaltentemplate begonnen, von der Spreadsheets, relationale und nicht-relationale Tabellen-Templates erben (letztere werden im nächsten Abschnitt behandelt).

Das Template *<table>* ist ein einfacher Satz von grundlegenden Tabellenmerkmalen, wie die Anzahl und eine Liste von Spalten, wobei jede Spalte auch ein eigenes Template beinhaltet (Tabelle 14, Template 32). Das Template *<column>* enthält grundlegende Informationen über

die Spalte, wie z.B. die Bedeutung des Wertes, den Datentyp, die Maßeinheit und ob der Spaltenwert nullable ist (Tabelle 15, Template 33). Es wird davon ausgegangen, dass Spalten nicht separat auf dem Datenmarktplatz angeboten werden können.

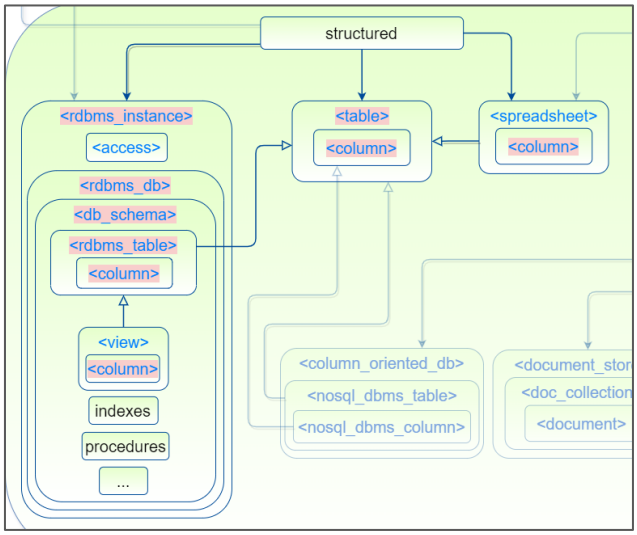


Abbildung 16. Teil des Frameworks mit den Templates für strukturierte Daten

Tabelle 14. *<table>*-Template

Attribut name	Value type
column_count	number
list_of_columns	*array<column>*

Tabelle 15. *<column>*-Template

Attribut name	Value type
name	string
meaning	string
table	*<rdbms_table>*, *<nosql_dbms_table>*, *<spreadsheet>*
data_type	string
units	string
is_nullable	boolean

Ein Typ von strukturierten Tabellen umfasst verschiedene Spreadsheets, wie Excel-Tabellen oder Google Spreadsheets. Es ist zu erkennen, dass das Template *<spreadsheet>* (Template 34) die Attribute des *<table>*-Templates erbt. Das bedeutet, dass das *<spreadsheet>*-Template zusätzlich zu den Attributen des *<table>*-Templates auch seine eigenen Attribute hat, die Informationen über die in der Tabelle verwendeten Separatoren enthalten.

Nun folgen die Templates für relationale Datenbanken. Das Dataedo-Repository[26], das Beispiele für relationale Datenbanken enthält, wurde zur Erstellung von Templates für relationale Datenbanken verwendet. Dies beginnt mit dem Template *<rdbms_instance>*, das Informationen über den Instanztyp, den Zugriff auf die Instanz, die Anzahl sowie eine Liste

[26] Dataedo Repository: https://dataedo.com/samples/html/AdventureWorks/index.html

aller Datenbanken in der Instanz enthält (Template 35). Mit *Instanz* ist ein bestimmtes Datenbankmanagementsystem gemeint, z.B. PostgreSQL oder MySQL. Hierzu zählt auch das *<access>*-Template, das bereits aus Abschnitt 6.3.3 bekannt ist. Das Template *<rdbms_instance>* ist zusätzlich in Tabelle 16 abgebildet.

Tabelle 16. *<rdbms_instance>*-Template

Attribut name	Value type
rdbms_type	string
access	*<access>*
rdbms_db_count	number
list_of_rdbms_dbs	*array_link<rdbms_db>*, array<string>

Jede Instanz kann eine bis mehrere Datenbanken umfassen. Das Template für die Datenbank *<rdbms_db>* enthält die Anzahl der Datenbankschemata sowie eine Liste der Links auf diese Schemata, wenn sie auch auf dem Datenmarktplatz registriert, und eine Liste ihrer Namen, wenn sie nicht registriert sind (Tabelle 17, Template 36). Hier wird zudem die Instanz angegeben, zu der die Datenbank gehört.

Tabelle 17. *<rdbms_db>*-Template

Attribut name	Value type
type	string
rdbms_instance	*link<rdbms_instance>*, string
db_schema_count	number
list_of_db_schemas	*array_link<db_schema>*, array<string>

Tabelle 18. *<db_schema>*-Template

Attribut name	Value type
rdbms_db	*link<rdbms_db>*, string
object_count	number
list_of_objects	*array_link<rdbms_table>*, *array_link<view>*, array<string>

Tabelle 19. *<rdbms_table>*-Template

Attribut name	Value type
db_schema	*link<db_schema>*, string
rdbms_db	*link<rdbms_db>*, string
primary_key	string
foreign_keys	array<string>
row_count	number

Jedes Schema, wobei es auch mehrere geben könnte, wird durch das Template *<db_schema>* beschrieben und enthält eine Liste und eine Anzahl von Objekten, die es enthält, sowie Links dazu (Tabelle 18, Template 37). Die Objekte können *tables*, *views*, *functions*, *procedures* und andere sein. Für *tables* und *views* wurden separate Templates entworfen (Templates 38, 39). Das Template für eine relationale Tabelle *<rdbms_table>* enthält Verweise auf das Schema

und die Datenbank, zu der die Tabelle gehört, sowie Informationen über die Anzahl der Zeilen und die Primär- und Fremdschlüssel (Tabelle 19, Template 38).

Jetzt lässt sich das Attribut *quality* betrachten, das die Templates für strukturierte Daten von dem allgemeinen <*dmp_data*>-Template erben. Für die Tabellen und Spalten wurden die folgenden Qualitätsmetriken ausgewählt: Genauigkeit (*accuracy*), Vollständigkeit (*completeness*), Konsistenz (*consistency*), Aktualität (*timeliness*), Eindeutigkeit (*uniqueness*) und Validität (*validity*). Alle diese Metriken werden in dem Template <*classic_metrics*> gesammelt (Template 26) und können dann auf der Grundlage der Informationen aus [33, 34] etwas detaillierter definiert werden.

Genauigkeit wird durch die Fehlerfreiheit der Werte definiert. Die Daten sind genau, wenn alle Werte der Realität entsprechen. Für die Berechnung der Genauigkeit von Daten wird im Allgemeinen eine Tabelle mit den korrekten Werten oder einem korrekten Wertebereich zum Vergleich benötigt. *Vollständigkeit* bedeutet, dass alle relevanten Informationen vorhanden sind. Die Daten gelten als vollständig, wenn sie keine fehlenden Werte enthalten. *Konsistenz* erfordert, dass die Daten auf die gleiche Weise dargestellt werden, z.B. immer im gleichen Format vorliegen oder vom gleichen Typ sind. *Aktualität* gibt an, wie aktuell die Daten zum Zeitpunkt ihrer Nutzung sind. *Eindeutigkeit* beschreibt, ob und welche Werte im Datensatz eindeutig sind. *Validität* bestimmt die Gültigkeit eines Wertes in Bezug auf sein Format oder seinen Typ. So kann beispielsweise das Uhrzeit-, Datums- oder Postleitzahlenformat überprüft werden. Somit kann die Anforderung *S1A6* durch die Verwendung der Metriken *accuracy, completeness* und *consistency* erfüllt werden. Wenn z.B. 20 der 100 Werte in der Spalte „Luftfeuchtigkeit“ fehlen, wird dem Datenanalysten die Vollständigkeit von 80% angezeigt. [33, 34]

Es wird außerdem vorgeschlagen, *Data-Profiling*, das Werte für einige statistische Parameter liefert, auf dem Datenmarktplatz darzustellen. Dafür wurde das Template <*profiling*> angelegt, das solche Werte wie z.B. *min_value*, *max_value*, *sum*, *average*, *null_count*, *distinct_count* und Informationen über Quartile enthält (Template 27). Wenn der Nutzer mit dem Wert der Datenqualitätsmetrik nicht zufrieden ist, kann die Data-Profiling ihm beispielsweise helfen zu verstehen, wie viele Werte in einem Datensatz fehlen oder wie viele eindeutige Werte vorhanden sind.

6.5 Templates für semi-strukturierte Daten

Der nächste Typ von Daten umfasst semi-strukturierte Daten, für die mehrere separate Templates ausgearbeitet wurden (Templates 40 – 59) und deren Teil des Frameworks in Abbildung 17 gezeigt wird. Wie bei den strukturierten Daten beginnt hier alles mit dem Template für Instanzen <*nosql_dbms_instance*> (Template 40). Im Allgemeinen enthält das Template dieselben Attribute wie das Template für die Instanzen von relationalen Datenbanken, mit dem Unterschied, dass sich alle Attribute auf nicht-relationale Datenbanken beziehen. Die nicht-relationalen Datenbankinstanzen können in diesem Fall z.B. MongoDB oder Cassandra sein.

NoSQL-Datenbanken werden mit dem Template <*nosql_dbms_db*> (Tabelle 20, Template 41) beschrieben und enthalten den Typ der Datenbank und den Link zur Instanz, zu der sie gehört, oder deren Namen, falls die Instanz nicht separat auf dem Datenmarktplatz registriert ist.

Dieses Template ähnelt dem Template für relationale Datenbanken, enthält aber im Gegensatz zu diesem keine Informationen über das Datenbank-Schema.

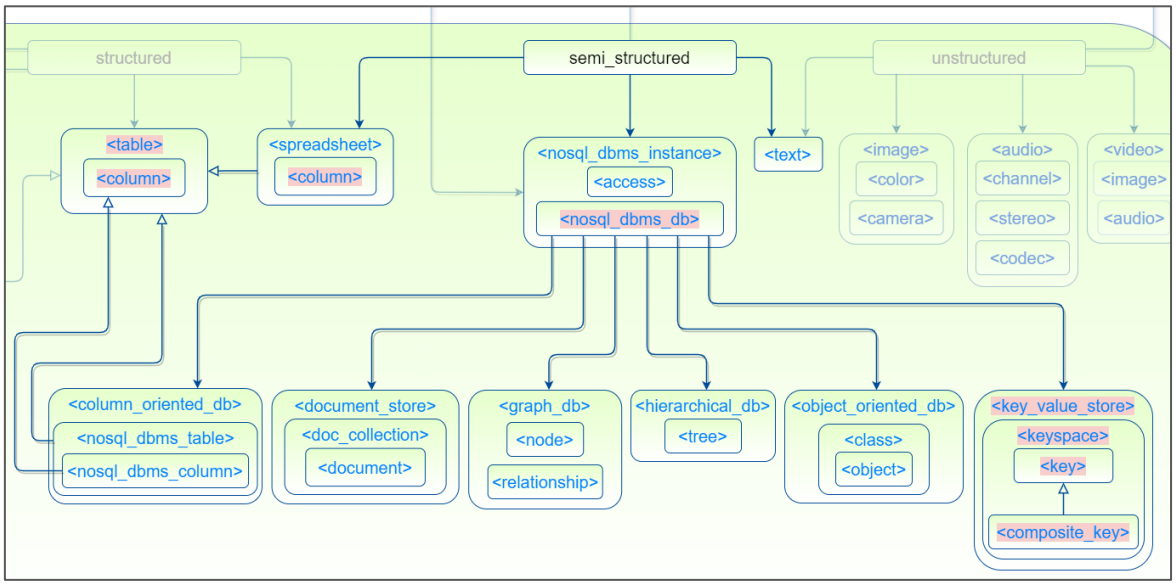


Abbildung 17. Teil des Frameworks mit den Templates für semi-strukturierte Daten

Tabelle 20. *<nosql_dbms_db>*-Template

Attribut name	Value type
type	*<column_oriented_db>*, *<document_store>*, *<graph_db>*, *<hierarchical_db>*, *<object_oriented_store>*, *<key_value_store>*, string
nosql_dbms_instance	*link<nosql_dbms_instance>*, string

Separate Templates wurden auch für solche nicht-relationale Datenbanktypen wie spaltenorientierte, Dokument-, Graph-, hierarchische sowie objektorientierte und Key-Value-Datenbanken erstellt (Templates 42 – 59).

Das Template für spaltenorientierte Datenbanken hat zwei Attribute, nämlich die Anzahl der Tabellen, die die Datenbank enthält, und eine Liste dieser Tabellen in Form von Links zu ihnen (Template 42). Wenn dieses Template im Detail betrachtet wird, stellt sich heraus, dass es zwei weitere Templates enthält, und zwar *<nosql_dbms_table>* und *<nosql_dbms_column>*, die von den Templates *<table>* bzw. *<column>* erben (Templates 43, 44). Daher sollte das Template für eine nicht-relationale Tabelle neben den Attributen des *<table>*-Templates, die sie erbt, auch auf die Datenbank verweisen, zu der die Tabelle gehört, während das Template für eine Spalte zusätzlich ein solches Attribut wie *record_count* umfasst.

Nach dem gleichen Prinzip wurden auch Templates für andere Typen von nicht-relationalen Datenbanken erstellt, bei denen andere Templates ebenfalls ein Teil davon sind, wie z.B. *<doc_collection>* und *<document>* in einer Dokument-, *<node>* und *<relationship>* in Graph-, *<tree>* in einer hierarchischen oder *<class>* und *<objekt>* in einer objektorientierten Datenbank. (Templates 45 – 55)

Interessant ist auch das Template für die Key-Value-Datenbank (Tabelle 21), das in seinem zusätzlichen *<keyspace>*-Template (Tabelle 22) zwei weitere Templates enthält, für einen einfachen und einen zusammengesetzten Schlüssel (Tabellen 23, 24). Hier ist zu sehen, dass der zusammengesetzte Schlüssel die Attribute des einfachen Schlüssels erbt, d.h. er hat alle Attribute des einfachen Schlüssels und zudem drei weitere. (Templates 56 – 59)

Tabelle 21. *<key_value_store>*-Template

Attribut name	Value type
keys_count	number
list_of_keys	*array<key>*, array<string>
keyspace_count	number
list_of_keyspaces	*array<keyspace>*, array<string>

Tabelle 22. *<keyspace>*-Template

Attribut name	Value type
keys_count	number
list_of_keys	*array<key>*, array<string>

Tabelle 23. *<key>*-Template

Attribut name	Value type
attribute_count	number
list_of_attributes	array<string>

Tabelle 24. *<composite_key>*-Template

Attribut name	Value type
partition_key	string
sort_keys_count	number
list_of_sort_keys	*array<key>*, array<string>

6.6 Templates für unstrukturierte Daten

Der letzte Datentyp, für den Templates erstellt wurden, umfasst unstrukturierte Daten. Wie aus Abschnitt 2.2.1 hervorgeht, kann es sich dabei um Texte, Bilder sowie Audio- und Videodateien handeln. Für alle diese Daten wurden separate Metadatentemplates, deren Teil des Frameworks in Abbildung 18 dargestellt ist, für den Datenmarktplatz entwickelt, auf die im Folgenden näher eingegangen wird.

Alle Metadatentemplates für unstrukturierte Daten wurden mit Hilfe von Microsoft Office[27], Webseiten zur Metadatenextraktion wie Metadata2Go[28] sowie von ExifTool[29] und Apache Tika[30] entwickelt (Typen von bestehenden Tools für die automatische Extraktion von Metadaten für verschiedene Datentypen werden in Kapitel 8 behandelt).

[27] Ressourcen zu Microsoft Office: https://www.microsoft.com/de-de/microsoft-365
[28] Metadata2go – Free online Exif viewer: https://www.metadata2go.com
[29] ExifTool by Phil Harvey: https://exiftool.org
[30] Apache Tika – a content analysis toolkit: https://tika.apache.org

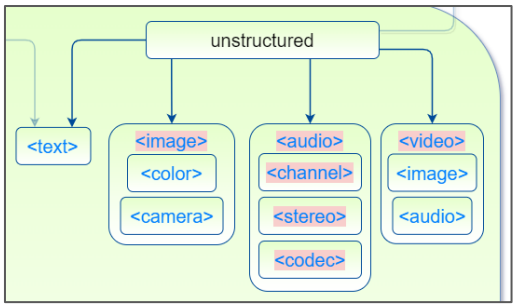


Abbildung 18. Teil des Frameworks mit den Templates für unstrukturierte Daten

6.6.1 Template für Textdaten

Zunächst werden die Textdaten betrachtet. Obwohl der Text im Framework (Abbildung 10) auch als Teil unstrukturierter Daten eingestuft wurde, kann er, wie bereits in Abschnitt 2.2.1 erwähnt, in einigen Fällen auch semi-strukturiert sein (z.B. in E-Mails), so dass es einen zusätzlichen Pfeil gibt, der den Text mit den semi-strukturierten Daten verbindet, deren Templates bereits beschrieben wurden. In diesem Abschnitt wird der Schwerpunkt auf den Text als Beispiel für unstrukturierte Daten gelegt.

Das Template *<text>* enthält grundlegende Metadaten wie die Anzahl der Seiten, Wörter, Zeichen, Zeilen und Absätze im Dokument oder Schlüsselwörter sowie technische Attribute wie Kodierung, Seitenlayout und Schriftarten (Template 60). Der Business-Analyst aus Szenario 2 kann z.B. anhand dieses Templates Berichte über Kundenfeedback auswählen, indem er das Attribut *keywords* verwendet (*S2A5*). Handelt es sich um eine Präsentation, die ebenfalls eine Variante von Textdaten darstellt, können Informationen über das Format der Präsentation und die Anzahl der Folien von Interesse sein. Es ist zu beachten, dass das Format der Präsentation nicht mit dem Format der Datei selbst identisch sein muss, das ebenfalls angegeben wird, da das *<text>*-Template vom allgemeinen Template erbt. Im Falle der Präsentation geht es um das Seitenverhältnis, d.h. ein klassisches Verhältnis von 4:3 oder Breitformat mit einem Verhältnis von 16:9.

Was das Attribut *quality* betrifft, das auch vom allgemeinen Template geerbt wird, so wurden in [57] und [58] die Anforderungen an die Qualität unstrukturierter Textdaten formuliert und Textqualitätsmetriken angegeben. Auf der Grundlage der Informationen aus diesen Quellen wird vorgeschlagen, die folgenden Metriken für die Textqualität zu verwenden: *completeness*, *correctness* (*Korrektheit*), *consistency* und *timeliness*, die bereits im Template *<classic_metrics>* enthalten sind (Templates 25, 26). Die Korrektheit des Textes kann sich z.B. auf die Grammatik oder Semantik beziehen [57].

6.6.2 Templates für Bilderdaten

Der zweite Typ von unstrukturierten Daten, für den Templates betrachtet werden sollen, betrifft Bilder. Um den Datenmarktplatz nicht mit zu vielen Metadaten zu überladen, wurden mit Hilfe der EXIF-Tags[31] und des MPEG-7-Standards, das bereits in Abschnitt 2.6.3 genannt wurde, nur diejenigen Metadaten ausgewählt, die beim automatischen Auslesen aus Bildern

[31] EXIF Tags: https://exiftool.org/TagNames/EXIF.html

am häufigsten vorkommen. So umfasst das Template *<image>* technische Daten, die die Größe des Bildes, seine Farbe und die Eigenschaften der Kamera beschreiben (Tabelle 25, Template 61). Außerdem gibt es Metadaten zu Geolokalisierung, Pixeldichte und Komprimierung. Auch die Attribute *camera* und *color_space* sind extra Templates, die zusätzliche Metadaten enthalten (Templates 62, 63). Solche technischen Daten könnten für den Datenanalysten aus Szenario 1 nützlich sein, wenn er nach Fotos von fehlerhaften Produkten sucht (*S1A11*).

Tabelle 25. *<image>*-Template

Attribut name	Value type
width	number
height	number
camera	*<camera>*, string
color_space	*<color>*, string
filter	string
geolocation	string
megapixel_count	number
image_compression	number, string

Wenn es um Bilder geht, gibt es eine große Anzahl möglicher Metriken zur Darstellung ihrer Qualität. In [59] wird beispielsweise erwähnt, dass es 100 Metriken für die Bildqualität gibt, und es wird festgestellt, dass verschiedene Metriken unterschiedliche Ziele verfolgen. Darüber hinaus kann sich die Bildqualität während der Erstellung, Übertragung oder Verarbeitung des Bildes ändern [59]. In [60] wird erwähnt, dass MSE (Mean Squared Error)-, PSNR (Peak Signal to Noise Ratio)- und SNR (Signal to Noise Ratio)-Metriken am häufigsten zur Bewertung der Bildqualität verwendet werden, worauf später noch eingegangen wird. Und in [61] wird eine Reihe grundlegender Parameter, die die Bildqualität beeinflussen, am Beispiel von Röntgenbildern vorgestellt und beschrieben. Dazu gehören räumliche Auflösung, Rauschen sowie das Vorhandensein von Artefakten.

Die *Bildauflösung* bezieht sich auf die Anzahl der Pixel pro Flächeneinheit. Je kleiner die Pixelgröße und der Abstand zwischen den Pixeln ist, desto höher ist die Auflösung [61]. In [62] wird erklärt, dass *Bildrauschen* dann auftritt, wenn die Pixelwerte variieren; das Auftreten von Rauschen führt zu einer erhöhten Körnigkeit des Bildes. Eine Methode zur Beschreibung des Rauschens ist das SNR, das das Verhältnis zwischen der Leistung des Bildsignals und der Leistung des Rauschens angibt. Je höher der SNR-Wert, desto weniger Rauschen ist zu sehen. Rauschmessverfahren werden auch in ISO 15739 [63] abgedeckt. Zwei weitere Metriken, die in [59] diskutiert werden, sind MSE und PSNR. Die MSE-Metrik wird als die mittlere quadratische Abweichung zwischen dem bewerteten Bild und dem Referenzbild berechnet. PSNR hingegen schätzt die Differenz zwischen dem maximal möglichen Signalwert und der Rauschleistung. *Artefakte* sind Elemente eines Bildes, die nicht zum abgebildeten realen Objekt gehören [64]. Es sei darauf hingewiesen, dass die Art und die Anzahl der Artefakte sowie ihre Ursachen von dem jeweiligen Typ des Bildes und der für seine Erstellung verwendeten Technologie abhängen. In [64] heißt es beispielsweise, dass es sich bei Artefakten in digitalen Röntgenbildern um weiße vertikale oder horizontale Linien handeln kann, die das Ergebnis einer fehlerhaft durchgeführten Röntgenuntersuchung sind.

So wurden auf der Grundlage der Informationen aus [59–64] die Bildauflösung, die SNR- und PSNR-Werte sowie eine Liste der eingestellten Artefakte als Qualitätsparameter für die Bilder auf dem Datenmarktplatz ausgewählt und in den Templates *<resolution>* und *<signal_quality>* dargestellt (Templates 28, 29). Das Template *<signal_quality>* ist ebenfalls in Tabelle 26 dargestellt.

Tabelle 26. *<signal_quality>*-Template

Attribut name	Value type
mean_squared_error	number
signal_to_noise_ratio	number
peak_signal_to_noise_ratio	number

6.6.3 Templates für Audiodaten

Der nächste Typ von unstrukturierten Daten wird von den Audiodaten gebildet. Die Metadaten wurden hier auf die gleiche Weise erfasst wie bei den Bildern mit den zu Beginn des Abschnittes 6.6 erwähnten Tools. Das *<audio>*-Template enthält grundlegende Metadaten wie z.B. *layer_count*, *bitrate*, *sample_rate*, *channel*, *stereo*, *codec* oder *duration* sowie beschreibende Metadaten wie *genre*, *album* und *artist* (Tabelle 27, Template 64). Es sei darauf hingewiesen, dass die Anforderung des Business-Analysten aus Szenario 2 bezüglich der Länge der Audioaufzeichnung von Gesprächen mit dem Kunden durch dieses Template für Audio abgedeckt werden kann (*S2A4*).

Tabelle 27. *<audio>*-Template

Attribut name	Value type
layers_count	number
bitrate	number
sample_rate	number
channel	*<channel>*
stereo	*<stereo>*
codec	*<codec>*
genre	string
album	string
artist	string
duration	number, string

Auch hier ist zu bemerken, dass die Attribute *channel*, *stereo* und *codec* separate Templates beschreiben, die weitere technische Metadaten enthalten (Templates 65 – 67). Zur besseren Orientierung sind die Templates *<channel>*, *<stereo>* und *<codec>* in diesem Kapitel zusätzlich in den Tabellen 28 – 30 aufgeführt.

Wie auch bei Bildern ist die Audioqualität kein eindeutiges Konzept und es wurde eine große Anzahl von Techniken entwickelt, um sie zu messen. Nach [65] sind SNR, MSE, PSNR, THD (Total Harmonic Distortion) und Frequency Response die wichtigsten Qualitätsmetriken, die am häufigsten verwendet werden. SNR, MSE und PSNR geben im Wesentlichen die gleichen Werte an, die auch bei der Messung von Bildqualität zum Einsatz kommen, mit dem

Unterschied, dass sie hier für das Audiosignal angewendet werden. Der THD-Wert bestimmt den Grad der Verzerrung des Signals während der Wiedergabe und wird in Dezibel gemessen. Der Frequency Response beschreibt den Grad der Frequenzumwandlung des Eingangssignals, gemessen in Hertz. Diese Metriken werden empfohlen, um die Qualität von Audiodateien auf dem Datenmarktplatz zu bestimmen (Templates 25, 29).

Tabelle 28. *<channel>*-Template

Attribut name	Value type
mode	string
count	number

Tabelle 29. *<stereo>*-Template

Attribut name	Value type
ms_stereo	string (on, off)
intensity_stereo	string (on, off)

Tabelle 30. *<codec>*-Template

Attribut name	Value type
name	string
long_name	string
type	string

6.6.4 Template für Videodaten

Den letzten Typ von unstrukturierten Daten bilden Videos. In dem Template *<video>* werden technische Metadaten wie *time_scale*, *avg_bitrate*, *track_volume* oder *frame_rate* angezeigt. Die Besonderheit dabei ist, dass Videos immer auch Bild- und Audio-Metadaten enthalten. Daher wurden alle Metadaten, die sich auf die Attribute *track* und *image* beziehen, als Templates *<audio>* und *<image>* dargestellt. (Tabelle 31, Template 68)

Tabelle 31. *<video>*-Template

Attribut name	Value type
time_scale	number
duration	number, string
preview_time	number, string
preview_duration	number, string
track	*<audio>*
track_volume	number
avg_bitrate	number
image	*<image>*
frame_rate	number
compressor_name	string
rotation	number

Auch für die Videoqualität gibt es eine Vielzahl von Metriken, von denen einige z.B. in [66] und [67] diskutiert werden. Wie im Falle von Bildern und Audiodaten werden auch hier die Metriken MSE, PSNR und SNR verwendet, die im Template *<signal_quality>* abgebildet sind (Template 29).

Es sei angemerkt, dass die Qualitätsmetriken für Bilder, Audio und Video spezifisch und ihre Werte keine Metadaten sind, die direkt aus den Datendateien entnommen werden können, sodass ihre Berechnung das Schreiben zusätzlicher Skripte oder den Einsatz spezieller Tools erfordert. Daher sollte je nach Typ des Datenmarktplatzes entschieden werden, ob die Verwendung dieser Metriken angemessen ist.

6.7 Template für Datenkollektionen

Angenommen, der Nutzer möchte, nachdem er sich mit allen in den vorherigen Abschnitten beschriebenen Metadaten vertraut gemacht hat, einen Datensatz finden, der mehrere verschiedene Datentypen enthält, oder sogar eine ganze Datenbank, die aus mehreren Tabellen besteht, dann kann er sich Kollektionen ansehen. *Kollektion* bezeichnet einen Typ von Produkt auf dem Datenmarktplatz, der mindestens aus zwei Datensätzen besteht. So können beispielsweise Kollektionen von Bildern, Dokumenten, Audio- oder Videodateien auf dem Datenmarktplatz angeboten werden. Es können auch mehrere Tabellen zu einer Kollektion zusammengefasst werden. Sind mehrere Tabellen in der gleichen Datenbank gespeichert und besteht zwischen ihnen ein solcher Zusammenhang, dass der Owner sie alle gemeinsam auf dem Datenmarktplatz zur Verfügung stellen möchte, kann die Datenbank auch das Produkt des Datenmarktplatzes sein. Das Gleiche gilt z.B. für mehrere Datenbanken, die zur gleichen Instanz gehören, wenn die Instanz auch als Produkt auf dem Datenmarktplatz vorkommen kann. Es ist zu beachten, dass die Kollektion nicht unbedingt dieselben Datentypen enthalten muss. Wenn beispielsweise auf Szenario 1 zurückgegriffen wird, könnte die Datenkollektion auch ein Produkt sein, das sowohl Tabellen mit Sensordaten als auch Bilder von fehlerhaften Holzleisten enthält. Das Template *<data_collection>* enthält eine Liste von Elementen, wobei ihre Typen und die Anzahl sowie der Komprimierungswert angegeben werden, wenn es sich z.B. um ein Archiv handelt (Template 69).

Nachdem nun alle Metadaten gesammelt und mit Hilfe von Templates strukturiert wurden, ist es sinnvoll, sie mit einem Metamodell zu beschreiben, um die Zusammenhänge zwischen ihnen zu verdeutlichen und somit diejenigen Templates oder Gruppen von Templates zu bestimmen, die für alle Datentypen, welche auf dem Datenmarktplatz angeboten werden können, relevant sind.

7 Metamodell für die Metadaten auf dem Datenmarktplatz

Abbildung 19 zeigt ein Metamodell in Form eines Entity-Relationship-Diagramms, das die Zusammenhänge zwischen den erstellten Templates beschreibt. Mit Hilfe dieses Modells ist es möglich, Metadaten zu jedem Datensatz auf dem Datenmarktplatz so zuzuordnen, dass alle in Kapitel 4 genannten Anforderungen erfüllt werden. So legt z.B. die Anforderung *A2* fest, dass die Datensätze mit einer Reihe von administrativen Metadaten versehen werden sollten. Die Anforderungen *A6* und *A8* besagen, dass die Daten Lineage und Qualität hinzugefügt werden sollten. Anforderung *A5* verlangt ihrerseits Metadaten über den Datenspeicherort. Daher sollen alle Templates oder Gruppen von Templates, die für alle Datentypen relevant sind und die somit zur Erfüllung der Anforderungen an den Datenmarktplatz beitragen, im Metamodell gesammelt werden. Das entwickelte Metamodell wird im Folgenden ausführlicher vorgestellt.

Der obere Teil des Modells enthält administrative und technische und der untere Teil beschreibende, technische und Business-Metadaten, die die Anforderungen aus Kapitel 4 abdecken. Wie in Abschnitt 2.6.2 erwähnt, ist diese Unterteilung in Metadatentypen nicht strickt, sodass sich einige Typen überschneiden können. Demzufolge ist die Zuordnung in der Abbildung 19 auch nicht völlig eindeutig und dient nur einer groben Unterscheidung. Im Folgenden werden die einzelnen Entitäten des Diagramms näher betrachtet und es wird in Klammern angegeben, welche Anforderungen an den Datenmarktplatz hinsichtlich der Verwendung von Metadaten sie abbilden.

Im Mittelpunkt des Modells steht die Entität *dataset*, die strukturierte, semi-strukturierte und unstrukturierte Daten abdeckt. Alle anderen Entitäten stellen Metadatentemplates dar. Alle im Diagramm dargestellten Attribute sind obligatorisch. Darüber hinaus kann es auch optionale Attribute geben, die von dem jeweiligen Template und dem Typ der im Datensatz enthaltenen Daten abhängen. Da die ER-Diagramme keine explizite Modellierung optionaler Attribute ermöglichen, werden sie in der Abbildung nicht gezeigt.

So sind *GUID*, *name*, *type*, *description* und *dmp type* (asset oder product) obligatorische Attribute. Eine GUID ermöglicht auch die eindeutige Identifizierung eines Datensatzes am Datenmarktplatz. (Anforderung *A1*)

Es ist erforderlich, einen Owner der Daten anzugeben, der den Zugriff auf die Daten gewähren oder verweigern kann. Für jeden Datensatz muss es mindestens einen Data Owner geben, einschließlich seines Namens und seiner Kontaktdaten. Im Diagramm ist dies die Entität *owner*. Die Entität *term of use* zeigt, dass Informationen über die Nutzungsbedingungen erforderlich sind, die vom Data Owner festgelegt werden müssen einschließlich der Lizenz, der Security Class und der Angabe, ob der Datensatz personenbezogene Daten enthält. Für jeden Datensatz muss es einen solchen Satz von Nutzungsbedingungen geben. Der Zugriff auf die Daten kann auf drei Weisen erfolgen: Zugriff auf die Datenbank, über API oder als Dateidownload. Dies wird im Diagramm durch die Entitäten *access to database*, *API* und *download* angezeigt. Die Angabe von mindestens einer dieser Optionen ist obligatorisch. (Anforderung *A2*)

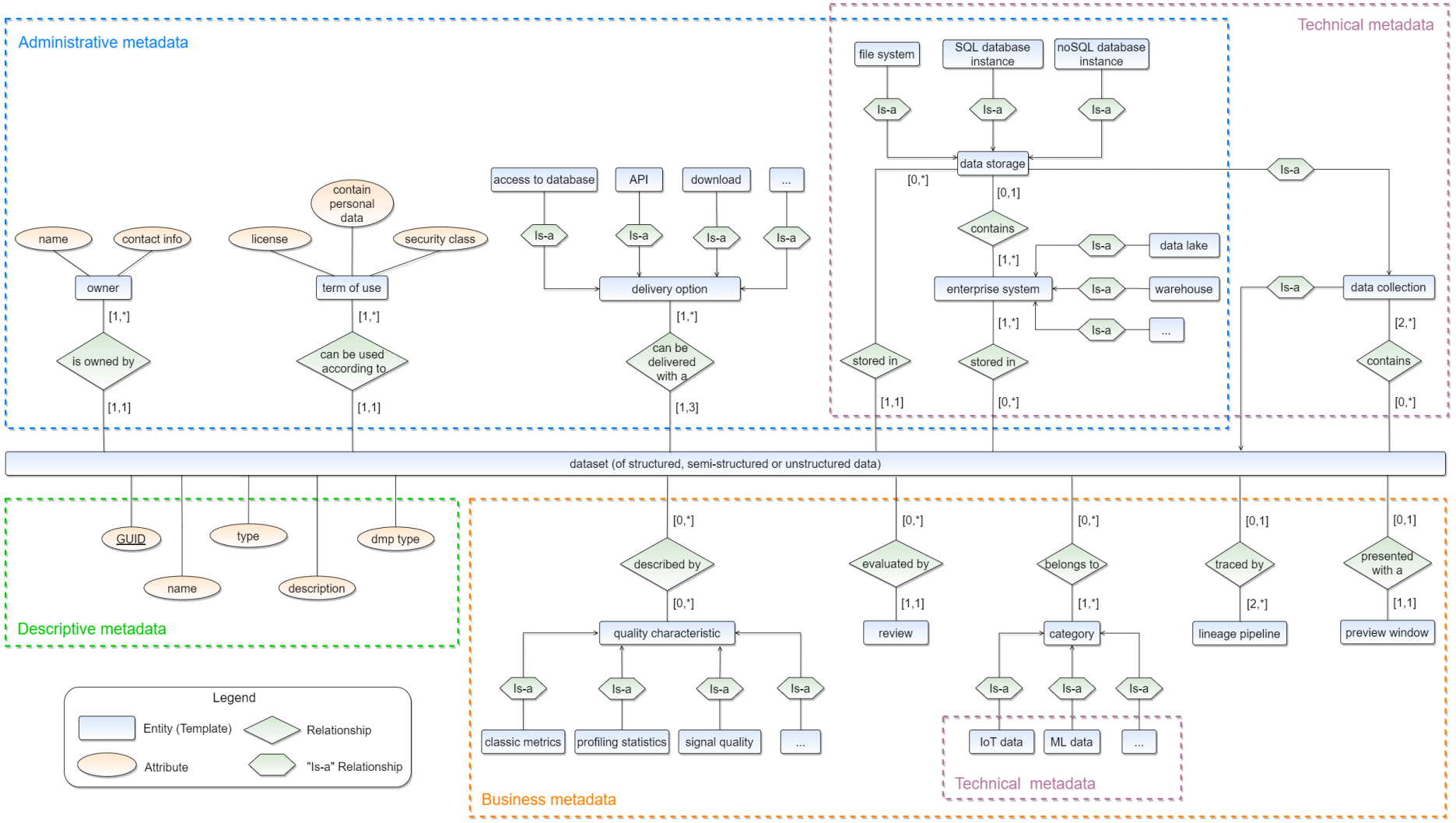


Abbildung 19. Metamodell der Metadaten, die auf dem internen Datenmarktplatz präsentiert werden sollen

Wie aus Abschnitt 2.4 hervorgeht, kann jeder auf dem Datenmarktplatz dargestellte Datensatz in einem Unternehmen in relationalen oder nicht-relationalen Datenbanken oder Dateisystemen sowie in einem Unternehmenssystem gespeichert sein, das wiederum aus einer beliebigen Anzahl verschiedener Datenbanken und Dateisystemen bestehen kann. Die Datenspeicherung wird im Diagramm durch die Entitäten *data storage*, *SQL database instance*, *NoSQL database instance*, *file system* und *enterprise system* dargestellt. In diesem Fall ist, wie im Modell ersichtlich, die Angabe mindestens eines Datenhaltungssystems obligatorisch, während die Angabe eines Unternehmenssystems nur dann relevant ist, wenn der Datenspeicher zu einem solchen System gehört. (Anforderungen *A3*, *A5*)

Die Entität *data collection* repräsentiert die Möglichkeit, wenn ein Datensatz aus mehreren Datenobjekten besteht, also z.B. mehrere Fotos, Dokumente oder Tabellen enthält. In diesem Fall wird der Datensatz als eine Kollektion betrachtet, die wiederum mehrere Elemente enthalten kann und selbst ein Datensatz ist. Darüber hinaus können relationale und nicht-relationale Datenbanken auch als Produkt angeboten werden und würden dann Kollektionen entsprechen. Wenn die Elemente in der Kollektion auch als einzelne Produkte am Datenmarktplatz registriert sind, kann der Benutzer von der Kollektion aus zu jedem dieser Datensätze navigieren. (Anforderungen *A3*, *A7*)

Darüber hinaus gibt es Entitäten, die Merkmale der Datenqualität abbilden (*quality characteristic*, *classic metrics*, *profiling statistics* und *signal quality*) die den Datensatz beschreiben und wünschenswert, aber nicht zwingend erforderlich sind. (Anforderungen *A4*, *A8*)

Die Bewertung des Datensatzes, dargestellt durch die Entität *review* im Diagramm, enthält die Meinungen der Nutzer, die bereits auf den Datensatz zugegriffen und ihn z.B. durch ein Sternerating ausgewertet haben. Die Anzahl der Bewertungen für jeden Datensatz ist unbegrenzt. Einige Datensätze können kein Rating besitzen. (Anforderung *A4*, *A11*)

Außerdem können die Daten in jedem Datensatz einer Kategorie zugeordnet werden, z.B. *IoT-Daten*, *Finanzen* oder *Marketing*. Die Angabe der Kategorie ist ebenfalls optional. Gehören die Daten zu den Kategorien des maschinellen Lernens oder der Sensordaten, können sie durch zusätzliche Metadaten beschrieben werden, die im Diagramm durch die Entitäten *IoT data* und *ML data* abgebildet werden. Es können auch Templates für andere Kategorien erstellt werden, wenn für diese spezifische Metadaten vorliegen. (Anforderungen *A3*, *A4*, *A10*)

Die Informationen über die Lineage lassen sich im Template abrufen, die durch die Entität *lineage pipeline* dargestellt wird. Wenn Informationen über die Lineage verfügbar sind, werden sie durch eine Pipeline angezeigt. Diese Angaben sind optional und gegebenenfalls nicht vorhanden. (Anforderungen *A4*, *A6*)

Wenn ein Datensatz keine personenbezogenen Daten enthält und es keine anderen rechtlichen oder technischen Beschränkungen für seine Verwendung gibt, kann eine Vorschau auf die Daten möglich sein, was im Modell durch die Entität *preview window* angegeben wird. Eine Datenvorschau ist nicht immer möglich und stellt demnach, wie im Modell zu sehen, keine Pflichtangabe dar. (Anforderungen *A4*, *A9*)

Nachdem nun alle Metadaten für den internen Datenmarktplatz in Templates zusammengefasst und durch ein Metamodell beschrieben werden, können die Tools und Methoden betrachtet werden, mit denen all diese Metadaten erfasst werden können. Darauf wird im nächsten Kapitel eingegangen.

8 Metadatenmanagement-Tool-Typen, die Metadaten für die Templates bereitstellen

Wie aus Abschnitt 2.7 bekannt, bleiben die Unternehmensdaten dank einer effizienten Verwaltung von Metadaten leicht auffindbar und für die Mitarbeiter verständlich. Um Metadaten verwalten zu können, müssen sie zunächst aus den Daten extrahiert und aufbereitet werden, wozu die *Metadatenmanagement-Tools* dienen. Ihr Zweck ist es, Informationen (Metadaten) über verschiedene Datentypen zu extrahieren bzw. zu sammeln und den Endnutzern (z.B. den Mitarbeitern des Unternehmens) bereitzustellen. In diesem Kapitel werden Tools und Methoden aufgeführt, anhand welcher die Metadaten für die unterschiedlichen Templates gewonnen werden können. Am Ende des Kapitels ist eine Tabelle mit Hinweisen zur Verwendung der Metadatenmanagement-Tool-Typen für verschiedene Gruppen von Metadaten-Templates zu finden. [68]

In [69] wird angemerkt, dass der Datenmarktplatz selbst zum Teil ein Metadatenmanagement-Tool ist, da eine seiner Funktionen, wie in Abschnitt 2.1.2 erläutert, die Verwaltung von Metadaten ist. Die administrativen und technischen Metadaten der Templates für Nutzungsbedingungen (*<term_of_use>*, *<employee>* und *<license>)*, Provisionierungsoptionen (*<delivery_options>*, *<download>*, *<access>* und *<api>*) sowie für Datenkollektionen (*<data_collection>)* und Reviews (*<reviews>*) sind z.B. Metadaten, die innerhalb des Datenmarktplatzes erstellt und gespeichert werden und nur für dessen Nutzung relevant sind.

Die weiteren verbreitetsten Tools sind Datenverzeichnisse, Glossare und Datenkataloge, sowie Datenqualitäts-, ETL-, Data-Profiling-Tools und Data Lake Management Plattformen. Es ist auch möglich, Metadaten mit Hilfe spezieller Bibliotheken und Skripte automatisch zu extrahieren. Auf alle diese Metadatenmanagement-Tool-Typen wird im Folgenden ausführlicher eingegangen. [21, 69]

8.1 Datenverzeichnisse, Glossare und Datenkataloge

Beim ersten Metadatenmanagement-Tool-Typ, handelt es sich um *Datenverzeichnisse*, in denen Informationen über Unternehmensdaten gesammelt werden. Zu diesen Informationen gehören Erläuterungen zu den in den Daten enthaltenen Werten, Details zum Kontext, in dem die Daten gewonnen werden, sowie die Angabe der Datentypen, ihres Anwendungsbereichs und die Abhängigkeiten zwischen den verschiedenen Datenobjekten. Dies kann auch Informationen über die verschiedenen Variablen, obligatorische Tabellenfelder, zulässige Spaltenwerte, Einzelheiten zu fehlenden Werten und Maßeinheiten umfassen. Der Schwerpunkt von Datenverzeichnissen liegt auf technischen Metadaten. Relationale und nicht-relationale Datenbankmanagementsysteme (DBMS) enthalten häufig integrierte Datenverzeichnisse, die Informationen über die Datenstruktur sowie Primär- und Fremdschlüssel enthalten [66]. So werden beispielsweise bei jeder Erstellung einer Oracle-Datenbank[32] alle Informationen über diese Datenbank, d.h. ihre Metadaten, dem Datenverzeichnis hinzugefügt. In einem Verzeichnis können technische Metadaten aus

[32] Oracle Concepts – Data Dictionary: http://www.dba-oracle.com/concepts/data_dictionary.htm

Templates für strukturierte und semi-strukturierte Daten gesammelt werden, wie z.B. *<rdbms_instance>*, *<rdbms_db>*, *<nosql_dbms_instance>* und *<nosql_dbms_db>*. [21, 70]

Ein weiterer Metadatenmanagement-Tool-Typ, *Glossar* ist eine Sammlung von Business-Begriffen und deren Definitionen, die in erster Linie im Zusammenhang mit ihrer Verwendung im Unternehmen stehen [25]. Ein Beispiel für ein Glossar für BVA (Bundesverwaltungsamt)[33] befindet sich in [67]. Glossare befassen sich hauptsächlich mit Business-Metadaten [27]. Business-Metadaten aus Templates für strukturierte und semi-strukturierte Daten, wie z.B. Bedeutungen der Spalten für Tabellen (*<table>*, *<column>*, *<spreadsheet>*, *<view>*) können aus Glossaren erfasst werden.

Datenkataloge stellen einen weiteren Typ von Metadatenmanagement-Tools dar. Der Zweck der Datenkatalogisierung besteht darin, alle Informationen über die im Unternehmen gespeicherten Daten an einem Ort zu sammeln, um die Suche zu erleichtern und die Duplizierung von Daten zu vermeiden. In der Regel verfügt ein Katalog über eine integrierte Suchfunktion nach verschiedenen Parametern. Die Daten im Katalog können kategorisiert sowie durch Business-Begriffe und Informationen zur Datenqualität ergänzt vorliegen. Einige Kataloge bieten auch die Möglichkeit, die Daten Lineage zu verfolgen. Im Gegensatz zu Datenverzeichnissen enthalten Datenkataloge ein breiteres Spektrum an Metadaten, das alle in dieser Arbeit zugrunde gelegten Metadatentypen abdeckt (beschreibende, administrative, technische und Business-Metadaten). So können in Datenkatalogen sowohl einige Attribute des allgemeinen *<dmp_data>*-Templates (z.B. *name*, *guid*, *description*, *type*, *dmp_type* und andere) als auch die Metadaten der Templates für Lineage (*<lineage>*), Datenqualität (*<quality>*, *<classic_metrics>*) und Datenspeicherung (*<file_system>*, *<enterprise_system>*), abgespeichert werden. [21, 71]

Ein Beispiel für einen Datenkatalog ist *Apache Amundsen*[34], der sowohl die automatische Erfassung von Metadaten aus verschiedenen Datenbanksystemen und die Datensuche als auch die Möglichkeit gewährt, benutzerdefinierte Attribute hinzuzufügen. Amundsen unterstützt auch das Data-Profiling und die Visualisierung der Datenqualität sowie die Angabe des Data Owners, jedoch keine Verfolgung der Daten Lineage. Als Beispiel für ein Tool, das die Daten Lineage unterstützt, kann *Apache Atlas*[35] genannt werden. Es ermöglicht auch die Erfassung von Metadaten für vordefinierte Datentypen, die Erstellung benutzerdefinierter Typen sowie die Suche und Klassifizierung von Datensätzen. Obwohl die Erfassung des Data Owners in Atlas nicht direkt vorgesehen ist, kann ein solches Feld für alle Datentypen hinzugefügt und als Pflichtfeld deklariert werden. Große Unternehmen können auch mehrere Datenkataloge haben, z.B. Atlas und Amundsen. In diesem Fall werden die Metadaten auf dem Datenmarktplatz für die entsprechenden Datensätze aus allen Katalogen extrahiert, zusammengeführt und zusammen dargestellt.

8.2 Datenqualitätstools

Datenqualitätstools können, wie der Name schon verrät, verschiedene Aufgaben im Zusammenhang mit der Datenqualität übernehmen. Die wichtigsten Typen von

[33] Bundesverwaltungsamt: https://www.bva.bund.de/DE/Home/home_node.html
[34] Open source data discovery and metadata engine: https://www.amundsen.io
[35] Apache Atlas: https://atlas.apache.org

Datenqualitätstools umfassen Tools für die Datenanalyse in Bezug auf Qualitätsmetriken, Data-Profiling, Datenbereinigung und -anreicherung. Die *Datenanalyse* beinhaltet die Überprüfung der Übereinstimmung der Daten mit ihrem Anwendungsbereich, z.B. durch Verwendung von Metriken, die im Template *<classic_metrics>* enthalten sind. Bei der Datenanalyse kann der Benutzer die Regeln festlegen, nach denen die Werte bestimmter Metriken berechnet werden, einschließlich der Erstellung zusätzlicher optimierter Tabellen zum Vergleich. Durch das *Data-Profiling* werden Qualitätsprobleme genauer und unabhängig vom Kontext, in dem die Daten verwendet werden, identifiziert; dies kann z.B. die Angabe von Minimal- und Maximalwerten, die Anzahl von Duplikaten sowie von gefundenen eindeutigen und ungültigen Werten beinhalten. Diese Attribute werden in dem Template *<profiling>* gesammelt. Während die Daten bei der Analyse und beim Data-Profiling unverändert bleiben, können die Daten auch bereinigt und angereichert werden, um bestimmte Bedingungen für die weitere Verwendung zu erfüllen. Die *Datenbereinigung* umfasst die Erkennung und Beseitigung von fehlerhaften Daten, Duplikaten und Nullwerten, d.h. sie befasst sich mit allen Problemen, die beim Data-Profiling festgestellt wurden. Bei der *Datenanreicherung* geht es darum, den Daten neue Informationen hinzuzufügen, um ihre Qualität zu verbessern, z.B. durch den Ersatz veralteter oder inkorrekter Daten. [18, 72]

Im Rahmen dieser Arbeit sind nur solche Funktionalitäten von Datenqualitätstools von Interesse, mit denen Informationen über die Datenqualität bereitgestellt werden, ohne die Daten selbst zu verändern. Beispiele für die Datenqualitätstools sind *Talend Data Quality*[36], *Apache Griffin*[37] und *OpenRefine*[38]. Es sei darauf hingewiesen, dass diese Tools für strukturierte und semi-strukturierte Daten konzipiert sind. Die Qualität unstrukturierter Daten wird in der Regel je nach Datentyp (Text, Bild, Audio oder Video) durch Bibliotheken oder speziell geschriebene Skripte bewertet, die in einem der folgenden Abschnitte behandelt werden.

8.3 Systemspezifische Tools

Um zu verstehen, wie spezielle Metadaten für das Data Warehouse und den Data Lake erhalten werden können, ist es notwendig, auch systemspezifische Metadatenmanagement-Tools zu betrachten. Dies sind z.B. die ETL-Tools und die Data Lake Management Plattformen, auf die im Folgenden weiter eingegangen wird.

8.3.1 ETL-Tools

Es wird in [27] angemerkt, dass das Instrument zur Gewinnung von technischen und Business-Metadaten im Unternehmen auch *ETL-Tools* sind, die die Rohdaten für Business-Nutzern in wertvolle Informationen umwandeln [23]. Z.B. umfasst dieser Prozess im Falle des Data Warehouse die Extraktion der Rohdaten, ihre Umwandlung einschließlich der Prüfung auf gültige Werte und die Konsistenz zwischen den Werten, die Entfernung von Duplikaten sowie die Zusammenführung mehrerer Datenquellen zu einer einzigen und das Laden der bereits aufbereiteten Daten in das Data Warehouse [73]. In [73] wird verdeutlicht, dass die Metadaten, die mit Hilfe von ETL-Tools extrahiert werden können, sich auf Änderungen und Verschiebungen von Daten beziehen, sodass es möglich ist, die Datenquelle,

[36] Datenqualitätslösungen: https://www.talend.com/de/products/data-quality
[37] Apache Griffin. Big Data Quality Solution For Batch and Streaming: https://griffin.apache.org
[38] OpenRefine. A free, open source, powerful tool for working with messy data: https://openrefine.org

Transformationsregeln und Ladestrategien zu verfolgen. Daher können neben der Daten Lineage (Template *<lineage>*) auch Metadaten für das Template *<data_warehouse>* mittels ETL-Tools gewonnen werden. Beispiele für ETL-Tools sind *Informatica ETL*[39], *IBM Information Server*[40] und *Talend Open Studio*[41].

8.3.2 Data Lake Management Plattformen

Bei der *Data Lake Management Plattform* handelt es sich um ein in den Data Lake integriertes Tool, das die Möglichkeiten des Datenkatalogs erweitert und den Data Lake um ETL-Tool-Funktionen ergänzt. Die erweiterte Funktionalität kann auch Informationen über die API enthalten, die für den Zugriff auf die Daten im Data Lake verwendet wird, was auch die Erfassung von API-Metadaten ermöglicht. [21]

Somit ist es mit einer Data Lake Management Plattform möglich, Metadaten für *<data_lake>*- und *<api>*-Templates zu sammeln. Ein Beispiel für eine solche Plattform, die auch in [21] angegeben wird, ist *Kylo*[42]. Kylo unterstützt das automatische Data-Profiling und die Bereinigung von Daten, ist in der Lage, Daten zu visualisieren sowie liefert Informationen über die Daten Lineage und Basisstatistiken.

8.4 Bibliotheken zum Extrahieren von Metadaten

Eine weitere Methode der Gewinnung von Metadaten ist die Verwendung von speziellen Bibliotheken. Solche Bibliotheken können dazu genutzt werden, beschreibende und technische Metadaten aus unstrukturierten Daten zu extrahieren. *ExifTool*[43] beispielsweise ist eine Perl-Bibliothek, die mit dem EXIF-Metadatenstandard arbeitet und Metadaten aus Textdaten, Bildern, Audio- und Videodaten auslesen kann. ExifTool kann Metadaten wie Typ, Größe und Format von unstrukturierten Daten, Sprache, Keywords, verschiedene technische Metadaten und sogar Thumbnails aus einer Bilddatei (was ein Attribut des Templates *<preview>* für Bilder ist) extrahieren. D.h., alle Metadaten, die in den Templates für unstrukturierte Daten (*<text>*, *<image>*, *<audio>* und *<video>*) sowie teilweise auch in dem allgemeinen *<dmp_data>*-Template gesammelt werden, können von ExifTool erfasst werden. Es ist zu bemerken, dass ExifTool die Daten nicht analysiert, sondern nur die Metadaten abliest, die in die Daten integriert sind.

Als nächstes kann *Apache Tika*, bestehend aus einer Reihe von Java-Bibliotheken, vorgestellt werden. Apache Tika wurde bereits in Abschnitt 6.6 als Hilfsmittel bei der Erstellung von Templates für unstrukturierte Daten erwähnt und ist für die Identifizierung, Analyse und Extraktion von Metadaten aus einer Vielzahl von Dateiformaten konzipiert. Solche Bibliotheken können auch bei der Pflege von IoT- [44] und Metadaten des maschinellen Lernens [56] helfen (Templates *<iot_data>* und *<ml_data>*), wenn diese Metadaten den Daten als Annotation hinzugefügt werden. Ein System zur automatischen Erfassung und Speicherung von Metadaten aus Experimenten zum maschinellen Lernen wird auch in [56] vorgestellt. In Bezug auf die Textqualität wird in [57] eine Methodik zur Berechnung von

[39] Informatica: https://www.informatica.com
[40] IBM. Enterprise server solutions: https://www.ibm.com/uk-en/it-infrastructure/servers
[41] Talend Open Studio: https://www.talend.com/de/products/talend-open-studio
[42] Kylo. An open-source data lake management software platform: https://kylo.io
[43] ExifTool by Phil Harvey: https://exiftool.org

Qualitätsmetriken erläutert, für die einige von Apache Tika gewonnene Metadaten verwendet werden können.

8.5 Erstellung von benutzerdefinierten Metadatenmanagement-Tools

Es könnte auffallen, dass nicht alle Metadaten, die in Templates gesammelt werden, automatisch aus den Daten extrahiert werden können. Die einfachste, aber zeitaufwändigste Lösung besteht darin, die Metadaten manuell einzugeben und z.B. in den Datenkatalog aufzunehmen. Damit die Extraktion und der Austausch von Metadaten automatisch erfolgen, können auch spezielle Skripte für die Erfassung von Metadaten erstellt werden, für die es kein Tool gibt, das direkt angewendet werden könnte. Es wird z.B. empfohlen, Skripte zu schreiben, um die Metadaten des Templates *<signal_quality>* zu bekommen. Wie bereits in Abschnitt 6.6 erläutert, sollte die Notwendigkeit, diese Attribute auf dem Datenmarktplatz bereitzustellen, auch in Abhängigkeit von der Spezialisierung des Datenmarktplatzes geprüft werden. Wenn z.B. bekannt ist, dass das Unternehmen keine große Anzahl von Bildern speichert, oder alle vorhandenen Bilder z.B. Screenshots mit informativem Charakter sind, können die SNR- und PSNR-Werte für die Mitarbeiter des Unternehmens keine wertvolle Information darstellen. Diese Parameter können jedoch für ein Unternehmen wichtig sein, das sich z.B. mit der Analyse medizinischer Bilder oder Fotos von fehlerhaften Produkten beschäftigt, wie in Szenario 1.

8.6 Gleichzeitige Nutzung mehrerer Metadatenmanagement-Tools

Aus den vorherigen Abschnitten wurde ersichtlich, dass einige der Tools ähnliche Funktionen aufweisen. So lässt sich beispielsweise die Daten Lineage sowohl durch ETL-Tools als auch durch Eintragung in einen Katalog nachverfolgen. Außerdem wird in [69] erwähnt, dass ein Katalog auch ein Glossar enthalten kann. Darüber hinaus wurde in Abschnitt 8.1 erwähnt, dass ein Datenkatalog auch Informationen über Data-Profiling und Datenqualität enthalten kann. [1, 2]

Verschiedene Typen von Metadatenmanagement-Tools können auch an ein System angeschlossen werden. So wird in [27] ein Beispiel für ein IBM-Metadaten-Repository gegeben (International Business Machines Corporation)[44], in dem Business-Glossar, Tool für die Data-Profiling, ETL- und Datenqualitätstool standardmäßig integriert sind. In [27] wird auch ein Beispiel für die Verwendung einer solchen Kombination angegeben: der Benutzer findet einen Business-Begriff im Glossar und liest dessen Definition, dann kann er die Ergebnisse des Data-Profilings einsehen und die Übereinstimmung der Daten mit der Definition im Glossar überprüfen. Falls die Daten nicht übereinstimmen, kann das ETL-Tool zu Rate gezogen werden und der Benutzer kann alle Datenänderungen nachverfolgen, um festzustellen, zu welchem Zeitpunkt die Dateninkongruenz aufgetreten ist. Dieser Prozess ähnelt dem Prozess der Auswahl eines Datensatzes auf dem Datenmarktplatz.

Die Kombination mehrerer Tools in einem System ist auch mit Hilfe von Egeria-Plattform möglich, die in Kapitel 5 bei der Betrachtung verwandter Arbeiten kurz vorgestellt wurde. Egeria ist ein Tool, das automatisch Metadaten aus verschiedenen Tools sammelt. Diese Aufgabe ähnelt wieder der Aufgabe des Datenmarktplatzes – verschiedene Metadaten zu

[44] IBM: https://www.ibm.com/de-de

sammeln, für die Templates entwickelt wurden, und dem Nutzer alle diese Metadaten auf einheitliche Weise zu präsentieren.

Um Metadaten auf dem Datenmarktplatz effektiv zu verwalten, ist es daher empfehlenswert, die folgenden Metadatenmanagement-Tools und Möglichkeiten zu kombinieren:

- Datenverzeichnis
- Datenkatalog
- Glossar
- Data-Profiling-Tool
- ETL-Tool
- Datenqualitätstool
- Bibliotheken zum Extrahieren von Metadaten
- Metadaten Repository des Datenmarktplatzes
- Skripte zur Extraktion von Metadaten

Tabelle 32 zeigt die Empfehlungen für die Verwendung von Metadatenmanagement-Tool-Typen auf dem Datenmarktplatz für jeden Metadatentemplate-Typ.

Tabelle 32. Übersicht über die empfohlenen Metadatenmanagement-Tools für die Extraktion von Metadaten auf dem Datenmarktplatz

Template-Gruppe	Metadatenmanagement-Tool
Allgemeines Template	Datenkatalog, Bibliotheken, Datenmarktplatz
Templates für Nutzungsbedingungen	Datenmarktplatz
Templates für Datenspeicherung	Datenkatalog, ETL-Tools, Data Lake Management Plattform
Templates für Daten-Provisionierungsoptionen	Datenmarktplatz
Templates für Kategorien	Datenkatalog, Bibliotheken
Templates für Daten Lineage	Datenkatalog, ETL-Tool, Data Lake Management Plattform
Templates für Datenqualität	Datenkatalog, Datenqualitätstools, Bibliotheken, Skripte zur Metadatenextraktion
Templates für Data Preview und Reviews	Datenmarktplatz, Bibliotheken
Templates für strukturierte Daten	Datenverzeichnis (RDBMS), Datenkatalog, Glossar
Templates für semi-strukturierte Daten	Datenverzeichnis (NoSQL-DBMS), Datenkatalog, Glossar
Templates für unstrukturierte Daten	Bibliotheken, Skripte zur Metadatenextraktion
Template für Datenkollektionen	Datenmarktplatz

Somit enthält die obige Tabelle Hinweise darauf, welche Typen von Metadatenmanagement-Tools an den Datenmarktplatz angeschlossen werden können, um Metadaten für die Templates zu erfassen. Die nächsten Schritte dieser Arbeit bestehen in der Entwicklung eines Konzepts zur Darstellung der gesammelten Metadaten im Datenmarktplatz-Frontend sowie in der Umsetzung dieses Konzepts, was in den nächsten beiden Kapiteln behandelt wird.

9 Entwurf zur Darstellung der Templates im Datenmarktplatz-Frontend

In diesem Kapitel wird ein Konzept zur Darstellung von Metadaten aus verschiedenen Templates im Datenmarktplatz-Frontend vorgestellt. Ein Blick auf die in Kapitel 6 vorgestellten Templates und das Metamodell aus Kapitel 7 zeigt, dass alle Metadaten bereits in einige Gruppen gegliedert sind, so dass sie auch auf der Detailseite auf gleiche Weise strukturiert werden können. Auch aus Kapitel 5, in dem verwandte Arbeiten erörtert wurden, geht hervor, dass die Strukturierung von Metadaten eine gute Möglichkeit bietet, sie zu organisieren. Um das Konzept der Darstellung von Metadaten aus Templates auf der Detailseite des internen Datenmarktplatzes zu präsentieren, wurden Mockups mit Balsamiq[45] entwickelt. Alle Mockups sind Anhang 2 zu entnehmen.

9.1 Konzept der Gliederung der Metadaten auf der Detailseite

Bei genauerer Betrachtung des allgemeinen *<dmp_data>*-Templates wird deutlich, dass es sechs zusätzliche Templates enthält, d.h. Attribute, die selbst Templates sind. Das bedeutet, dass diese Attribute nicht durch einen String oder einen anderen Datentyp ausgedrückt werden können, da sie komplex sind und auch weitere Attribute oder eingebaute Templates enthalten. Es sollte klargestellt werden, dass sich der Begriff „eingebaute Templates“ auf Templates bezieht, die Teil anderer Templates sind, aber keine eigenständige Metadatengruppe sein können, da sie von einem Attribut abhängen und auch durch einen Stringwert ersetzt werden können, wenn ihre detaillierten Metadaten nicht verfügbar sind. Z.B. können die Werte für die Attribute *location* und *category*, wie aus dem Template *<dmp_data>* hervorgeht, auch nur Strings sein, während z.B. Lineage oder Datenqualität nicht durch einen einzigen String ersetzt werden können. Somit kann die Gruppierung der Metadaten auf dem allgemeinen Template zusammen mit sechs weiteren Templates basieren. D.h. die folgenden Templates werden bei der Gruppenbildung einbezogen:

<dmp_data> <term_of_use> <quality> <lineage> <preview> <delivery_options> <reviews>.

Schon anhand dieser Zeile wird deutlich, dass die ideale Darstellung von Gruppen von Metadatentemplates im Frontend in Registerkarten besteht. Dementsprechend zeigt Abbildung 20 ein Konzept für die Gliederung von Templates mittels Registerkarten. Es wird vorgeschlagen, nicht jede der Registerkarten als separate Seite zu erstellen, sondern das Scrollen zu verwenden. Das bedeutet, dass alle Metadatengruppen auf einer Seite zu finden sind, der Benutzer aber mit Hilfe von Registerkarten durch sie navigieren (scrollen) kann, da die Position jeder Templates-Gruppe mit einer bestimmten Registerkarte verknüpft ist. Diese Registerkarten verschwinden nicht, sondern bewegen sich gleichzeitig mit dem Scrollen weiter, so dass der Benutzer die Registerkarten von jeder Stelle der Seite aus wechseln kann. Ein ähnliches Konzept wird auch bei AWS Marketplace und Advaneo verwendet, nur gibt es bei Advaneo weniger Registerkarten, kein Scrollen (die Registerkarten befinden sich auf verschiedenen Seiten) und die Registerkarten verschwinden, wenn der Benutzer auf der Seite nach unten scrollt, was bedeutet, dass der Benutzer manuell nach oben gehen muss, um die Registerkarte zu ändern. Dies könnte unpraktisch sein, wenn die Seite mit den Metadaten lang ist. Es sollte auch beachtet werden, dass ein Link neben den Rating-Sternen anstelle einer

[45] Balsamiq Wireframes: https://balsamiq.com/wireframes

zusätzlichen Registerkarte verwendet wird, um zu den Bewertungen zu navigieren. Damit soll Redundanz vermieden werden.

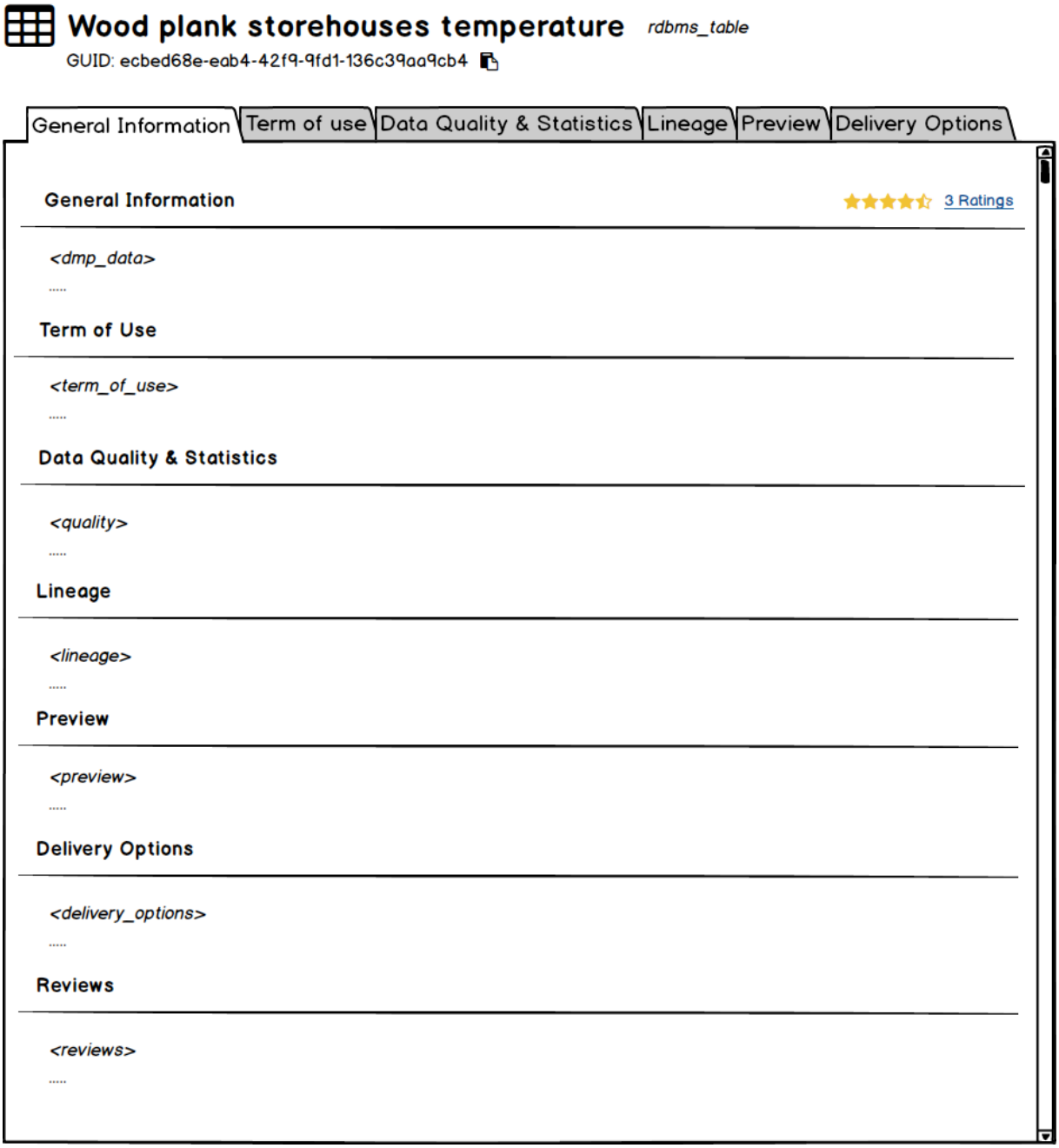


Abbildung 20. Entwurf der Darstellung von Metadaten-Gruppen mittels Registerkarten

9.2 Darstellung verschiedener Elemente der Detailseite

Nachdem das Grundkonzept der Gruppierung von Templates erläutert wurde, lassen sich nun Typen von Elementen zur Darstellung von Metadaten in den verschiedenen Bereichen (Registerkarten) der Detailseite erkennen. Das einfachste Element, das auch am häufigsten vorkommt, ist das *Textelement*, von dem ein kleines Beispiel in Abbildung 21 zu sehen ist. Hier ist zu erkennen, dass auf den Attributnamen der Wert nach einem Doppelpunkt folgt.

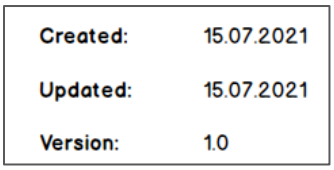


Abbildung 21. *Textelement* der Detailseite

Das zweite Element, das die eingebauten Templates präsentiert, ist das *Aufklappelement*. Dieses Element sieht aus wie das in Abbildung 22 für die Attribute *Location* und *Category* gezeigte. Ein Klick auf dieses Element lässt einen zusätzlichen Container erscheinen, der die Metadaten der eingebauten Templates enthält. Solche Aufklappelemente werden in mehreren Bereichen der Detailseite verwendet, eine detaillierte Darstellung findet sich in Anhang 2 dieser Arbeit. Neben dem Bereich „General Information“ (Mockups 1 – 3), wo, wie bekannt, die Aufklappelemente die Attribute *Location* und *Category* umfassen, befinden sich hier auch der Bereich „Term of use“ (Mockups 4, 5) mit dem aufklappbaren Element für das Attribut *Owner* sowie der Bereich „Delivery Oprions“ (Mockups 9, 10), in dem das Attribut *Access Method* aufgeklappt werden kann.

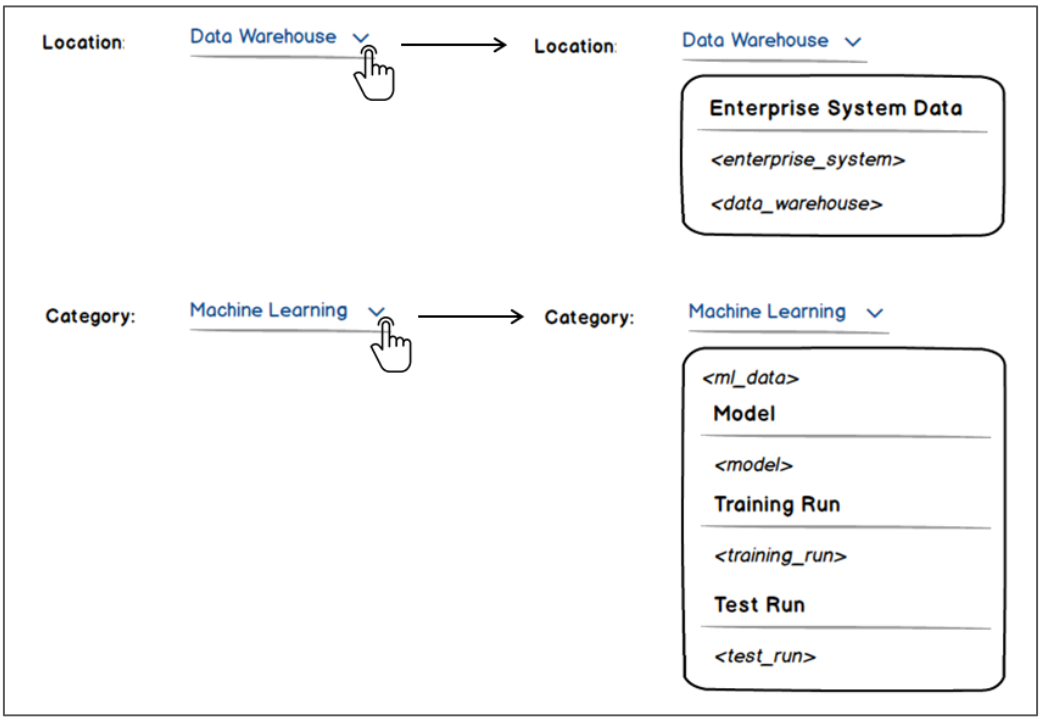


Abbildung 22. *Aufklappelemente* der Detailseite

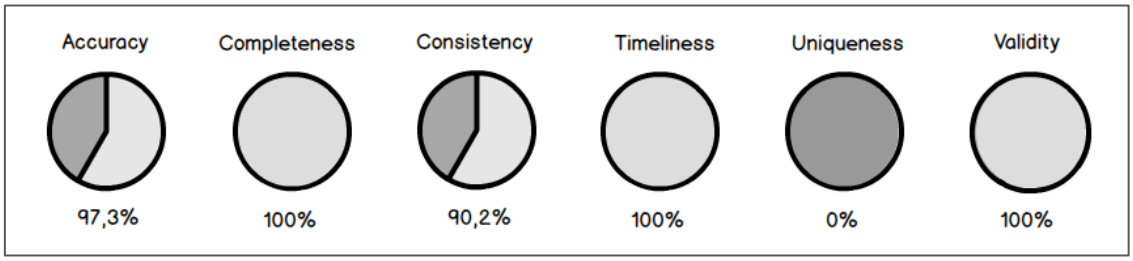


Abbildung 23. *Pie-Chart-Diagramme* zur Darstellung der Datenqualität auf der Detailseite

Das nächste Element besteht aus einem *Pie-Chart-Diagramm*, das zur Darstellung der Datenqualität verwendet wird (*<classic_metrics>*-Template). Ein einfaches Beispiel für derartige Pie-Chart-Diagramme für sechs Qualitätsmetriken ist in Abbildung 23 wiedergegeben.

Ein weiteres Element ist eine *Tabelle*. Sie ist dazu gedacht, Profiling-Daten oder andere zusätzliche Informationen über Datensätze anzuzeigen, die Spalten enthalten (z.B. eine Liste aller Spalten oder eine Liste der Spalten Bedeutungen). Ein Beispiel für eine solche Tabelle mit Profiling-Daten, die Informationen über die Anzahl der Nullwerte in der Tabelle enthalten, ist in Abbildung 24 dargestellt. Die Tabelle kann auch zur Vorschau von Daten verwendet werden, die in Tabellenform vorliegen. Ein Überblick über den Bereich „Preview" auf der Detailseite ist in Mockup 8 gezeigt.

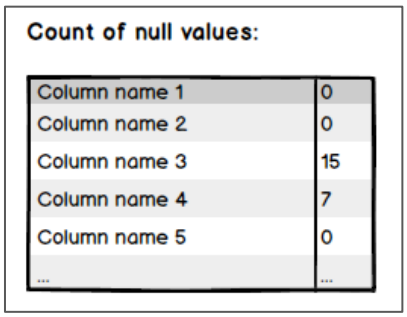


Abbildung 24. Element *Tabelle* der Detailseite

Die Datenqualität, die durch die Pie-Chart-Diagramme dargestellt wird, und die Tabellen zu den Profiling-Daten, sind im Bereich „Data Quality & Statistik" in Mockup 6 genauer dargelegt. Das Mockup ist für einen Datensatz vom Typ *rdbms_table* konzipiert. Je nach Typ des Datensatzes können hier auch Metadaten der Templates *<resolution>* oder *<signal_quality>* erfasst werden.

Die Daten Lineage ist als *Graph-Element* vorgesehen, wobei die Datensätze, aus denen der aktuelle Datensatz gewonnen wurde, durch Knoten und die Prozesse durch Kanten dargestellt werden, wie in Abbildung 25 zu sehen ist. Ein Umriss des Bereichs „Lineage" der Detailseite ist in Mockup 7 abgebildet.

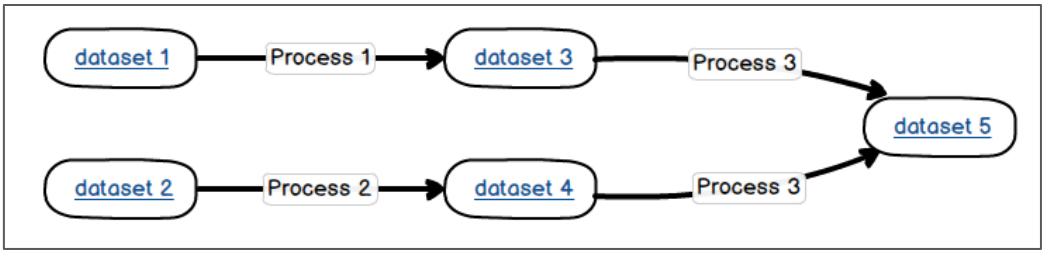


Abbildung 25. *Graph-Element* der Detailseite, das zur Darstellung der Lineage verwendet wird

Obwohl das Review nicht als eigenständiges Element bezeichnet werden kann, muss sie dennoch gesondert betrachtet werden, da sie nicht nur Text enthält. Es wird vorgeschlagen,

die klassischen fünf Sterne zu verwenden, um die Bewertung abzugeben, wie bei Amazon[46] oder Ebay[47]. Eine einzelne Bewertung ist also ein Container, der die Attribute des *<review>*-Templates enthält, wobei das Attribut *rating* durch die Sterne repräsentiert wird. Die vollständige Seite mit mehreren Bewertungen ist in Mockup 11 zu finden.

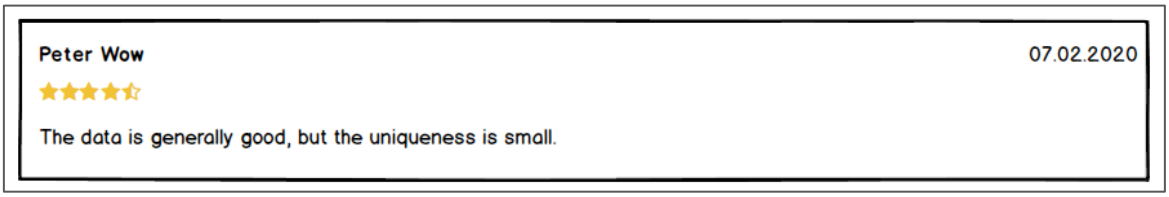


Abbildung 26. Datensatz-Review, in dem das Attribut *rating* durch Sterne abgebildet ist

Alle anderen Templates, die in diesem Kapitel nicht explizit genannt wurden, werden mit dem Textelement visualisiert. Nachdem nun das Konzept der Darstellung von Metadaten auf der Detailseite erklärt wurde, geht es weiter mit der Umsetzung der in diesem Kapitel vorgestellten Elemente sowie der in Mockups gezeigten Detailseite-Bereiche.

[46] Amazon: https://www.amazon.de
[47] Ebay: https://www.ebay.de

10 Implementierung

Um zu zeigen, wie Metadaten aus Templates umgesetzt und im Frontend dargestellt werden können, wird zunächst der bestehende Datenmarktplatz-Prototyp betrachtet, für den die Implementierung durchgeführt werden soll, und dann genau gezeigt, wie die verschiedenen Elemente im Frontend angezeigt werden können. Ein weiterer wichtiger Aspekt der Implementierung umfasst die Simulation der Verwendung eines Metadaten-Repositorys, die ebenfalls in diesem Kapitel behandelt wird.

10.1 Prototyp des Datenmarktplatzes

Im Rahmen eines vom Institut für Parallele und Verteilte Systeme (IPVS) angebotenen Studienprojekts, wurde ein Datenmarktplatz-Prototyp entwickelt. Für die prototypische Umsetzung der internen Datenmarktplatzplattform wurde eine einfache Unternehmensumgebung bestehend aus einem Data Lake und mehreren Datenbanken, in denen Unternehmensdaten gespeichert sind, implementiert. Der Prototyp des Backends besteht aus Microservices und wurde mit *Java* und *Spring Boot* geschrieben, das Frontend mit *Angular* und *CoreUI*. Die Interaktion zwischen dem Backend und dem Frontend erfolgt über REST-APIs.

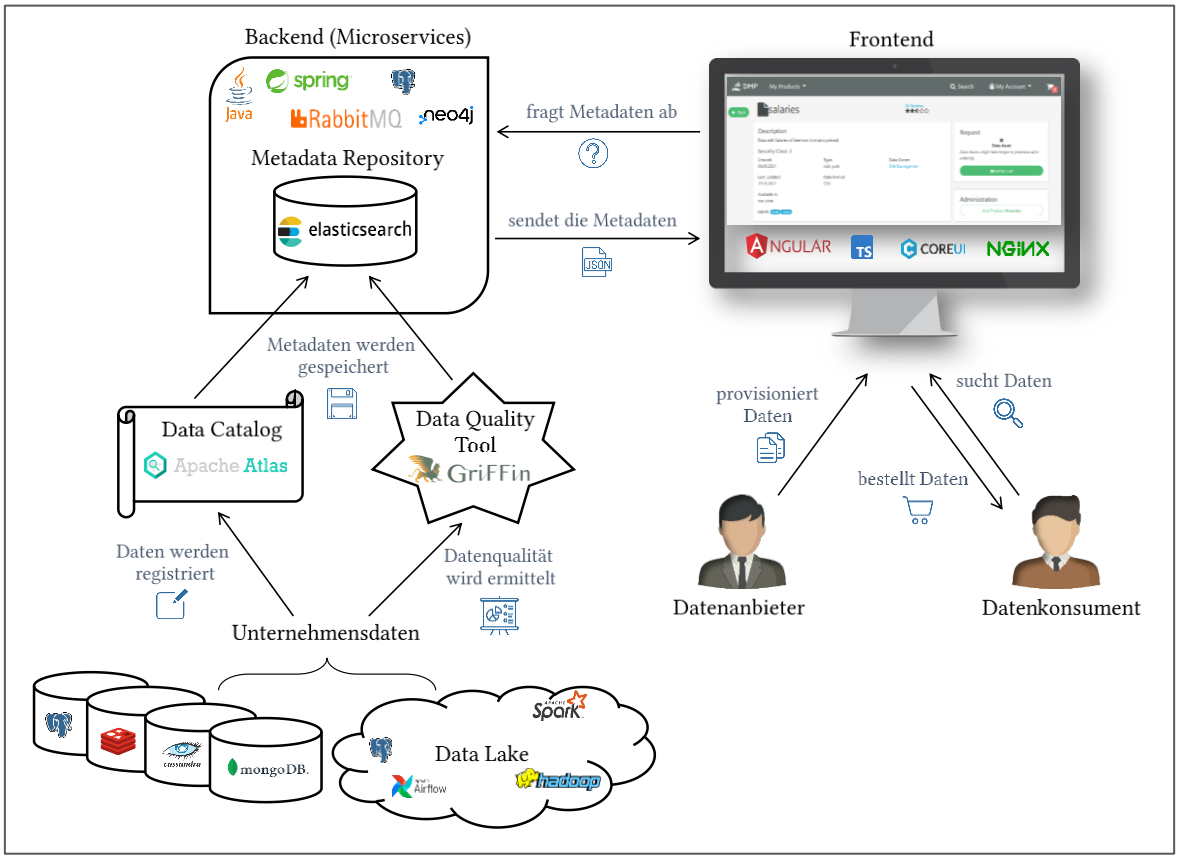


Abbildung 27. Schematische Darstellung des Datenmarktplatz-Prototyps

Der Prototyp ermöglicht die Provisionierung, Suche und Bestellung von Daten. Die Datensätze werden anhand der im Apache-Atlas-Katalog gespeicherten Metadaten durchsucht und mit Informationen über sie versehen. D.h., Informationen über die Datensätze

werden vom Frontend angefordert, die Anfrage wird vom Backend verarbeitet und es werden die Metadaten von Atlas zurückgegeben, die auf der Detailseite des Datensatzes dargestellt werden. Das bedeutet, dass die Detailseite des Datensatzes die Seite ist, die die aus dem Backend abgeleiteten Metadaten präsentiert und somit dem Datenkonsumenten Informationen zur Verfügung stellt, auf deren Grundlage er entscheiden kann, ob der Datensatz für seinen Anwendungsfall geeignet ist.

Um verschiedene Metadatenmanagement-Tools modular mit dem Datenmarktplatz zu verbinden, wurde der Datenmarktplatz-Prototyp im Rahmen des Forschungsprojekts [74] modifiziert. Im Gegensatz zur vorherigen Version des Prototyps wurde hier ein Metadaten-Repository integriert und zusätzlich Apache Griffin zur Bestimmung der Datenqualität angeschlossen. Der Datenkatalog ist nach wie vor Apache Atlas, aber das gesamte Backend ist so umgebaut, dass die Metadaten aus Atlas und Griffin zunächst mit Hilfe von Indexern im Metadaten-Repository gespeichert, von dort gelesen und an das Frontend zur Darstellung auf der Detailseite zurückgegeben werden können. Abbildung 27[48] stellt eine schematische Darstellung des modifizierten Prototyps dar, die alle verwendeten Technologien zeigt. Elasticsearch[49] wird für das Metadaten-Repository verwendet. [74]

Um die Darstellung von Metadaten aus Templates im Frontend, d.h. auf der Detailseite, zu demonstrieren, wurde Szenario 1 aus Kapitel 3 prototypisch umgesetzt. Die Verifizierung der Umsetzung von Szenarien wird im nächsten Kapitel vorgestellt, in diesem Kapitel soll jedoch ein Blick auf die technischen Aspekte der Implementierung geworfen werden.

10.2 Backendseitige Umsetzung

Im Rahmen des bereits erwähnten Forschungsprojekts [74] wurde die Verwendung von Metadatentemplates vorgeschlagen, um das Frontend dynamischer zu gestalten. In dieser Arbeit wird das Konzept der Templates entwickelt und umgesetzt, worauf im Folgenden detaillierter eingegangen wird.

Das genannte Konzept besteht darin, dass das Frontend vom Backend Informationen über die einzelnen Attribute erhält (d.h. über die Metadaten aus Templates) und sie auf der Grundlage dieser Informationen auf der Detailseite anzeigt. Hierfür wird die Datei *default.json* verwendet, die für jedes der Attribute Informationen über den Elementtyp, seinen Namen und seine Position im Frontend enthält. Abbildung 28 zeigt die Struktur der *default.json*-Datei und Abbildung 29 enthält die Beispieleinträge für die Attribute *create_date*, *modify_date*, *version* und *location* des allgemeinen <*dmp_data*>-Templates. So ist beispielsweise „create_date“ der Name des Attributs, der als Key dient und „text“ steht für den Typ der Komponente, mit der das Attribut im Datenmarktplatz-Frontend dargestellt wird. Der nächste Eintrag „Created“ ist der Titel des Attributes, der im Frontend angezeigt wird, während „body-111“ seine Position im Frontend bezeichnet. Somit liegt die Definition der Templates selbst auf der Backendseite und ihre Implementierung erfolgt direkt im Frontend, worauf nachfolgend näher eingegangen wird.

[48] Die Icons für die Abbildung sind der Website https://de.freepik.com entnommen
[49] Elasticsearch: https://www.elastic.co

```
{
  "layout": "default_details",
  "fields": {
    "attribute_1": {
      "component": "Type of Attribute 1",
      "title": "Title of Attribute 1",
      "position": "Position of Attribute 1 in Frontend"
    },
    "attribute_2": {
      "component": "Type of Attribute 2",
      "title": "Title of Attribute 2",
      "position": "Position of Attribute 2 in Frontend"
    }
  }
}
```

Abbildung 28. Template-Layout auf der Backendseite, präsentiert in der *default.json*-Datei

```
"create_date": {
  "component": "text",
  "title": "Created",
  "position": "body-111"
},
"modify_date": {
  "component": "text",
  "title": "Updated",
  "position": "body-111"
},
"version": {
  "component": "text",
  "title": "Version",
  "position": "body-111"
},
"location": {
  "component": "collapsable_element",
  "title": "Location",
  "position": "body-111"
}  ...
```

Abbildung 29. Beispiel für die Beschreibung der Attribute *create_date*, *modify_date*, *version* und *location* auf der Backendseite

10.3 Frontendseitige Umsetzung

Nachdem nun bekannt ist, wie die Metadatentemplates auf der Backendseite aussehen, ist es an der Zeit, zum Frontend-Teil überzugehen, denn hier erfolgt die Interaktion mit dem Benutzer und hier werden die Metadaten direkt angezeigt, was dem Benutzer helfen soll, die Datensätze zu verstehen und auszuwählen.

10.3.1 Verwendung eines Demo-Metadaten-Repositorys

Bevor mit der Implementierung des Frontends begonnen wird, müssen die Metadaten, die im Frontend angezeigt werden sollen, vorbereitet werden. Da derzeit nur zwei Tools mit dem Datenmarktplatz-Prototyp verbunden sind (Apache Atlas und Apache Griffin), können nicht

alle in den Templates gesammelten Metadaten automatisch erfasst werden. Um jedoch das Konzept der Metadaten-Darstellung aus Templates auf der Detailseite darstellen zu können, wird *JSON-Server*[50] eingesetzt, der die Erstellung und Abfrage von JSON-Einträgen ermöglicht und damit die Antwort des Backends simuliert. Alternative Metadatenformate könnten XML oder RDF sein, aber JSON wurde gewählt, weil es das Format ist, in dem die Metadaten an die mit dem Datenmarktplatz-Prototyp verbundenen Tools geliefert werden. Darüber hinaus wird dieses Format häufig zur Beschreibung von Metadaten verwendet, wie in Kapitel 5 erläutert. Außerdem erfolgt die Backend-Antwort über die REST-API ebenso im JSON-Format.


```
{
  "result": [
    {
      "id": "3d46a5e0-442c-48e9-93f0-60231c2e187b",
      "properties": {
        "display_name": "Wood plank storehouses data",
        "type": "collection",
        "...": "..."
      }
    },
    {
      "id": "9726b357-ff19-415b-bb4e-bc036edb81fd",
      "properties": {
        "display_name": "Wood plank storehouses humidity",
        "type": "rdbms_table",
        "...": "..."
      }
    },
    {
      "...": "..."
    }
  ]
}
```


Abbildung 30. Struktur der *response_example.json*-Datei

Abbildung 30 verdeutlicht die Struktur der Datei *response_example.json*, bei der es sich um die Demo-Einträge in Elasticsearch handelt, d.h. in ein Metadaten-Repository, das zur Speicherung von Metadaten verwendet wird. Mit dem bereits erwähnten JSON-Server können diese Daten abgefragt werden, als ob es um ein echtes Elasticsearch-Metadaten-Repository ginge. Anschließend lässt sich darüber diskutieren, wie verschiedene Templates-Attribute im Frontend dargestellt werden können, die aus diesem Repository stammen und auf der Detailseite angezeigt werden.

10.3.2 Darstellung der verschiedenen Elemente im Frontend

Wie aus Abschnitt 10.2 bekannt, hat jedes der Template-Attribute seinen eigenen Typ, der in der Datei *default.json* als *component* angegeben ist. Dieser Komponententyp kann einer der in

[50] JSON Server: https://www.npmjs.com/package/json-server

Abschnitt 9.2 besprochenen Typen sein. Im Folgenden wird ausführlich auf die Darstellung dieser Elemente im Frontend eingegangen.

Das Textelement ist einfach und wird durch den HTML-Tag ** dargestellt, so dass es nicht im Detail erläutert werden muss.

Das zweite Element ist ein Aufklappelement. Abbildung 31 zeigt die Serverantwort und die Implementierung des Aufklappelements im Frontend. Hier wird ein einfacher Link verwendet, der, wenn er angeklickt wird, einen zusätzlichen Container mit Metadaten aufklappt. Nebenbei sei auch die zweite Attributeigenschaft, die in der Datei *default.json* für jedes der Attribute angegeben wird, erwähnt, nämlich *title*, d.h. der Name des Attributs. In Abbildung 31 ist es z.B. *owner*.

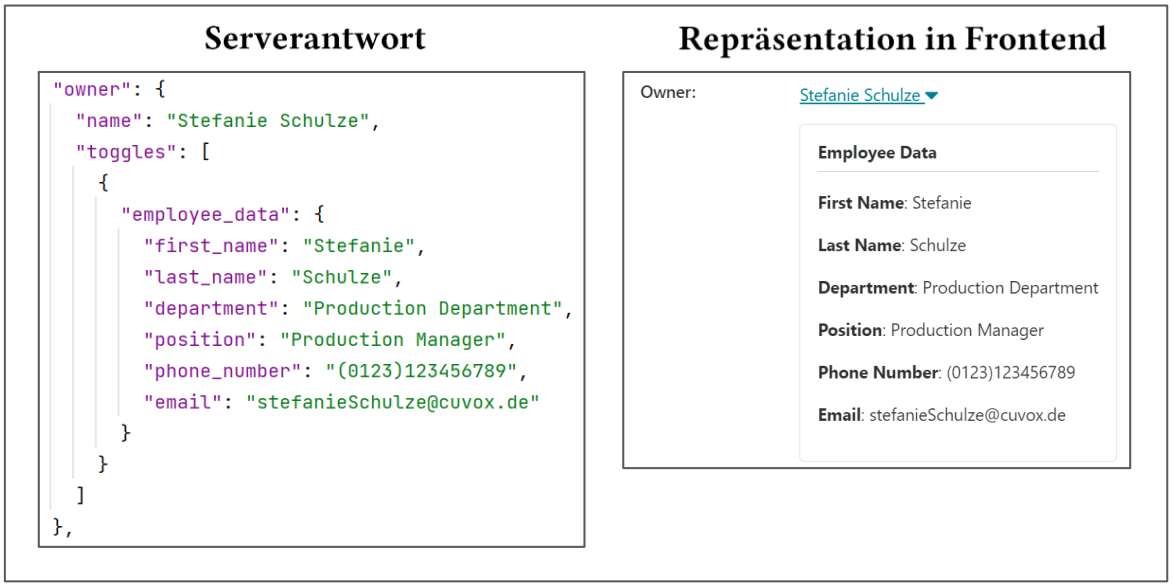


Abbildung 31. Darstellung des Aufklappelements im Frontend

Zur Implementierung des Pie-Chart-Diagramms wird *canvas baseChart* verwendet. Der HTML-Codeausschnitt ist in Abbildung 32 dargestellt, und Abbildung 33 zeigt die Serverantwort sowie ihre Wiedergabe im Frontend. D.h., für jede der Datenqualitätsmetriken kommt ein Wert vom Server, dessen Variable im Frontend als *value* bezeichnet und dann als Pie-Chart visualisiert wird.

```
<div style="display: block; margin-left: -95px;">
  <canvas baseChart
          [data]="[100 - value, value]"
          [colors]="pieChartColors"
          [chartType]="'pie'"
          [options]="pieChartOptions">
  </canvas>
  <div style="font-size: 14px; text-align: center;">{{value.toFixed( fractionDigits: 0)}}%</div>
</div>
```

Abbildung 32. HTML-Codefragment zur Implementierung des Pie-Chart-Diagramms

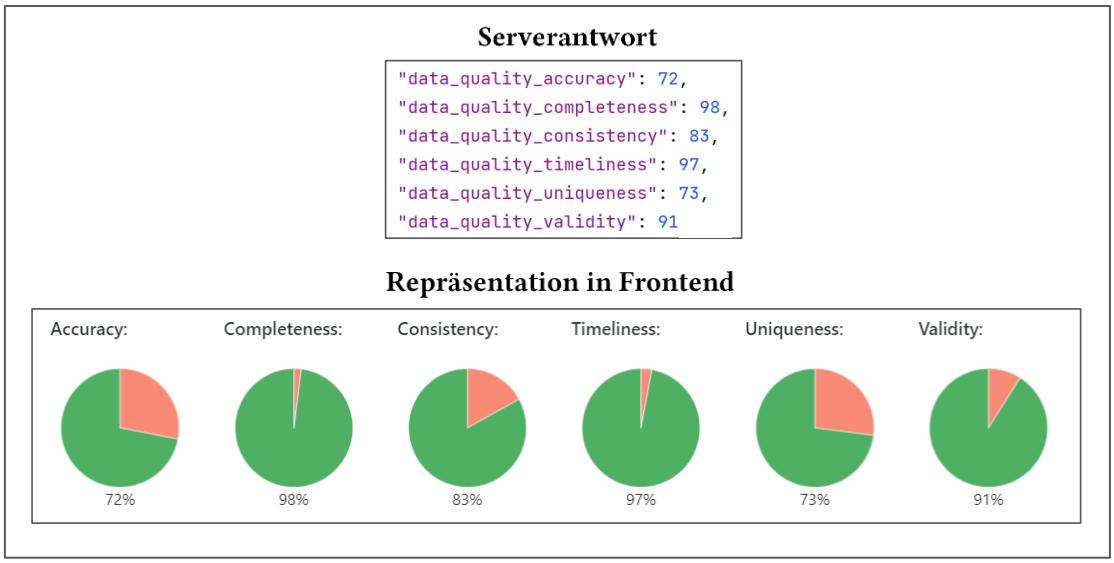


Abbildung 33. Darstellung der Pie-Chart-Diagramme im Frontend

Die Repräsentation des nächsten Elements, einer Tabelle, ist in Abbildung 34 dargestellt. Die Serverantwort enthält die Namen und Werte der Spalten, die im Frontend in eine Tabelle umgewandelt werden.

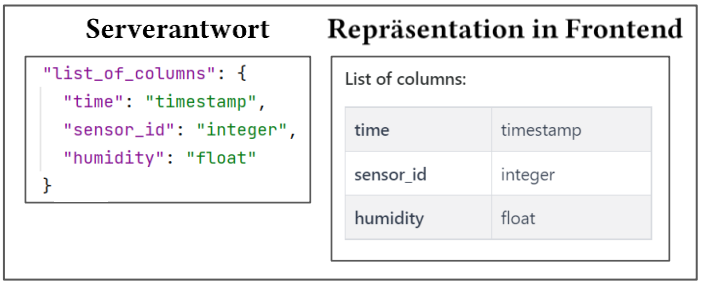


Abbildung 34. Darstellung der Tabelle im Frontend

Das letzte Element, der Graph, wird im Frontend mit der Angular-Bibliothek *ngx-graph*[51] implementiert. Ein Auszug aus dem HTML-Code ist in Abbildung 35 zu sehen, wobei die Variable *value.processes* ein Array von Prozessen und die Variable *value.assets* ein Array von Datensätzen abliest. Wie in Kapitel 9 vorgeschlagen, sind die Datensätze als Knoten und die Prozesse als Kanten dargestellt (Abbildung 36).

```
<ngx-graph *ngIf="value.processes.length != 0"
           class="chart-container"
           [draggingEnabled]="false"
           [panningEnabled]="false"
           [enableZoom]="false"
           [links]="value.processes"
           [nodes]="value.assets">
</ngx-graph>
```

Abbildung 35. HTML-Codefragment zur Implementierung des Graphs

[51] ngx-graph: https://swimlane.github.io/ngx-graph/

Abbildung 36. Darstellung des Graphen im Frontend

Nachdem die verschiedenen Elemente betrachtet wurden, sollte auch gezeigt werden, wie im Frontend bestimmt wird, welches Element verwendet werden soll. Abbildung 37 zeigt, wie die Elementtypen im Backend definiert werden (links) sowie einen Codeausschnitt, der die Zuordnung mittels *ngSwitch* in Abhängigkeit von dem Typ des Elements im Frontend verdeutlicht (rechts). Der Name *pipeline* bezieht sich auf den Graphen.

```
"format": {
  "component": "text",
  "title": "Format",
  "position": "body-121"
},
"data_quality_accuracy": {
  "component": "pieChart",
  "title": "Accuracy",
  "position": "body-312"
},
"list_of_columns": {
  "component": "table",
  "title": "List of columns",
  "position": "body-313"
},
"lineage": {
  "component": "pipeline",
  "title": "Lineage",
  "position": "body-411"
},
"owner": {
  "component": "collapsable_element",
  "title": "Owner",
  "position": "body-211"
}
```

```
<div>
  <div [ngSwitch]="type">
    <dmp-text *ngSwitchCase="'text'"
                  [value]="value"
                  [fieldName]="fieldName">
    </dmp-text>
    <dmp-pie-chart *ngSwitchCase="'pieChart'"
                  [value]="value"
                  [fieldName]="fieldName">
    </dmp-pie-chart>
    <dmp-table *ngSwitchCase="'table'"
              [value]="value"
              [fieldName]="fieldName">
    </dmp-table>
    <dmp-pipeline *ngSwitchCase="'pipeline'"
                 [value]="value"
                 [fieldName]="fieldName">
    </dmp-pipeline>
    <dmp-collapsable-element *ngSwitchCase="'collapsable_element'"
                             [value]="value"
                             [fieldName]="fieldName">
    </dmp-collapsable-element>
  </div>
</div>
```

Abbildung 37. Definition der Elementtypen im Backend (links) und die Auswahl von Elementen in Abhängigkeit von ihrem Typ im Frontend (rechts)

10.3.3 Anordnung der Elemente im Frontend

Die letzte Attributeigenschaft, die in *default.json* auf der Backendseite angegeben ist, ist die *position* des Elements. Um alle in den Templates gesammelten Metadaten in die in Kapitel 9 vorgeschlagenen Gruppen einzuordnen, wird die folgende Bezeichnung der Attributpositionen vorgeschlagen: jede Position besteht aus drei Ziffern, wie in Abbildung 38 dargestellt. Die erste Ziffer bezeichnet den *Bereich* (Registerkarte) der Detailseite, die zweite Ziffer steht für die *Spaltennummer* und die dritte für die *Zeilennummer*, in der die Elemente platziert werden sollen. Ein detaillierteres Layout für die gesamte Detailseite ist in Anhang 2 in Mockup 10 zu sehen. Für jeden Bereich kann das Frontend dynamisch mehrere Elemente laden.

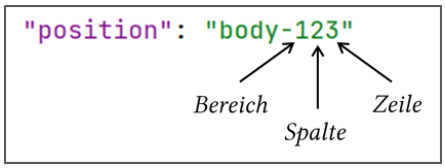


Abbildung 38. Erklärung der vom Backend übermittelten Positionsbezeichnung

Wie in Abbildung 39 gezeigt, wird diese Position direkt im Frontend definiert, wobei die beschriebene dreistellige Bezeichnung der Position angegeben wird.

```
<div class="row">
  <div class="col-6">
    <dmp-layout-pos name="body-111" [asset]="asset" ... ></dmp-layout-pos>
  </div>
  <div class="col-6">
    <dmp-layout-pos name="body-121" [asset]="asset" ... ></dmp-layout-pos>
  </div>
</div>
```

Abbildung 39. Angabe der Positionsbezeichnung auf der Frontseite

Nachdem nun die grundlegenden Aspekte der prototypischen Implementierung der Darstellung der Metadaten aus Templates auf der Datenmarktplatz-Detailseite abgedeckt sind, sollte geprüft werden, ob damit alle Anforderungen an den Datenmarktplatz erfüllt sind, was im nächsten Kapitel durchgeführt wird.

11 Evaluation

Um zu beurteilen, ob die Ziele dieser Arbeit erreicht werden, muss sichergestellt werden, dass die entwickelten Metadatentemplates zum Verständnis der Daten beitragen, wofür es notwendig ist, die Ergebnisse zu evaluieren. Die Evaluierung erfolgt in zwei Schritten: zunächst werden die in Kapitel 3 beschriebenen Szenarien betrachtet und anschließend die Anforderungen an den Datenmarktplatz hinsichtlich der Verwendung von Metadaten aus Kapitel 4 verifiziert.

11.1 Szenarienbasierte Evaluation

Um die erfüllten Aufgaben der Sammlung und Strukturierung von Metadaten für den internen Datenmarktplatz mittels Templates zu demonstrieren, ist es notwendig, an die zuvor entworfenen Szenarien zu erinnern und zu zeigen, dass technische und nicht-technische Nutzer des Datenmarktplatzes die von ihnen benötigten Datensätze finden und verstehen können.

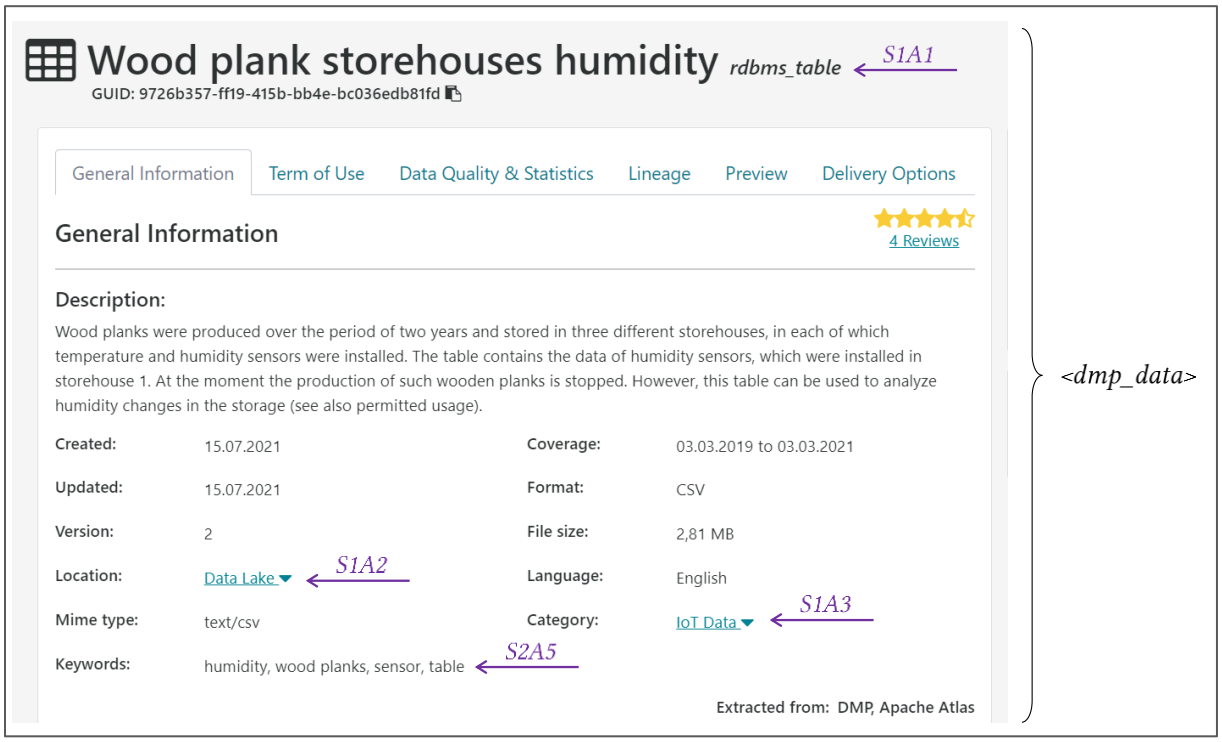


Abbildung 40. Abschnitt der Detailseite des Datensatzes vom Typ *rdbms_table*, der die Metadaten des *<dmp_data>*-Templates enthält

In Szenario 1 möchte ein technischer Benutzer, nämlich ein Datenanalyst, Datensätze mit Feuchtigkeits- und Temperatursensordaten für Lagerräume finden, in denen Holzleisten für Fußböden gelagert wurden. Angenommen, der Nutzer gibt die Suchanfrage „wood plank“ ein, wie er es wünscht, und findet mehrere Tabellen und Bilder. Anschließend lässt sich die Metadatenstruktur dieser Ergebnisse genauer betrachten. Abbildung 40 zeigt einen Screenshot der Detailseite der Tabelle, die die Daten des Feuchtigkeitssensors enthält. Die Pfeile in der Abbildung zeigen die Anforderungen aus den Szenarien, deren Erfüllung dem einen oder anderen Bereich der Detailseite zugeordnet ist. Oben wird der Name der Tabelle

angezeigt, rechts daneben der Datensatztyp. Damit ist die *S1A1*-Anforderung erfüllt, die sich auf den Datentyp bezieht und bei der der Datenanalyst Daten in Form einer Tabelle bevorzugt.

Es folgen, wie in Kapitel 9 gezeigt, die Registerkarten mit verschiedenen Gruppen von Metadaten. In Abbildung 40 ist die Registerkarte „General Information“ aktiv, die die Metadaten des allgemeinen *<dmp_data>*-Templates enthält. Hier ist auch das Attribut *Location* zu sehen, dessen Wert der „Data Lake“ ist und bei dem es sich um ein Aufklappelement handelt. Beim Aufklappen dieses Elements (Abbildung 41) werden Informationen über das Unternehmenssystem, d.h. das Template *<enterprise_system>* und insbesondere den Data Lake angezeigt, der durch das Template *<data_lake>* dargestellt wird, das vom Template *<enterprise_system>* erbt. Anhand der Informationen über die Data Lake Zone (es ist zu erkennen, dass es sich um die Harmonized Zone handelt) kann der Benutzer schließen, dass die vor ihm liegenden Daten vorverarbeitet wurden, womit der erste Teil der *S1A2*-Anforderung abgedeckt ist, der besagt, dass die Sensordaten vorzugsweise vorverarbeitet bzw. bereinigt vorliegen sollten.

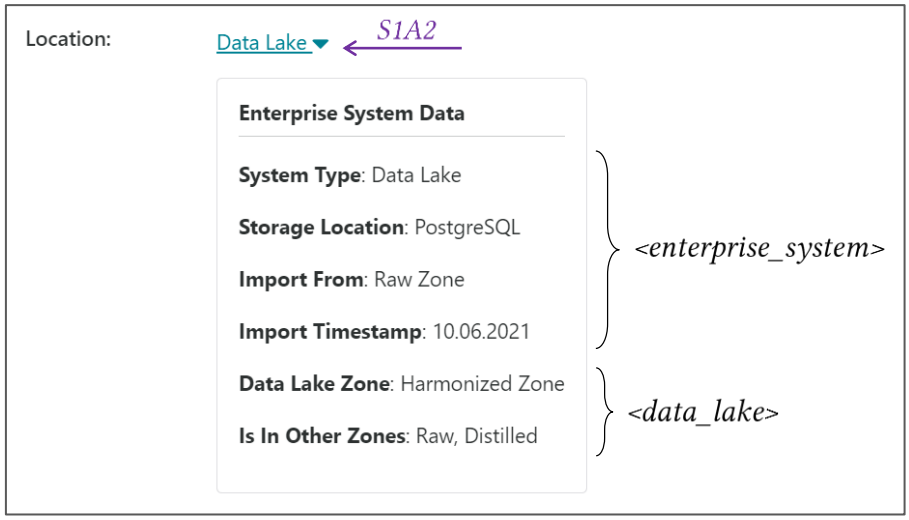


Abbildung 41. *Location*-Attribut, das aus den Metadaten der Templates *<enterprise_system>* und *<data_lake>* besteht

Wenn jetzt Szenario 2 betrachtet wird, bei dem ein nicht-technischer Benutzer, der Business-Analyst, eine Reihe von Berichten über die Interaktionen mit Kunden finden möchte, erscheint auf demselben Abschnitt der Detailseite (Abbildung 40) ein solches Attribut wie *Keywords*, mit dem der Business-Analyst eine Sammlung von Kundenbewertungen finden kann, was wiederum eine Erfüllung der Anforderung *S2A5* bedeutet, die die Keywords aufzählt, welche die Übersicht von Kundenbewertungen enthalten sollte.

Im Screenshot (Abbildung 40) ist ein weiteres Aufklappelement zu sehen, nämlich die *Category*. In diesem Fall bezieht sich der Datensatz auf die IoT-Daten-Kategorie, für die, wie aus Abschnitt 6.3.4 bekannt, ein separates *<iot_data>*-Template entwickelt wurde. In Szenario 1 möchte der Nutzer Auskunft über die Eigenschaften der Sensoren, einschließlich ihrer Modelle und Software, erhalten, um beurteilen zu können, ob die Daten für ihn geeignet sind. Wenn das *Category*-Attribut aufgeklappt wird, wird eine Liste von Metadaten über den Sensor und seine Software zur Verfügung gestellt, die auch in den Templates *<sensor>* und *<software>* enthalten sind (Abbildung 42). Damit ist die Anforderung *S1A3* erfüllt.

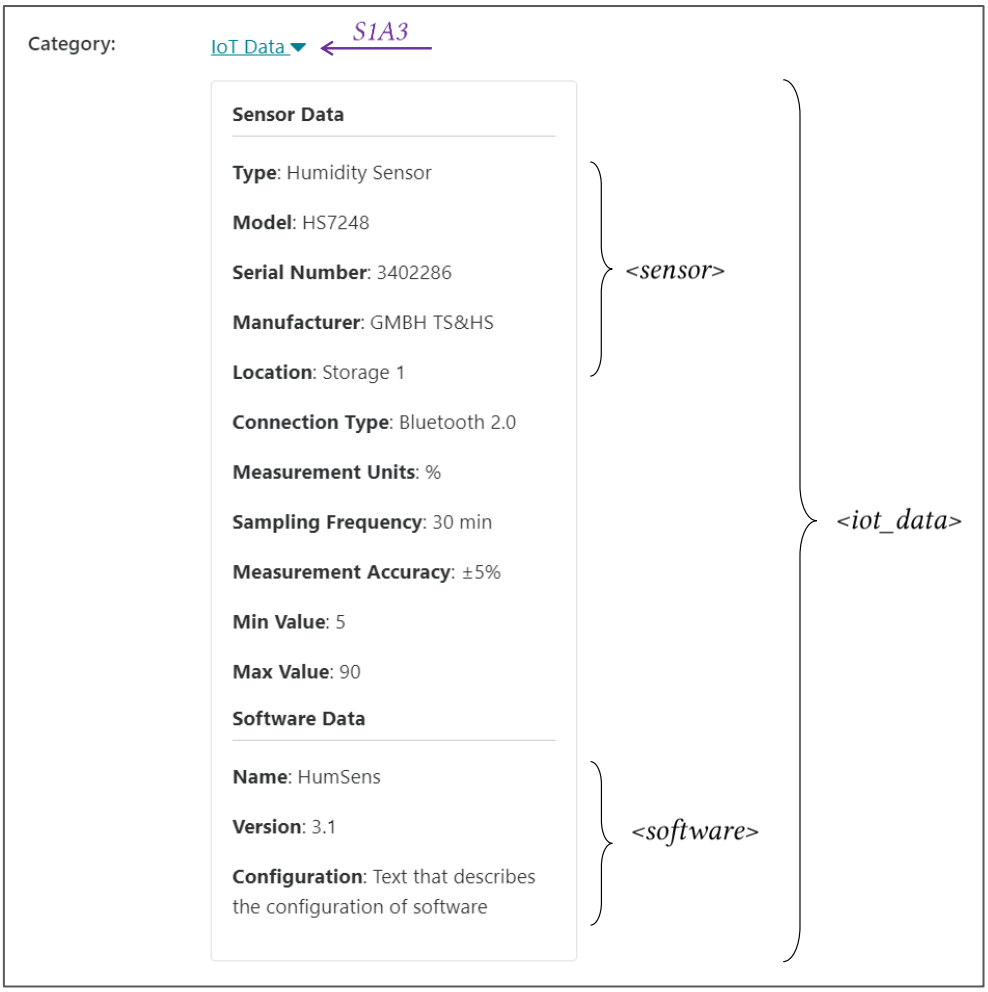


Abbildung 42. *Category*, dargestellt durch die Metadaten des *<iot_data>*-Templates

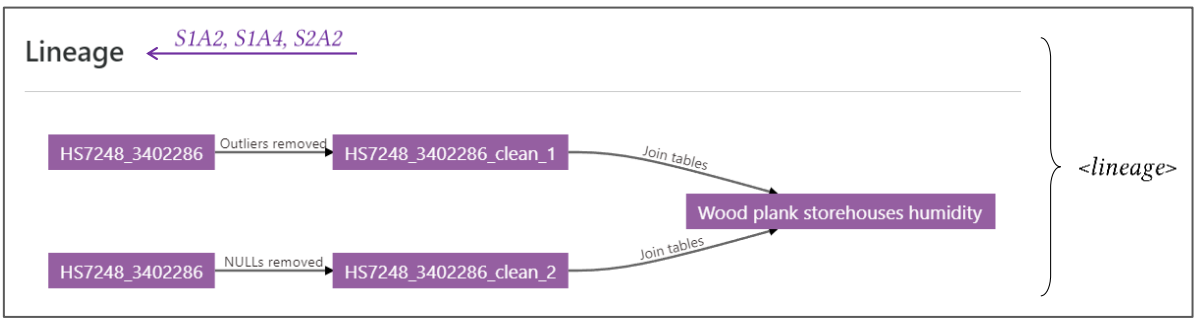


Abbildung 43. „Lineage“-Bereich der Detailseite

Eine der Registerkarten auf der Detailseite des Datensatzes ist die „Lineage“, die, wie aus Abschnitt 2.6.2 hervorgeht, einen Teil des Lebenszyklus der Daten darstellen kann und für die es das Template *<lineage>* zur Verfügung steht. Der Datenanalyst in Szenario 1 möchte nachvollziehen, wie die Daten verändert wurden (Anforderung *S1A4*). Ein Screenshot dieses Bereichs der Detailseite ist in Abbildung 43 zu sehen. In diesem Fall wird deutlich, dass die Tabelle „Wood plank storehouses humidity“ durch die Verknüpfung der beiden Tabellen „HS7248_3402286_clean_1“ und „HS7248_3402286_clean_2“ entstanden ist. Dabei ist die Tabelle „HS7248_3402286_clean_1“ das Ergebnis der Entfernung von Ausreißern aus den Sensor-Rohdaten, und die Tabelle „HS7248_3402286_clean_2“ wurde nach dem Löschen aller

Nullwerte aus denselben Sensor-Rohdaten erstellt. Wenn nun noch Szenario 2 hinzukommt, benötigt der Business-Analyst anonymisierte Daten, d.h. die Daten sollten keine Kundennamen enthalten (Anforderung *S2A2*). In diesem Fall wird die Lineage ähnlich aussehen. Damit ist der zweite Teil der Anforderung *S1A2* abgedeckt, bei dem genaue Angaben zur Vorverarbeitung bzw. Bereinigung der Daten erwünscht sind (die Anforderung *S1A2* ist nun vollständig erfüllt), ebenso wie die Anforderung *S1A4* und ein Teil der Anforderung *S2A2*.

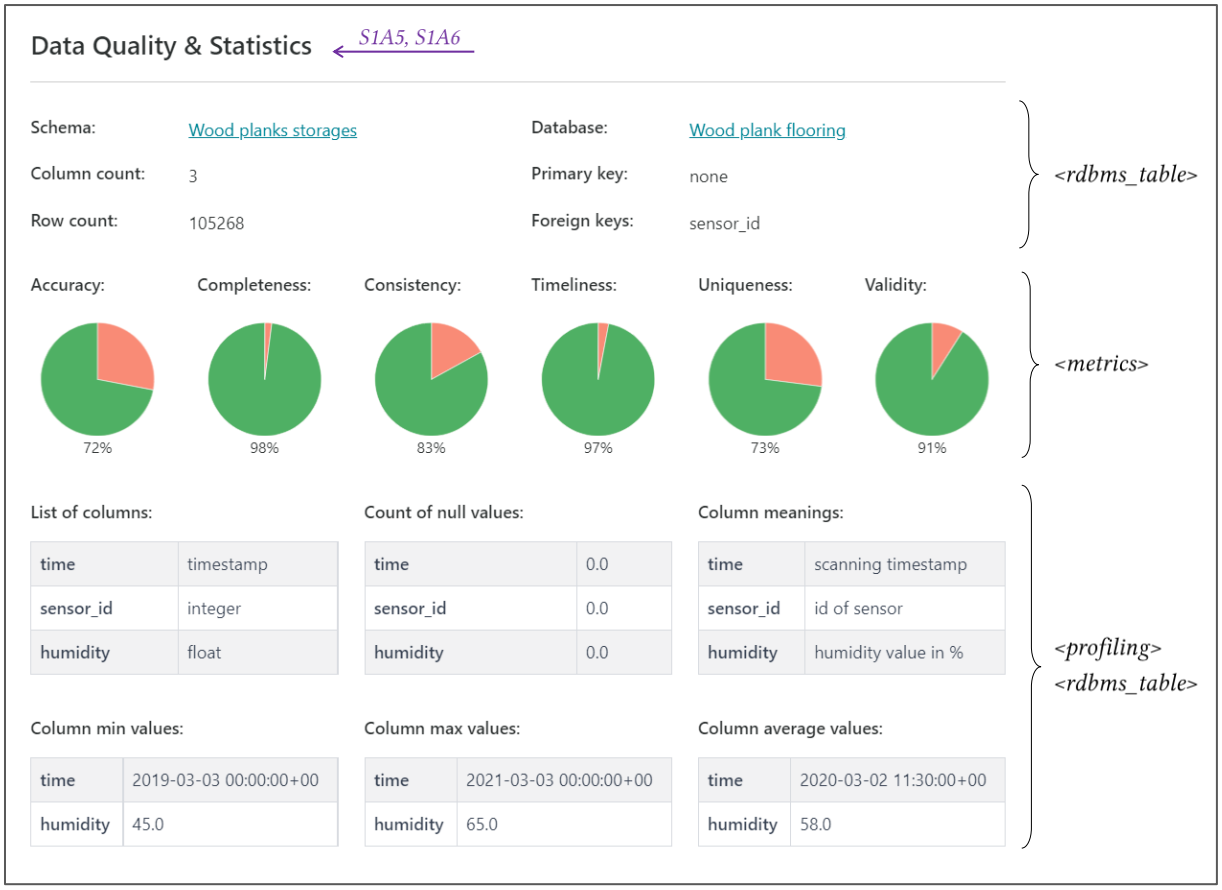


Abbildung 44. Bereich „Data Quality & Statistics“ der Detailseite

Daraufhin kann zu den Anforderungen *S1A5* und *S1A6* übergegangen werden. Anforderung *S1A5* besagt, dass der Benutzer zusätzliche Informationen über einfache Statistiken zu den Daten sehen möchte. Anforderung *S1A6* bezieht sich auf die Qualität der Daten, und zwar möchte der Datenanalyst, dass die Daten genau, vollständig und konsistent sind. Auf der Detailseite sind die „Data Quality & Statistics“ auf einer separaten Registerkarte zusammengefasst; ein Screenshot dieses Bereichs der Detailseite ist in Abbildung 44 zu finden. Hier werden grundlegende Angaben über die Tabelle angezeigt, z.B. zu welchem Schema und welcher Datenbank sie gehört, wie viele Zeilen und Spalten sie umfasst und welche Fremd- und Primärschlüssel vorhanden sind. Nachfolgend sind die sechs Qualitätsmetriken dargestellt, die in dem Template *<classic_metrics>* gesammelt wurden, von denen die ersten drei dem Benutzer in Szenario 1 helfen können, eine Auswahl zu treffen, und somit die Anforderung *S1A6* erfüllen. Das *<profiling>*-Template wird in Tabellen mit Basisstatistiken abgebildet, die sich unter den Qualitätsmetriken befinden. Zu diesen Statistiken gehören z.B.

die Spaltenliste, die Anzahl der Nullwerte, die minimalen und maximalen Spaltenwerte und andere, wodurch die Anforderung *S1A5* jetzt ebenfalls als erfüllt gilt.

Um dem Benutzer ein besseres Verständnis bezüglich des Aussehens der Daten zu vermitteln, wird eine Vorschau-Registerkarte angeboten, die einen Teil der Beispieldaten enthält, so dass der Nutzer deren Struktur und Aussehen kennenlernen kann. Im Falle von Szenario 1 ist eine Vorschau der Tabelle wünschenswert, ebenso wie Bilder von Holzleistenmängeln (Anforderung *S1A7*). Screenshots dieser Previews sind in den Abbildungen 44 und 45 abgebildet. Sofern das Template *<preview>* bekannt ist, hängt die Vorschau von dem Typ der im Satz dargestellten Daten ab. Für die Tabelle, nach der der Datenanalyst in Szenario 1 sucht, sind dies beispielsweise die ersten fünf Zeilen, und für das Bild ist es die Miniaturansicht. Daraus lässt sich schließen, dass die *S1A7*-Anforderung auf diese Weise erfüllt ist.

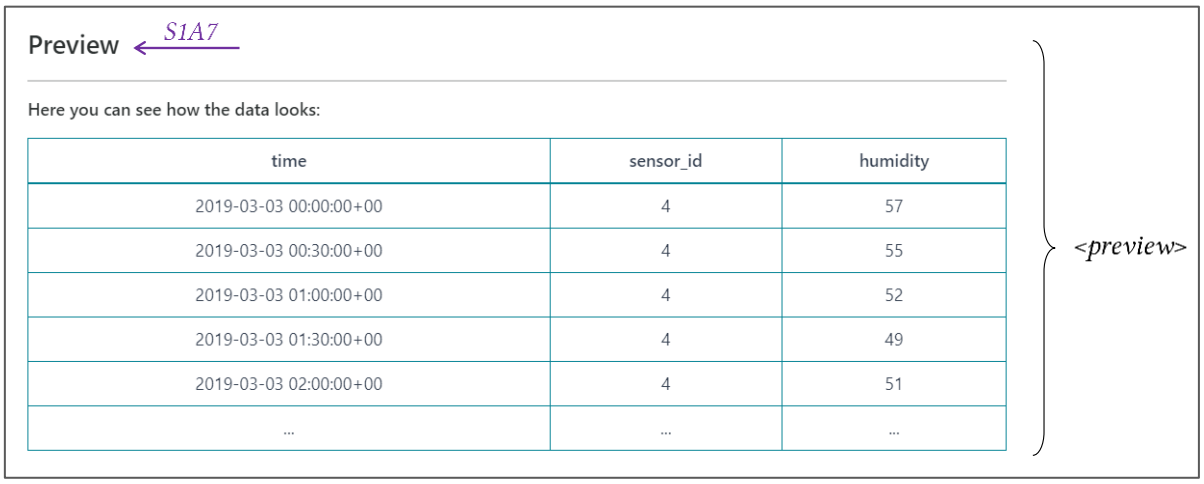


Abbildung 45. „Preview"-Bereich der Detailseite für den Datensatz von Typ *rdbms_table*

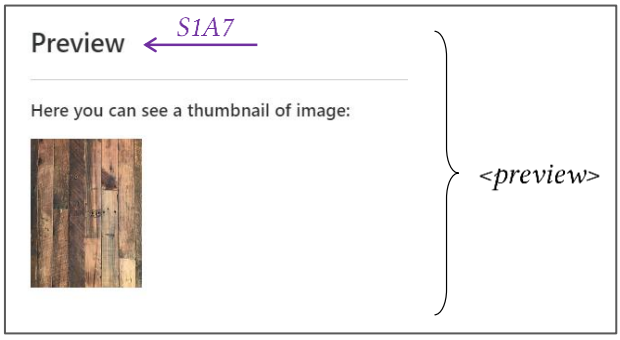


Abbildung 46. „Preview"-Bereich der Detailseite für den Datensatz von Typ *image*

Wenn der Datenmarktplatznutzer wissen möchte, wer der Data Owner ist und welche Lizenz- und Nutzungsbedingungen für die Daten gelten, kann er den Bereich „Term of use" auf der Detailseite einsehen. Wie dem Screenshot in Abbildung 47 zu entnehmen ist, werden hier Informationen über den Data Owner, die Lizenz und darüber, ob die Daten personenbezogen sind oder nicht, angezeigt. Das Attribut *License* wird als Link zu einer Seite angegeben, auf der detailliertere Daten zu den Creative Common Lizenzen zu finden sind (Abbildung 48). Bei dem Attribut *Owner* handelt es sich um ein Aufklappelement, das die Metadaten des Templates *<employee>* umfasst, d.h. die Daten des Mitarbeiters, der über die Daten verfügt.

Abbildung 49 zeigt ein solches Aufklappelement, hieraus ist z.B. zu entnehmen, dass Stefanie Schulze Mitarbeiterin der Produktionsabteilung ist. Durch diese Angaben wird die Anforderung *S1A8* eingehalten, wonach der Data Owner bekannt und vorzugsweise ein Mitarbeiter der Produktabteilung sein soll. Die Informationen über die Lizenz befriedigen zudem die Anforderung *S1A9*, die den Wunsch nach öffentlich lizenzierten Daten zum Ausdruck bringt. Darüber hinaus möchte der Business-Analyst aus Szenario 2, dass die Daten anonymisiert und nicht personenbezogen sind (Anforderung *S2A2*). Es wurde bereits erörtert, dass er Informationen darüber, ob die Daten aus anderen Daten gewonnen wurden, in dem Bereich „Lineage“ erhalten kann. In dem Bereich „Term of use“ kann der Business-Analyst auch zusätzlich sicherstellen, dass die Daten keine personenbezogenen Daten enthalten, womit ein weiterer Teil der *S2A2*-Anforderung abgedeckt ist (die *S2A2*-Anforderung ist nun ebenfalls vollständig erfüllt).

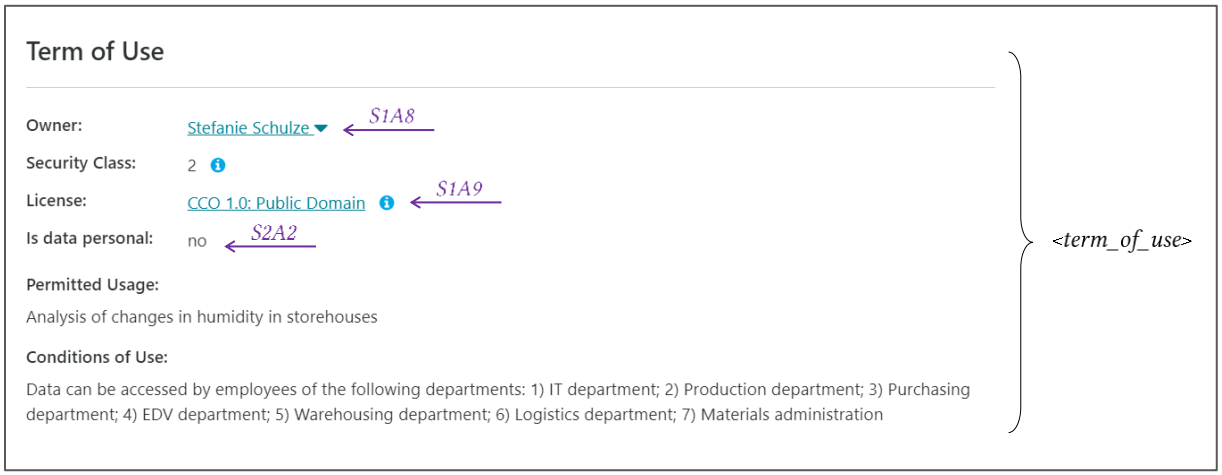
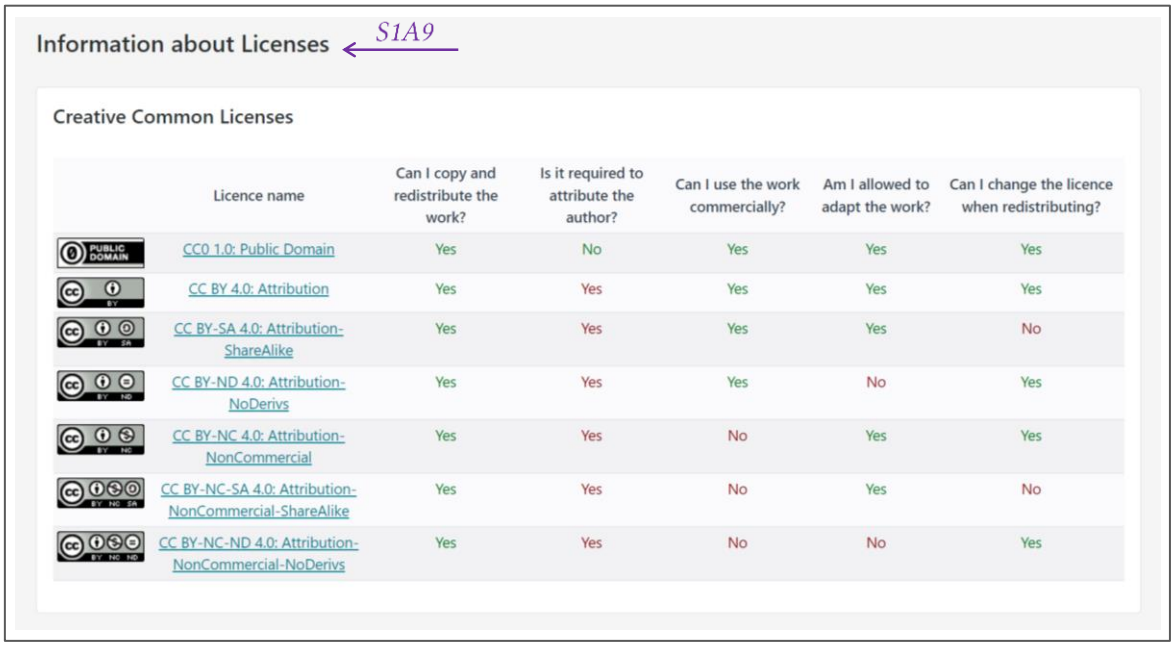


Abbildung 47. „Term of use“-Bereich der Detailseite, der das Template *<term_of_use>* abbildet



	Licence name	Can I copy and redistribute the work?	Is it required to attribute the author?	Can I use the work commercially?	Am I allowed to adapt the work?	Can I change the licence when redistributing?
	CC0 1.0: Public Domain	Yes	No	Yes	Yes	Yes
	CC BY 4.0: Attribution	Yes	Yes	Yes	Yes	Yes
	CC BY-SA 4.0: Attribution-ShareAlike	Yes	Yes	Yes	Yes	No
	CC BY-ND 4.0: Attribution-NoDerivs	Yes	Yes	Yes	No	Yes
	CC BY-NC 4.0: Attribution-NonCommercial	Yes	Yes	No	Yes	Yes
	CC BY-NC-SA 4.0: Attribution-NonCommercial-ShareAlike	Yes	Yes	No	Yes	No
	CC BY-NC-ND 4.0: Attribution-NonCommercial-NoDerivs	Yes	Yes	No	No	Yes

Abbildung 48. Informationsseite des Datenmarktplatzes zu Creative Common Lizenzen

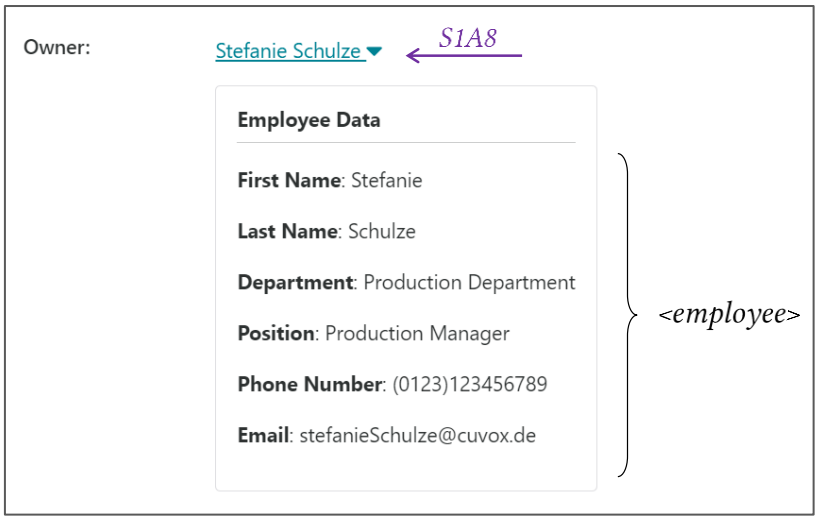


Abbildung 49. Das Attribut *Owner*, dargestellt durch die Metadaten des *<employee>*-Templates

Im Bereich „Delivery Options“ erfährt der Nutzer, wie oft Datenaktualisierungen geplant sind und wie die Daten bereitgestellt werden können. In Szenario 1 beispielsweise ist die wünschenswerte Zugangsoption für den Datenanalysten der Zugriff auf die Datenbank (Anforderung *S1A10*), während für den Business-Analysten in Szenario 2 ein Download bevorzugt wird (Anforderung *S2A1*). In dem in Abbildung 50 dargestellten Screenshot ist zu sehen, dass zusätzliche Informationen über den Datenzugriff angezeigt werden, wenn das Element *Access method* aufgeklappt wird. Im Falle von Szenario 1 sind dies die Metadaten des *<access>*-Templates, im Falle von Szenario 2 werden in der Aufklappliste die im *<download>*-Template erfassten Metadaten angezeigt. Somit können die Benutzer dank der „Delivery Options“-Registerkarte und den für mehrere Provisionierungsoptionen erstellten Templates die Daten auswählen, die ihren Wünschen in Bezug auf den Datenzugriff entsprechen. Dies erfüllt die Anforderungen *S1A10* und *S2A1*.

Abbildung 50. „Delivery Options“-Bereich der Detailseite, der das Template *<delivery_options>* abbildet

Neben Metadaten, die für alle Datensätze relevant sind (z.B. Format, Größe, Typ oder Beschreibung), kann der Bereich „General Information“ auch zusätzliche Attribute für verschiedene Datentypen (strukturiert, semi-strukturiert und unstrukturiert) enthalten. So möchte der Datenanalyst in Szenario 1 neben den Sensordatentabellen auch Bilder von fehlerhaften Produkten finden. Anforderung *S1A11* bezieht sich auf Informationen über die Anzahl der Pixel und Farbkomponenten im Bild sowie die Bildauflösung und die Verwendung von Blitzlicht. Der Bereich der Detailseite, der eine detaillierte Beschreibung des Bildes enthält, ist in Abbildung 51 dargestellt. Hier werden die grundlegenden Attribute eines Bildes abgebildet, die im Template *<image>* gesammelt werden. Bei den *Camera-* und *Color-*Attributen handelt es sich um Aufklappelemente, die die weiteren Metadaten zur Beschreibung des Bildes enthalten (Abbildung 52).

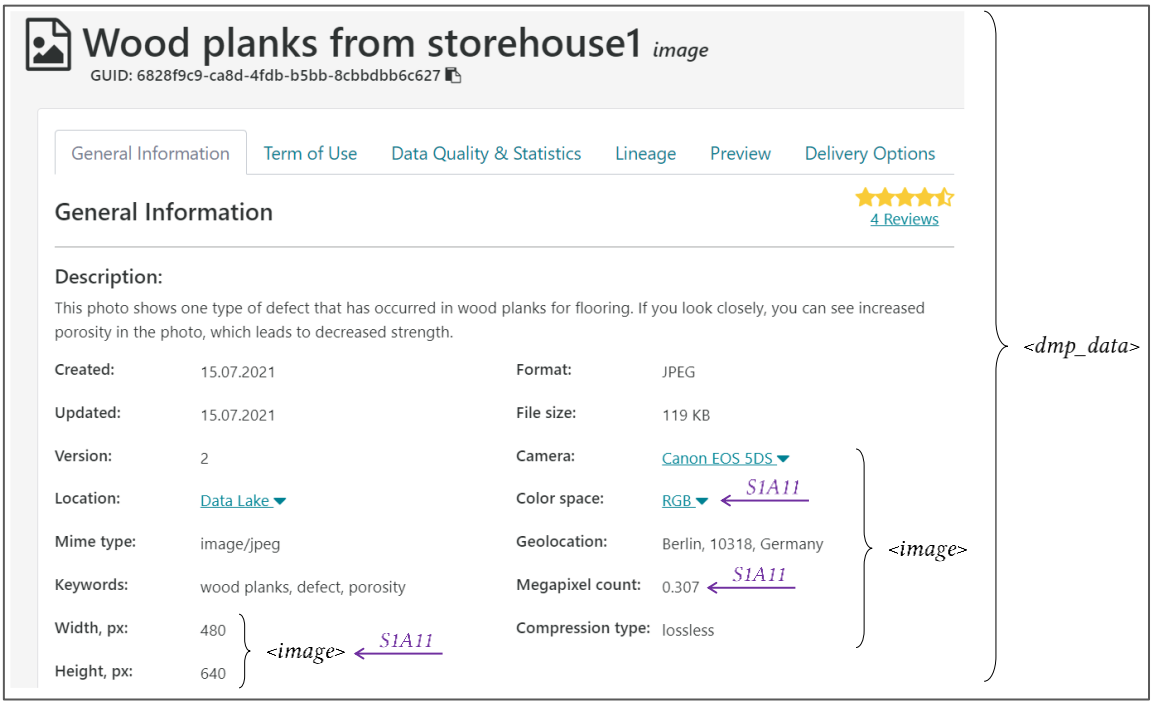


Abbildung 51. Abschnitt der Detailseite des Datensatzes vom Typ *image*, die sowohl die Metadaten des *<dmp_data>*-Templates als auch des Templates *<image>* abbildet

Ein weiteres Beispiel für einen Datensatz, dessen Detailseite Metadaten für einen bestimmten Datentyp enthalten würde, ist ein Textdokument. Dazu gehört z.B. die Detailseite eines Berichts, nach der der Business-Analyst in Szenario 2 sucht. Genau wie bei einem Bild wären in diesem Fall alle Informationen aus dem Template *<dmp_data>* verfügbar, ebenso wie Angaben über die Anzahl der Seiten, Wörter und Zeichen in der Textdatei sowie andere, d.h. Metadaten aus dem Template *<text>*. Die Angabe dieser Metadaten erfüllt die Anforderung *S2A3*, wonach der der Business-Analyst die Jahres- und Quartalsberichte in Textform finden können möchte.

Das letzte Beispiel für eine Detailseite für unstrukturierte Daten ist eine Detailseite für einen Datensatz, der Audioaufnahmen enthält. In Szenario 2 möchte der Business-Analyst Tonaufnahmen finden und hat Wünsche bezüglich ihrer Dauer (Anforderung *S2A4*). Ebenso wie bei Bildern und Texten wird die Detailseite verschiedene technische Metadaten inkl. dem Attribut *Duration* enthalten, dieses Mal aus dem *<audio>*-Template. Auch hier werden für

einige Attribute, die detailliertere Metadaten enthalten, Aufklapplisten angezeigt. Dies werden die Attribute der Templates *<channel>*, *<stereo>* und *<codec>* sein.

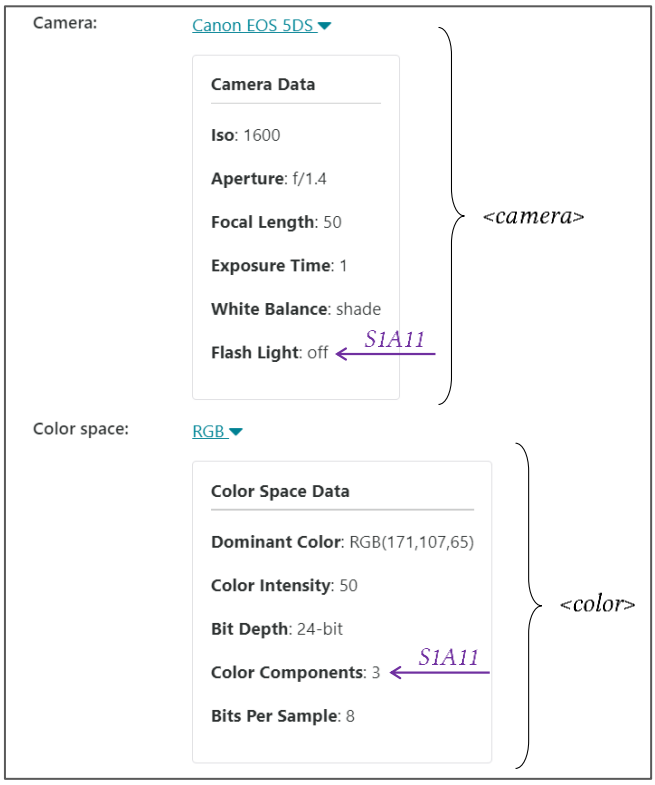


Abbildung 52. *Camera*- und *Color Space*-Attribute, dargestellt durch die Metadaten des *<camera>*- und *<color>*-Templates

Anhand der beschriebenen Beispiele für Bilder, Texte und Audiodateien lässt sich feststellen, dass die Templates für die verschiedenen Datentypen die Anforderung *S1A11* des Szenarios 1 sowie die Anforderungen *S2A3* und *S2A4* des Szenarios 2 abdecken.

Somit wurde in diesem Abschnitt gezeigt, dass die Konzepte der Metadatentemplates und ihrer Darstellung im Datenmarktplatz-Frontend sowohl die Anforderungen *S1A1* – *S1A11* als auch *S2A1* – *S2A5* der beiden Anwendungsszenarien aus Kapitel 3 erfüllen.

11.2 Überprüfung der Erfüllung der Anforderungen an den Datenmarktplatz

Da alle Anforderungen aus Szenario 1 und Szenario 2 erfüllt sind, lässt sich schlussfolgern, dass die Metadaten aus den Templates sowohl technischen (dem Datenanalysten aus Szenario 1) als auch nicht-technischen Nutzern (dem Business-Analysten aus Szenario 2) des Datenmarktplatzes helfen, Daten zu finden und ihre Eigenschaften zu verstehen. Jetzt ist auch zu klären, ob die in Kapitel 4 aufgeführten allgemeinen Anforderungen an den Datenmarktplatz erfüllt sind. Dazu ist vorgesehen, den Weg der Datenmarktplatznutzer noch einmal sequenziell zu durchlaufen und dabei zu überprüfen, ob die Anforderungen eingehalten werden. Die funktionalen Anforderungen sind ein guter Ausgangspunkt.

Alles beginnt damit, dass der Nutzer die Detailseite öffnet und auf der Registerkarte mit den grundlegenden Informationen („General Information“) landet. Zur Verdeutlichung zeigt Abbildung 53 einen Screenshot der Detailseite des Datensatzes von Typ *collection*. Wie bereits

im vorherigen Abschnitt erwähnt, enthält die Detailseite den Namen des Datensatzes, seinen Typ und seine Beschreibung. Unter dem Namen befindet sich auch eine GUID-Nummer, mit der der Datensatz eindeutig identifiziert werden kann. Dies alles sind *beschreibende Metadaten*. Die Anforderung *A1* lautet, dass Datensätze mit beschreibenden Metadaten versehen werden sollten, die zur eindeutigen Identifizierung und effizienten Suche nach den Datensätzen beitragen. Damit ist die Anforderung *A1* erfüllt.

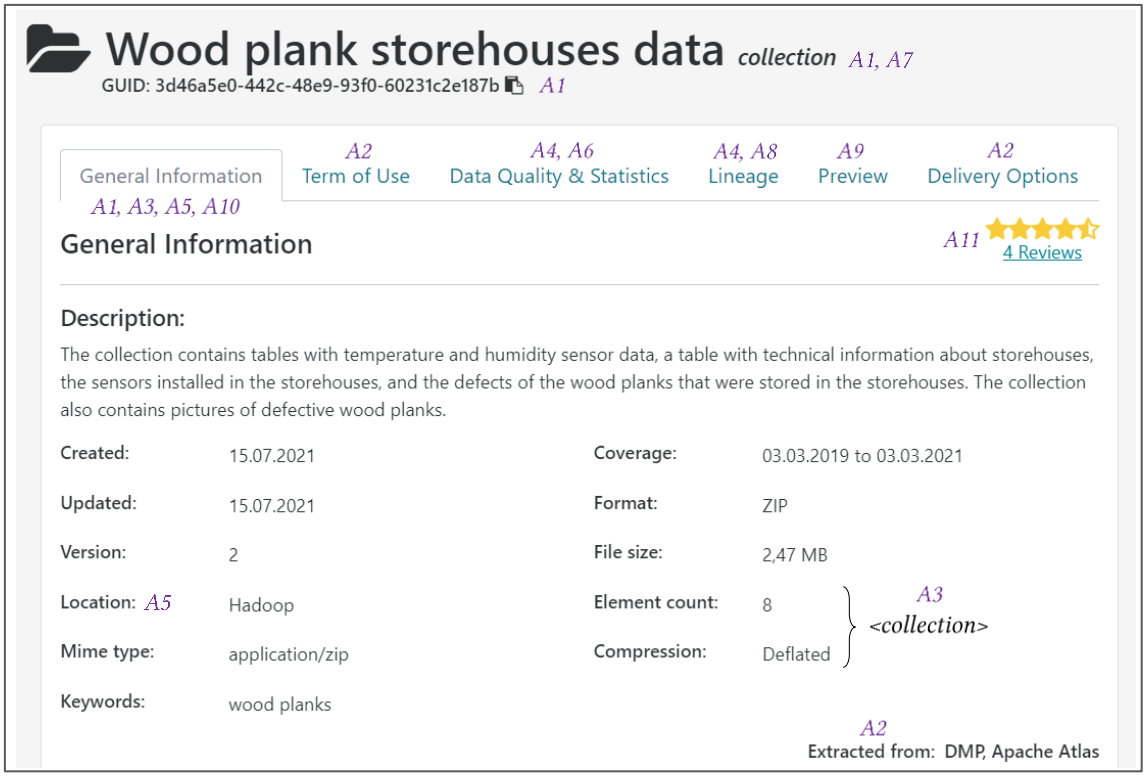


Abbildung 53. Abschnitt der Detailseite des Datensatzes vom Typ *collection*

Die zweite Registerkarte umfasst die Nutzungsbedingungen, aus denen hervorgeht, wer der Data Owner ist, welche Nutzungsbedingungen für die Daten gelten und welche Verwendung für die Daten vorgesehen ist. Hierbei handelt es sich um *administrative Metadaten*, deren Vorkommen in der Registerkarte „Term of Use" einen Teil der Anforderung *A2* abdeckt. In diesem Teil geht es darum, dass Datensätze auch mit administrativen Metadaten beschrieben werden sollten. Weitere administrative Metadaten befinden sich auch auf der Registerkarte „Delivery options", wo Informationen darüber verfügbar sind, wie und wie oft die Daten geliefert werden können. Damit ist ein weiterer Teil der Anforderung *A2* erfüllt, in dem es um den Bedarf an Informationen über die Provisionierungsoptionen für Datensätze geht. Der letzte Teil dieser Anforderung besteht darin, dem Benutzer Informationen darüber zur Verfügung zu stellen, wie die Metadaten gewonnen wurden, indem in der unteren rechten Ecke jedes Detailseite-Bereichs die Tools aufgeführt werden, die zum Extrahieren der Metadaten verwendet wurden. Damit ist die Anforderung *A2* bezüglich der Verfügbarkeit von administrativen Metadaten auf dem Datenmarktplatz vollständig erfüllt.

Auf der Registerkarte „General Information", sind je nach Datentyp auch *technische Metadaten* wie z.B. *Bitrate*, *Channel*, *Codec* für *audio* oder *Camera*, *Compression Type* und *Color space* für *image* (Abbildung 51) enthalten. Die Bereitstellung von technischen Metadaten für

Datensätze ist in Anforderung *A3* festgelegt, die somit durch Templates für verschiedene Datentypen umgesetzt wird, deren technische Metadaten in der Registerkarte „General Information“ dargestellt werden.

Anforderung *A4* verdeutlicht den Bedarf an *Business-Metadaten* zur besseren Erklärung der Daten für nicht-technische Business-Nutzer, z.B. die Bedeutungen der Spalten, wenn es sich um eine Tabelle handelt. Abschnitt 2.6.2 weist darauf hin, dass zu den Business-Metadaten auch die Datenqualität und die Daten Lineage gehören. Die Bereiche der Detailseite, in denen sich die Business-Metadaten befinden, sind der Bereich „Data Quality & Statistics“ (Abbildung 44) und der Bereich „Lineage“ (Abbildung 43). Da in den Anforderungen an den Datenmarktplatz Anforderungen an die Verfügbarkeit der Lineage (Anforderung *A6*) und die Datenqualität (Anforderung *A8*) extra festgelegt sind, erfüllen die beschriebenen Detailseiten-Bereiche insgesamt drei Anforderungen, nämlich *A4*, *A6* und *A8*.

Um die Erfüllung der Anforderung *A5* zu kontrollieren, ist es notwendig, zur Registerkarte „General Information“ zurückzukehren. Diese Anforderung besagt, dass der Datenspeicherort bekannt sein sollte, einschließlich der Angabe, ob die Daten in einem Unternehmenssystem gespeichert sind. Auf der Registerkarte „General Information“ kann der Benutzer detaillierte Informationen über den Speicherort (Attribut *Location*) der Daten finden. Dies kann entweder ein String sein (z.B. *Hadoop*, wie in Abbildung 53) oder ein Aufklappelement mit zusätzlichen Attributen, wenn es sich um ein Unternehmenssystem handelt (z.B. *Data Lake*, wie in Abbildung 40). Somit ist auch die Anforderung *A5* erfüllt.

Anforderung *A7* bezieht sich auf die Möglichkeit, mehrere Datensätze gleichzeitig auf dem Datenmarktplatz anzubieten. Eine solche Möglichkeit ist, wie in Abschnitt 0 beschrieben, ebenfalls vorgesehen, und alle relevanten Metadaten werden in dem Template *<collection>* erfasst. Zurück zu Szenario 1: der Datenanalyst könnte auch alle Tabellen und Bilder der Produkte in derselben Datensammlung finden. Abbildung 53 zeigt einen Screenshot mit den allgemeinen Informationen einer solchen Sammlung. Hier werden der Typ *collection* sowie zusätzliche Attribute wie *Compression* und *Element count* angezeigt, was bestätigt, dass die Anforderung *A7* ebenfalls erfüllt wurde.

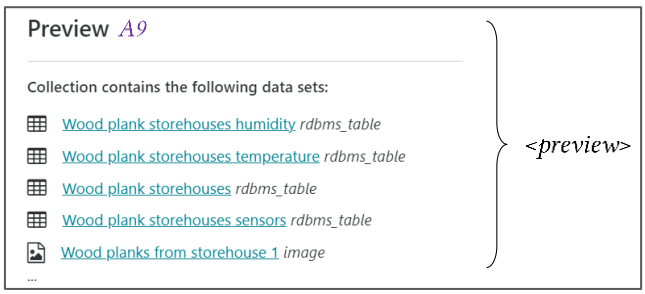


Abbildung 54. „Preview“-Bereich der Detailseite für den Datensatz vom Typ *collection*

Abbildung 54 besteht aus einem Screenshot des „Preview“-Bereichs für die Datenkollektion, der die ersten fünf Elemente der Kollektion zeigt, die als Links zu den entsprechenden Datensätzen dargestellt sind. Diese Datensätze können auch separat angesehen und angefragt werden. Die Previews für die Tabelle (Abbildung 45) und das Bild (Abbildung 46) wurden

bereits im vorherigen Abschnitt gezeigt. Daraus lässt sich schließen, dass die Anforderung *A9* erfüllt ist, die besagt, dass eine Vorschau der Daten wünschenswert ist.

Aus Abschnitt 6.3.4 ist bekannt, dass IoT-Daten häufig in Unternehmen gespeichert werden. Daher legt Anforderung *A10* nahe, dass geeignete Metadaten auch auf dem Datenmarktplatz wünschenswert sind. Solche Metadaten befinden sich, wie im vorigen Abschnitt erörtert, in dem Aufklappelement für das *Categorie*-Attribut (Abbildung 42), wodurch die Anforderung *A10* ebenfalls vollständig abgedeckt wird.

Die letzte funktionale Anforderung *A11* betrifft eine der Funktionen des Datenmarktplatzes, nämlich das Crowdsourcing. Es wird vorgeschlagen, diese Funktion durch die Bereitstellung der Möglichkeit zur Bewertung von Datensätzen zu implementieren, bei der die Nutzer ihre Meinungen austauschen können. Das bedeutet, dass der „Reviews"-Bereich, der auch in Abbildung 55 zu finden ist und die Metadaten des Templates *<reviews>* darstellt, die Anforderung *A11* erfüllt.

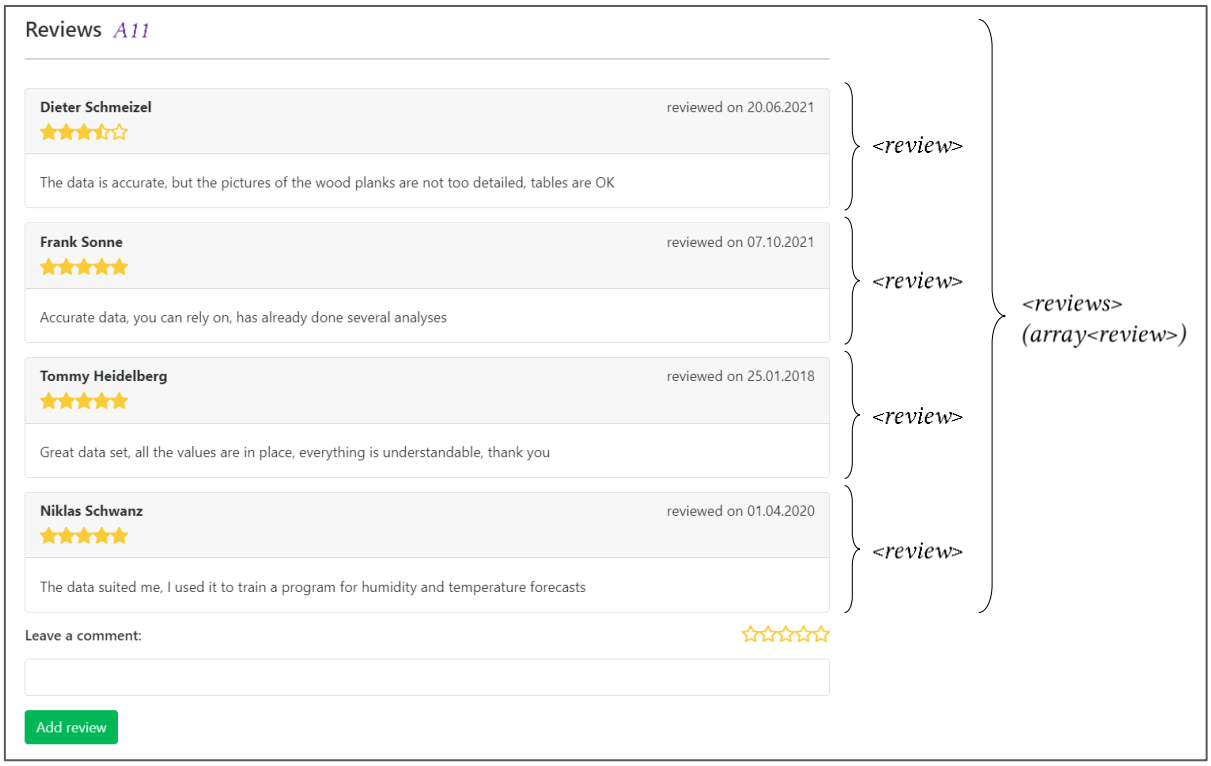


Abbildung 55. „Reviews"-Bereich der Detailseite, der das Template *<reviews>* abbildet

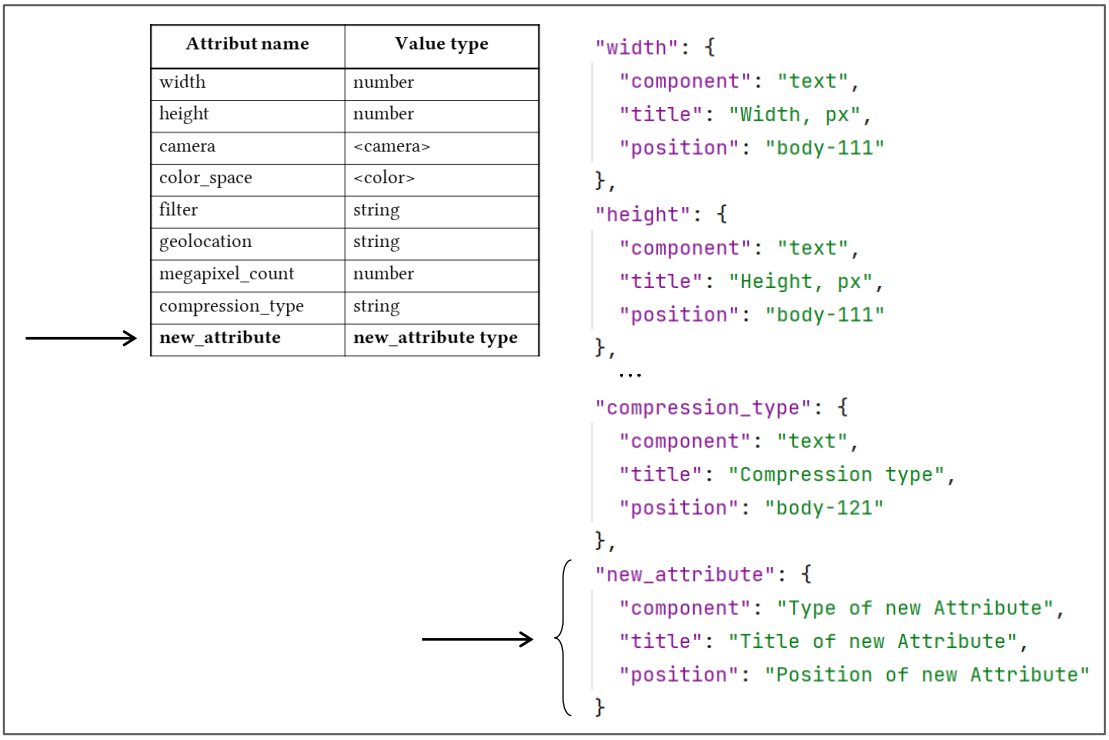
Anschließend lassen sich die nicht-funktionalen Anforderungen auf ihre Einhaltung überprüfen.

Es wurde eine einheitliche Darstellung implementiert, bei der die Registerkarten zur Strukturierung der Detailseite beitragen, sodass die Navigation durch alle Metadaten erleichtert wird. Alle Registerkarten befinden sich immer an der gleichen Stelle und haben die gleiche Reihenfolge. Wenn für einen der Abschnitte keine Metadaten verfügbar sind, bleibt die Registerkarte bestehen, so dass der Benutzer nicht vergeblich danach suchen muss. In dem Bereich „Lineage", wie in Abbildung 56 dargestellt, wird z.B. angezeigt, dass für diesen

Datensatz keine Informationen vorliegen. Außerdem verschwinden die Registerkarten beim Scrollen nicht, wie in Abbildung 56 ebenfalls zu sehen ist. Dies bedeutet, dass der Benutzer mit einem einzigen Klick aus dem Bereich „Lineage“ z.B. zum Bereich „General Information“ zurückkehren kann. All dies erfüllt die Anforderung *A12*, bei der es um die erwähnte einheitliche Darstellung und Benutzerfreundlichkeit handelt.

Abbildung 56. Ansicht des „Lineage“-Bereichs im Falle, dass die Metadaten zur Lineage für den Datensatz nicht verfügbar sind

Wenn die Menge der in den Templates gesammelten Metadaten in Zukunft nicht ausreicht oder wenn ein größerer Bedarf an spezifischen Metadaten entsteht, kann jedes der Templates mit Metadaten ergänzt werden. Dies ist sowohl auf schematischer Ebene möglich, z.B. in der Planungsphase (Abbildung 57, links), als auch auf der Umsetzungsebene, wo Metadaten zur json-Datei hinzugefügt werden sollen (Abbildung 57, rechts). Der Bedarf an einer Erweiterbarkeit von Templates wird in Anforderung *A13* erwähnt, die somit ebenfalls abgedeckt ist.



Attribut name	Value type
width	number
height	number
camera	<camera>
color_space	<color>
filter	string
geolocation	string
megapixel_count	number
compression_type	string
new_attribute	**new_attribute type**

```
"width": {
  "component": "text",
  "title": "Width, px",
  "position": "body-111"
},
"height": {
  "component": "text",
  "title": "Height, px",
  "position": "body-111"
},
  ...
"compression_type": {
  "component": "text",
  "title": "Compression type",
  "position": "body-121"
},
"new_attribute": {
  "component": "Type of new Attribute",
  "title": "Title of new Attribute",
  "position": "Position of new Attribute"
}
```

Abbildung 57. Der Prozess des Hinzufügens neuer Metadaten in Templates auf schematischer (links) und Umsetzungsebene (rechts)

So wurden auch die allgemeinen funktionalen und nicht-funktionalen Anforderungen an den Datenmarktplatz hinsichtlich der Verwendung von Metadaten überprüft und gezeigt, dass die Anforderungen *A1* – *A12* vollständig erfüllt wurden.

12 Zusammenfassung und Ausblick

Nachdem gezeigt wurde, dass alle funktionalen und nicht-funktionalen Anforderungen an den Datenmarktplatz hinsichtlich der Verwendung von Metadaten im Rahmen dieser Arbeit erfüllt wurden, ist es auch möglich, die wichtigsten Schlussfolgerungen zu formulieren und darüber zu diskutieren, was in Zukunft weiter verbessert werden könnte.

12.1 Zusammenfassung

Es ist festzustellen, dass die Hauptaufgabe dieser Arbeit darin bestand, Metadaten in Templates zu sammeln und zu strukturieren, um sie auf dem internen Datenmarktplatz zu präsentieren. Es wurde erklärt, dass der interne Datenmarktplatz eine Plattform darstellt, die es den Mitarbeitern des Unternehmens ermöglicht, Daten auszutauschen. Die wichtigsten Bestandteile des Datenmarktplatzes bestehen aus dem Informationsprodukt, dem Datenkatalog sowie dem Metadaten-Repository. Die Hauptakteure stellen die Datenanbieter und die Datenkonsumenten dar. Nach der Betrachtung der Funktionen des Datenmarktplatzes lässt sich sagen, dass seine Umsetzung im Unternehmen zur Datendemokratisierung beiträgt, d.h. einer besseren Zugänglichkeit der Daten dient. Somit kann das Problem der heterogenen Daten gelöst werden, da es der Datenmarktplatz ermöglicht, Informationen (Metadaten) über die Daten an einem Ort zu sammeln, unabhängig davon, in welcher Datenbank oder welchem Unternehmenssystem sich die Daten befinden.

Die Heterogenität der Daten in Unternehmen ergibt sich aus der Tatsache, dass viele Datenquellen existieren, sowohl externe als auch interne, sowie mehrere Datenbanken und Unternehmenssysteme, die wiederum weitere Datenbanken und Filesysteme enthalten. Darüber hinaus können die in diesen Datenspeichern abgelegten Daten entweder strukturiert, semi-strukturiert oder unstrukturiert sein. Für jeden dieser Datentypen eignen sich unterschiedliche Datenbanken und Verarbeitungsmethoden. Zudem durchlaufen die Daten einen komplizierten Lebenszyklus, der alle Änderungen, einschließlich Bereinigung und Anreicherung, umfassen kann.

Um all diese vielfältigen Daten zu verstehen und alle daran vorgenommenen Änderungen nachzuvollziehen, ist es notwendig, Informationen darüber zu speichern. Diese Informationen, also Daten über die Daten, werden als Metadaten bezeichnet. Die Hauptfunktionen von Metadaten bestehen darin, das Informationsprodukt zu beschreiben und zu identifizieren sowie das Urheberrecht, Änderungen und den Zugriff darauf zu verfolgen. Vier Typen von Metadaten wurden in dieser Arbeit berücksichtigt, nämlich beschreibende, administrative, technische und Business-Metadaten. Jeder dieser Typen trägt zu einem besseren Verständnis der Daten und ihrer Eigenschaften bei. Es können auch allgemeine oder spezifische Metadatenstandards verwendet werden, um den Austausch von Metadaten zwischen verschiedenen Systemen zu gewährleisten. Damit Metadaten ihre Funktionen in vollem Umfang erfüllen können, müssen sie unter anderem aktuell, genau, reichhaltig und zugänglich sein. Auf dem Datenmarktplatz spielen Metadaten eine zentrale Rolle, da durch sie alle auf dem Datenmarktplatz angebotenen Datensätze beschrieben werden, d.h. die Nutzer des Datenmarktplatzes werden sich bei ihrer Auswahl an Metadaten orientieren.

Zwei verschiedene Nutzer des Datenmarktplatzes, nämlich der Datenanalyst, der als technischer Nutzer auftritt, und der Business-Analyst, der ein nicht-technischer Nutzer ist, fungierten in dieser Arbeit als Akteure in den Anwendungsszenarien für den Datenmarktplatz. Es wurden zwei solcher Szenarien ausgearbeitet, und auf deren Grundlage wurden die Anforderungen an den Datenmarktplatz hinsichtlich der Verwendung von Metadaten formuliert.

Um diese Anforderungen zu erfüllen, wurden zunächst verwandte Arbeiten betrachtet, welche die Tendenz aufweisen, die Metadaten nach bestimmten Kriterien zu gruppieren. Diese Idee der Gruppierung wurde auch für die Erstellung von Metadatentemplates verwendet. Es wurden ein allgemeines Template sowie Templates für Nutzungsbedingungen, Datenspeicherung und Unternehmenssysteme, Provisionierungsoptionen, Lineage, Datenqualität sowie Previews und Reviews erstellt. Alle diese Templates können für alle Datensätze auf dem Datenmarktplatz verwendet werden, unabhängig von deren Typ. Das allgemeine Template enthält beschreibende Metadaten, die zur eindeutigen Identifizierung und zum effizienten Auffinden der Daten beitragen. Administrative Metadaten sind in den Templates für Nutzungsbedingungen, Provisionierungsoptionen und Datenspeicherung gesammelt. Darüber hinaus enthalten die Templates für Datenspeicherung auch technische Metadaten, einschließlich Metadaten für Unternehmenssysteme wie Data Warehouse und Data Lake. Templates für Lineage, Datenqualität, Reviews und Preview verfügen sowohl über technische als auch über Business-Metadaten. Weitere technische Metadaten sind in Templates für strukturierte, semi-strukturierte und unstrukturierte Daten enthalten, die ebenfalls im Rahmen dieser Arbeit erstellt wurden. Diese Templates umfassen sowohl Metadaten für verschiedene relationale und nicht-relationale Datenbanken als auch Metadaten für Texte, Bilder, Audio- und Videoaufnahmen. Außerdem wurden für zwei Datenkategorien, nämlich Sensordaten und Daten des maschinellen Lernens, separate Templates entwickelt, die aus den entsprechenden technischen Metadaten bestehen. Zudem wurde die Möglichkeit vorgesehen, Datenkollektionen, die mehrere Datensätze umfassen, auf dem Datenmarktplatz anzubieten, wofür ebenfalls ein separates Template erarbeitet wurde.

So wurden insgesamt 69 verschiedene Templates entwickelt und deren Framework vorgestellt sowie sieben Gruppen von Templates festgelegt. Alle diese Templates bzw. Template-Gruppen wurden auch in Form eines Metamodells dargestellt, das die Zusammenhänge zwischen ihnen verdeutlicht. Alle in Templates gesammelten Metadaten können mit Hilfe von speziellen Metadatenmanagement-Tools erhalten werden, wie Verzeichnissen, Glossaren, Katalogen, Datenqualitäts-Tools und verschiedenen Bibliotheken zur Extraktion von Metadaten. Diese Arbeit gibt Hinweise darauf, welche Typen von Tools für die Extraktion verschiedener Gruppen von Metadaten eingesetzt werden können.

Um die Einsatzmöglichkeiten der entwickelten Templates auf dem Datenmarktplatz zu veranschaulichen, wurde das Konzept der Darstellung von Metadaten aus den Templates auf der Datensatz-Detailseite entwickelt. Dieses Konzept besteht darin, Gruppen von Metadatentemplates mit Hilfe von Registerkarten darzustellen. Es wurde anschließend für den Datenmarktplatz-Prototyp umgesetzt, der zuvor im Rahmen von zwei verschiedenen studentischen Projekten entwickelt und modifiziert wurde. Im Anschluss wurden die Ergebnisse auf der Grundlage der ausgearbeiteten Szenarien evaluiert, und die Erfüllung aller Anforderungen an den Datenmarktplatz wurde sichergestellt.

Somit können mit den in dieser Arbeit entwickelten Metadatentemplates die Daten unterschiedlicher Typen, die im Unternehmen vorkommen, detailliert beschrieben werden, was das Auffinden und Verstehen der Daten erleichtert sowie deren Nutzung damit effizienter macht. Die Mitarbeiter sind in der Lage, Datensätze zu finden, miteinander zu vergleichen sowie für verschiedene Anwendungsszenarien zu gebrauchen, was die Bearbeitung von Aufgaben erleichtert und beschleunigt. Da die Daten detailliert beschrieben sind, reduziert sich die Anzahl der Datenduplikate im Unternehmen, weil die Daten nicht mehrfach von verschiedenen Mitarbeitern gekauft werden, sondern mit Hilfe von Metadaten auf dem Datenmarktplatz gefunden werden können. Infolgedessen werden mehr Unternehmensdaten wiederverwendet, es wird mehr Wert daraus gewonnen und damit neue Erkenntnisse erworben.

12.2 Ausblick

Obwohl die erstellten Metadatentemplates eine Reihe vielfältiger Metadaten enthalten und viele Aspekte bei ihrer Erstellung berücksichtigt wurden, wie z.B. verschiedene Daten- und Datenspeichertypen, Urheberrechte, Zugriffstypen und andere, kann immer ein Bedarf an detaillierteren oder spezifischeren Informationen über den Datensatz aufkommen. Daher könnte der Schwerpunkt künftiger Arbeiten auf der Erweiterung von Metadatentemplates liegen, einschließlich des Hinzufügens neuer Templates-Gruppen. So könnten analog zu den Templates für die Kategorien „IoT“ und „Machine Learning“ weitere Templates für die anderen in Abschnitt 6.3.4 vorgeschlagenen Kategorien für den Datenmarktplatz erstellt werden. Es können auch Templates für andere, in dieser Arbeit nicht behandelte Unternehmenssysteme ausgearbeitet werden. Darüber hinaus können weitere Typen des Datenzugriffs, wie z.B. der „access to sandbox“, hinzugefügt sowie Templates für verschiedene Dateisysteme, wie z.B. Hadoop, entwickelt werden. Als weitere Änderung könnte die Erweiterung der auf dem Datenmarktplatz verwendeten Lizenztypen hinzukommen.

Eine zusätzliche Aufgabe besteht darin, verschiedene Tools für die automatische Extraktion von Metadaten an den Datenmarktplatz anzuschließen und die extrahierten Metadaten in dem Metadaten-Repository zu speichern. Das Konzept einer solchen Anbindung wurde bereits in dem Forschungsprojekt entwickelt, das in [74] vorgestellt wird. Obwohl dieses Konzept nicht direkt mit Metadatentemplates in Zusammenhang steht, können Templates bei seiner Umsetzung helfen, Konflikte zwischen Metadaten zu vermeiden, die von verschiedenen Tools erfasst werden. Bei der Implementierung können auch Metadatenstandards berücksichtigt werden, z.B. der Dublin Core, dessen Kernelemente den Metadaten des allgemeinen <*dmp-data*>-Templates zugeordnet und ebenfalls in dieser Arbeit erläutert wurden.

Es kann auch angebracht sein, Templates für Datendienstleistungen, Anwendungen und Softwarecode zu entwickeln bzw. bereitzustellen, sofern der Datenmarkplatz um diese Produkte erweitert wird.

Literaturverzeichnis

[1] S. Mohanty, M. Jagadeesh und H. Srivatsa, Big Data Imperatives: Enterprise Big Data Warehouse, BI Implementations and Analytics, Apress, 2013.

[2] T. Pellegrini, H. Sack und S. Auer, Linked Enterprise Data - Management und Bewirtschaftung vernetzter Unternehmensdaten mit Semantic Web Technologien, Springer Vieweg, 2015.

[3] L. Meisel und M. Spiekermann, „Datenmarktplätze - Plattformen für Datenaustausch und Datenmonetarisierung in der Data Economy," Dortmund, 2019.

[4] D. Haynes, Metadata for Information Management and Retrieval, London: facet publishing, 2018.

[5] D. Wells, „The Rise of the Data Marketplace: Data as a Service," 2017.

[6] A. Thamm, M. Gramlich und A. Borek, The Ultimate Data and AI Guide: 150 FAQs about Artificial Intelligence, Machine Learning and Data, Data AI Press, 2020.

[7] G. Berezhna, *Datenmarktplätze,* Universität Stuttgart: Seminararbeit, 2021.

[8] C. Gröger, „There Is No AI Without Data. Industry Experiences on the Data Challenges of AI and Call for a Data Ecosystem for Industrial Enterprises," *Communications of the ACM,* 2021.

[9] *Die transformative Kraft des Data Marketplace,* Early Adopter Research.

[10] M. Spiekermann, B. Otto, M. Hofmann, S. Lehmann-Brauns und R. Tontsch, *Datenmarktplätze in Produktionsnetzwerken,* Bundesministerium für Wirtschaft und Energie, 2020.

[11] T. John und P. Misra, Data Lake for Enterprises, Birmingham: Packt Publishing, 2017.

[12] B. C. Witt, Datenschutz Kompakt und Verständlich: Eine Praxisorientierte Einführung, Springer Vieweg, 2019.

[13] S. Gupta und V. Giri, Practical Enterprise Data Lake Insights, Apress, 2018.

[14] M. Pasha, „Data Warehousing and the Unstructured Data," 2016.

[15] J. V. Luisi, Pragmatic Enterprise Architecture: Strategies to Transform Information Systems in the Era of Big Data, Boston: Morgan Kaufmann, 2014.

[16] A. Virtuoso, M. T. Hocanin, A. Wishnick und R. Pathak, Serverless Analytics with Amazon Athena, Packt Publishing, 2021.

[17] C. Giebler, C. Gröger, E. Hoos, R. Eichler, H. Schwarz und B. Mitschang, „Data Lakes auf den Grund gegangen: Herausforderungen und Forschungslücken in der Industriepraxis," *Datenbank-Spektrum,* Bd. 20, 2020.

[18] A. Gorelik, The Enterprise Big Data Lake: Delivering the Promise of Big Data and Data Science, iO'Reilly Media, Incorporated, 2019.

[19] N. Fishman, C. Stryker und G. Booch, Smarter Data Science: Succeeding with Enterprise-Grade Data and AI Projects, Wiley, 2020.

[20] P. Reveliotis und M. Carey, „Your Enterprise on XQuery and XML Schema: XML-based Data and Metadata Integration," *22nd International Conference on Data Engineering Workshops,* 2006.

[21] C. Gröger und E. Hoos, „Ganzheitliches Metadatenmanagement im Data Lake: Anforderungen, IT-Werkzeuge und Herausforderungen in der Praxis," Nr. 3, 2019.

[22] C. Giebler, C. Gröger, E. Hoos, H. Schwarz und B. Mitschang, „A Zone Reference Model for Enterprise-Grade Data Lake Management," *Proceedings of the 24th IEEE Enterprise Computing Conference,* 2020.

[23] J. Wang und B. Liu, „Design of ETL Tool for Structured Data Based on Data Warehouse," *Proceedings of the 4th International Conference on Computer Science and Application Engineering,* 2020.

[24] H. Yu, H. Cai, Z. Liu, B. Xu und L. Jiang, „An Automated Metadata Generation Method for Data Lake of Industrial WoT Applications," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* 2021.

[25] M.-A. Sicilia, Handbook of metadata, semantics and ontologies, Singapore: World Scientific, 2014.

[26] M. Foulonneau und J. Riley, Metadata for Digital Resources. Implementation, Systems Design and Interoperability, Oxford: Chandos Publishing, 2008.

[27] W. Inmon, B. O'Neil und L. Fryman, Business Metadata: Capturing Enterprise Knowledge, Morgan Kaufmann Publishers, 2010.

[28] F. Castanedo und S. Gidley, Understanding Metadata. Create the Foundation for a Scalable Data Architecture, Sebastopol: O'Reilly Media, 2017.

[29] P. Sawadogo und J. Darmont, „On data lake architectures and metadata management," *Journal of Intelligent Information Systems,* 2021.

[30] R. Gartner, Metadata. Shaping Knowledge from Antiquity to the Semantic Web, London: Springer, 2016.

[31] R. Mahanti, Data Governance and Data Management, Strathfield: Springer, 2021.

[32] F. Ravat und Y. Zhao, „Metadata Management for Data Lakes," *23rd East-European Conference on Advances in Databases and Information Systems,* pp. 37-44, 2019.

[33] J. P. Rohweder, G. Kasten, D. Malzahn, A. Piro und J. Schmid, „Informationsqualität - Definitionen, Dimensionen und Begriffe," in *Daten- und Informationsqualität: Auf dem Weg zur Information Excellence*, Wiesbaden, Vieweg+Teubner, 2008, pp. 25-45.

[34] E. Curry, A. Freitas und S. O'Riáin, „The Role of Community-Driven Data Curation for Enterprises," *Linking Enterprise Data,* 2010.

[35] K. Hüner, B. Otto und H. Oesterle, „Collaborative management of business metadata.," *International Journal of Information Management,* pp. 366-373, 2011.

[36] C. A. Mattmann und J. L. Zitting, Tika in Action, Manning Publications, 2012.

[37] *IEEE Standard for Learning Object Metadata,* 2020.

[38] G. Stamou und S. Kollias, Multimedia Content and the Semantic Web: Standards, Methods and Tools, Wiley & Sons, 2005.

[39] Z. O. Gokturk, N. K. Cicekli und I. Cicekli, „Metadata extraction from text in soccer domain," in *23rd International Symposium on Computer and Information Sciences*, 2008, pp. 1-6.

[40] S. Michalczyk, M. Nadj, A. Maedche und C. Gröger, „Demystifying Job Roles in Data Science: A Text Mining Approach," 2021.

[41] B. Underdahl, Data Integration, 2nd Informatica Special Edition Hrsg., 2018.

[42] K. Mišura und M. Žagar, „Data marketplace for Internet of Things," *2016 International Conference on Smart Systems and Technologies,* pp. 255-260, 2016.

[43] S. Mahanthappa und B. Chandavarkar, „Data Formats and Its Research Challenges in IoT: A Survey," in *Evolutionary Computing and Mobile Sustainable Networks*, 2021, pp. 503-515.

[44] M. Milenkovic, Internet of Things: Concepts and System Design, Dublin, CA, USA: Springer, 2020.

[45] T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. Daumé III und K. Crawford, „Datasheets for Datasets," *Communications of the ACM,* pp. 89-92, 2021.

[46] M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji und T. Gebru, „Model Cards for Model Reporting," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*, New York, USA, Association for Computing Machinery, 2019, pp. 220-229.

[47] J. Richards, D. Piorkowski, M. Hind, S. Houde und A. Mojsilovic, „A Methodology for Creating AI FactSheets," 2020.

[48] Y. Ma und H. Du, Enterprise Data at Huawei. Methods and Practices of Enterprise Data Governance, Shenzhen, China: Springer, 2022.

[49] G. Drakopoulos, E. Spyrou, Y. Voutos und P. Mylonas, „A Semantically Annotated JSON Metadata Structure For Open Linked Cultural Data In Neo4j," *Association for Computing Machinery,* 2019.

[50] R. Eichler, C. Giebler, C. Gröger , H. Schwarz und B. Mitschang, „Modeling metadata in data lakes - A generic model," *Data & Knowledge Engineering,* 2021.

[51] *Richtlinie (EU) 2019/790 des Europäischen Parlaments Und des Rates über das Urheberrecht und die verwandten Schutzrechte im digitalen Binnenmarkt.*

[52] D. Sonntag, Assessing the Quality of Natural Language Text Data, 2004, pp. 259-263.

[53] D. Larry, Digital Forensics for Legal Professionals: Understanding Digital Evidence from the Warrant to the Courtroom, Elsevier, 2011.

[54] M. Biehl, API Architecture: The Big Picture for Building APIs, 2015.

[55] S. Fischer, K. Neubauer, R. Hackenberg, L. Hinterberger und B. Weber, „IoTAG: An Open Standard for IoT Device IdentificAtion and RecoGnition," in *The Thirteenth International Conference on Emerging Security Information, Systems and Technologies*, 2019.

[56] S. Schelter, J.-H. Böse, J. Kirschnick, T. Klein und S. Seufert, „Automatically Tracking Metadata and Provenance of Machine Learning Experiments," in *Machine Learning Systems Workshop at NIPS*, Long Beach, CA, USA., 2017.

[57] D. Sonntag, „Assessing the quality of natural language text data," 2004.

[58] O. Azeroual, G. Saake, M. Abuosba und J. Schöpfel, „Text data mining and data quality management for research information systems in the context of open data and open science," 2018.

[59] M. Pedersen und J. Y. Hardeberg, „Full-Reference Image Quality Metrics: Classification and Evaluation," *Foundations and Trends in Computer Graphics and Vision,* 2012.

[60] I. Avcıbas, B. Sankur und K. Sayood, „Statistical evaluation of image quality measures," *Journal of Electronic Imaging,* p. 206–223, 2002.

[61] H. Alsleem und R. Davidson, „Quality parameters and assessment methods of digital radiography images,“ *The Radiographer*, 2012.

[62] A. Gnanasambandam und S. H. Chan, „Exposure-Referred Signal-to-Noise Ratio for Digital Image Sensors,“ 2021.

[63] *ISO 15739: Photography - Electronic still-picture imaging - Noise measurements,* Geneva, Switzerland, 2017.

[64] K. Thung und P. Raveendran, „A survey of image quality measures,“ 2010.

[65] A. J. Aude, „AN9789: Audio Quality Measurement Primer,“ 2002.

[66] J. L. Harrington, „Chapter 5 - The Relational Data Model,“ in *Relational Database Design and Implementation*, Morgan Kaufmann, 2016, pp. 89-105.

[67] K. Brunnström, A. Djupsjöbacka und B. Andrén, „Objective video quality assessment methods for Video assistant refereeing (VAR) System,“ RISE Research Institute of Sweden, 2021.

[68] A. Vaduva und K. R. Dittrich, „Metadata management for data warehousing: between vision and reality,“ in *Proceedings 2001 International Database Engineering and Applications Symposium*, 2001, pp. 129-135.

[69] R. Eichler, C. Giebler, C. Gröger, Hoos, H. Schwarz und B. Mitschang, „Enterprise-Wide Metadata Management: An Industry Case on the Current State and Challenges,“ *Business Information Systems,* p. 269–279, 2021.

[70] M. Bialke, P. Penndorf, D. Fredrich und K. Weitmann , *Leitfaden zur Beschreibung eines Data Dictionary,* 2017.

[71] Z. Zhao und M. Hellström, Towards Interoperable Research Infrastructures for Environmental and Earth Sciences. A Reference Model Guided Approach for Common Challenges, Cham, Switzerland: Springer, 2020.

[72] J. Barateiro und H. Galhardas, „A survey of data quality tools,“ *Datenbank-Spektrum,* pp. 15-21, 2005.

[73] M. Macura, „Integration Of Data From Heterogeneous Sources Using ETL Technology,“ *Computer Science,* 2014.

[74] M. Fischer, S. Müller und G. Berezhna, *Modular Integration of Metadata Management Tools to the Data Marketplace,* Universität Stuttgart: Forschungsprojekt, 2022.

Anhangssverzeichnis

Anhang 1. Templates

1. Allgemeines Template

Template 1. <dmp_data>

Attribut Name	Dublin Core Attribut	Value Type
name*	dc: title	string
guid*	dc: identifier	string
description*	dc: description	string
type*	dc: type	string
dmp_type*		string (asset, product)
location*		<rdbms_db>, <nosql_dbms_db>, <filesystem>, <enterprise_system>, string
term_of_use* (for product)	dc: rights	<term_of_use>
delivery_options* (for product)		<delivery_options>
language	dc: language	string
create_date	dc: date.created	date, string
modify_date	dc: date.modified	date, string
mime_type		string
format	dc: format	string
size	dc: format.extent	number
coverage	dc: coverage	string
category	dc: subject	<ml_data>, <iot_data>, string
keywords		array<string>
version		number, string
lineage		<lineage>
quality		<quality>
preview		<preview>
reviews		array<review>
metadata_tool		string, link

2. Templates für Nutzungsbedingungen

Template 2. <term_of_use>

Attribut Name	Dublin Core Attribut	Value Type
owner*	dc: creator	<employee>, string
security_class*	dc: rights.accessRights	number (0, 1, 2, 3, 4)
contain_personal_data*		boolean
license*	dc: rights.license	<license>, string
permitted_usage		string
conditions_of_use		string

Template 3. <employee>

Attribut name	Value type
first_name*	string
last_name*	string
department	string
position	string
phone_number	number
email*	string

Template 4. <license>

Attribut name	Value type
name*	string
description	string
link	string

3. Templates für Datenspeicherung und Unternehmenssysteme

Template 5. <file_system>

Attribut name	Value type
name	string
file_path	string
timestamp	date, string

Template 6. <enterprise_system>

Attribut name	Value type
system_type	<warehouse>, <data_lake>, string
storage_location	<rdbms_db>, <nosql_dbms_db>, <filesystem>, string
import_from	<rdbms_db>, <nosql_dbms_db>, <filesystem>, string
import_timestamp	date, string

Template 7. <data_lake>

Attribut name	Value type
data_lake_zone	string
is_in_other_zones	string

Template 8. <data_warehouse>

Attribut name	Value type
currency_of_data	string (active, archived, purged)
transformation_logic	string
warehouse_zone	string (staging, normalized, datamart)

4. Templates für Daten-Provisionierungsoptionen

Template 9. <delivery_options>

Attribut name	Value type
update_cycle	string
data_access_method	<download>, <access>, <api>, string

Template 10. <download>

Attribut name	Value type
link (not completely!)	string
comment	string

Template 11. <access>

Attribut name	Value type
hostname	string
port	int
protocol	string
access_description	string

Template 12. <api>

Attribut name	Value type
api_type	string
root_path	string
endpoint	sting
http_method	string
description	string
parameters	array<parameter>
responses	array<response>

Template 13. <parameter>

Attribut name	Value type
name	string
type	string
description	string
is_required	boolean

Template 14. <response>

Attribut name	Value type
code	number
description	string
media_type	string
example_value	string

5. Templates für Kategorien

Template 15. <iot_data>

Attribut name	Value type
sensor_data	<sensor>
measurement_units	string
sampling_frequency	string
measurement_accuracy	number
min_value	number
max_value	number
connection_type	string (Ethernet, Wi-Fi, Bluetooth)
software	<software>

Template 16. <sensor>

Attribut name	Value type
type	string
model	string
serial_number	number, string
manufacturer	string
location	string

Template 17. <software>

Attribut name	Value type
name	string
version	number
configuration	text

Template 18. <ml_data>

Attribut name	Value type
model	<model>
type	string (traning set, test set)
training_run	<training_run>
test_run	<test_run>
prediction_examples	text

Template 19. <model>

Attribut name	Value type
name	string
version	number, string
learning_algorithm	text
hyperparameters	array<hyperparameter>

Template 20. <training_run>, <test_run>

Attribut name	Value type
start_time	date, string
end_time	date, string
environment_config	text, link<dmp_data>
hyperparameters	array<hyperparameter>

Template 21. <hyperparameter>

Attribut name	Value type
name	string
type	string
value	number

6. Templates für Daten Lineage

Template 22. <lineage>

Attribut name	Value type
input	link<dmp_data>, string
output	link<dmp_data>, string
process	<process>, string

Template 23. <process>

Attribut name	Value type
name	string
actor	<employee>, <application>, string
timestamp	date, string
description	string

Template 24. <application>

Attribut name	Value type
name	string
description	string

7. Templates für Datenqualität

Template 25. <quality>

Attribut name	Value type	For data type
classic_metrics	<classic_metrics>	table, column, text
data_profilling	<profiling>	table, column
image_resolution	<resolution>	image, video
image_artefacts	text	image, video
signal_quality	<signal_quality>	image, audio, video
audio_total_harmonic_distortion	number	audio, video
audio_frequency_response	number	audio, video

Template 26. <classic_metrics>

Attribut name	Value type
accuracy	number
completeness	number
consistency	number
timeliness	number
uniqueness	number
validity	number
correctness	number

Template 27. <profiling>

Attribut name	Value type
min_value	number
max_value	number
sum	number
average	number
median	number
null_count	number
distinct_count	number
unique_count	number
duplicate_count	number
inter_quartile_range	number
lower_quartile	number
upper_quartile	number

Template 28. <resolution>

Attribut name	Value type
x_resolution	number
y_resolution	number
units	string

Template 29. <signal_quality>

Attribut name	Value type
mean_squared_error	number
signal_to_noise_ratio	number
peak_signal_to_noise_ratio	number

8. Templates für Data Preview und Reviews

Template 30. <preview>

Dmp Type	Preview Option
<rdbms_instance>, <nosql_dbms_instance>, <db_schema>, all databases	first 1-7 objects
<spreadsheet>, <rdbms_table>, <view>, <nosql_dbms_table>	first 1-10 rows
<text>	first 1-50 words
<image>	image(s) thumbnail
<audio>, <video>	first 15 seconds
<collection>	first 1-7 items

Template 31. <review>

Attribut name	Value type
user	<employee>, string
rating	number
commentar	string
date	date, string

9. Templates für strukturierte Daten

Template 32. <table>

Attribut name	Value type
column_count	number
list_of_columns	array<column>

Template 33. <column>

Attribut name	Value type
name	string
meaning	string
table	<rdbms_table>, <nosql_dbms_table>, <spreadsheet>
data_type	string
units	string
is_nullable	boolean

Template 34. <spreadsheet>

Attribut name	Value type
column_separator	string
thousands_separator	string
decimals_separator	string
row_count	number

Template 35. <rdbms_instance>

Attribut name	Value type
rdbms_type	string
access	<access>
rdbms_db_count	number
list_of_rdbms_dbs	array_link<rdbms_db>, array<string>

Template 36. <rdbms_db>

Attribut name	Value type
type	string
rdbms_instance	link<rdbms_instance>, string
db_schema_count	number
list_of_db_schemas	array_link<db_schema>, array<string>

Template 37. <db_schema>

Attribut name	Value type
rdbms_db	link<rdbms_db>, string
object_count	number
list_of_objects	array_link<rdbms_table>, array_link<view>, array <string>

Template 38. <rdbms_table>

Attribut name	Value type
db_schema	link<db_schema>, string
rdbms_db	link<rdbms_db>, string
primary_key	string
foreign_keys	array<szting>
row_count	number

Template 39. <view>

Attribut name	Value type
source_table	link<rdbms_table>, string
db_schema	link<db_schema>, string
script	text

10. Templates für semi-strukturierte Daten

Template 40. <nosql_dbms_instance>

Attribut name	Value type
no_sql_dbms_type	string
access	<access>
nosql_dbms_db_count	number
list_of_nosql_dbms_dbs	array_link<nosql_dbms_db>, array<string>

Template 41. <nosql_dbms_db>

Attribut name	Value type
type	<column_oriented_db>, <document_store>, <graph_db>, <hierarchical_db>, <object_oriented_store>, <key_value_store>, string
nosql_dbms_instance	link<nosql_dbms_instance>, string

Template 42. <column_oriented_db>

Attribut name	Value type
table_count	number
list_of_tables	array_link<table>, array<string>

Template 43. <nosql_dbms_table>

Attribut name	Value type
nosql_dbms_db	link<nosql_dbms_db>, string

Template 44. <nosql_dbms_column>

Attribut name	Value type
record_count	number

Template 45. <document_store>

Attribut name	Value type
collection_count	number
list_of_collections	array<doc_collection>, array<string>
document_count	number
list_of_documents	array<document>, array<string>

Template 46. <doc_collection>

Attribut name	Value type
document_count	number
list_of_documents	array<document>, array<string>

Template 47. <document>

Attribut name	Value type
id	number, string
field_count	number
list_of_fields	array<string>

Template 48. <hierarchical_db>

Attribut name	Value type
tree_count	number
list_of_trees	array<tree>, array<string>

Template 49. <tree>

Attribut name	Value type
root_record	string
level_count	number
leaf_count	number
record_count	number

Template 50. <object_oriented_db>

Attribut name	Value type
class_count	number
list_of_classes	array<class>, array<string>

Template 51. <class>

Attribut name	Value type
object_count	number
list_of_objects	array<object>, array<string>

Template 52. <object>

Attribut name	Value type
field_count	number
list_of_fields	array<string>

Template 53. <graph_db>

Attribut name	Value type
nodes_count	number
list_of_nodes	array<node>, array<string>
relationships_count	number
list_of_relationships	array<relationship>, array<string>

Template 54. <node>

Attribut name	Value type
id	number
property_count	number
list_of_properties	array<string>

Template 55. <relationship>

Attribut name	Value type
id	number
property_count	number
list_of_properties	array<string>

Template 56. <key_value_store>

Attribut name	Value type
keys_count	number
list_of_keys	array<key>, array<string>
keyspace_count	number
list_of_keyspaces	array<keyspace>, array<string>

Template 57. <keyspace>

Attribut name	Value type
keys_count	number
list_of_keys	array<key>, array<string>

Template 58. <key>

Attribut name	Value type
attribute_count	number
list_of_attributes	array<string>

Template 59. <composite_key>

Attribut name	Value type
partition_key	string
sort_keys_count	number
list_of_sort_keys	array<key>, array<string>

11. Templates für unstrukturierte Daten

Template 60. <text>

Attribut name	Value type
page_count	number
words_count	number
characters_count	number
characters_count_with_spaces	number
lines_count	number
paragraphs_count	number
encoding	string
page_layout	string
fonts	array<string>
notes	string
presentation_format	string (4:3, 16:9)
slides_count	number

Template 61. <image>

Attribut name	Value type
width	number
height	number
camera	<camera>, string
color_space	<color>, string
filter	string
geolocation	string
megapixel_count	number
image_compression	number, string

Template 62. <camera>

Attribut name	Value type
model	string
iso	number
aperture	number, string
focal_length	number
exposure_time	number
white_balance	string
flash_light	boolean, string (on, off)

Template 63. <color>

Attribut name	Value type
color_space_name	string
dominant_color	string
intencity	number
bit_depth	number, string
components	number
bit_per_samples	number

Template 64. <audio>

Attribut name	Value type
layers_count	number
bitrate	number
sample_rate	number
channel	<channel>
stereo	<stereo>
codec	<codec>
genre	string
album	string
artist	string
duration	number, string

Template 65. <channel>

Attribut name	Value type
mode	string
count	number

Template 66. <stereo>

Attribut name	Value type
ms_stereo	string (on, off)
intensity_stereo	string (on, off)

Template 67. <codec>

Attribut name	Value type
name	string
long_name	string
type	string

Template 68. <video>

Attribut name	Value type
time_scale	number
duration	number, string
preview_time	number, string
preview_duration	number, string
track	<audio>
track_volume	number
avg_bitrate	number
image	<image>
frame_rate	number
compressor_name	string
rotation	number

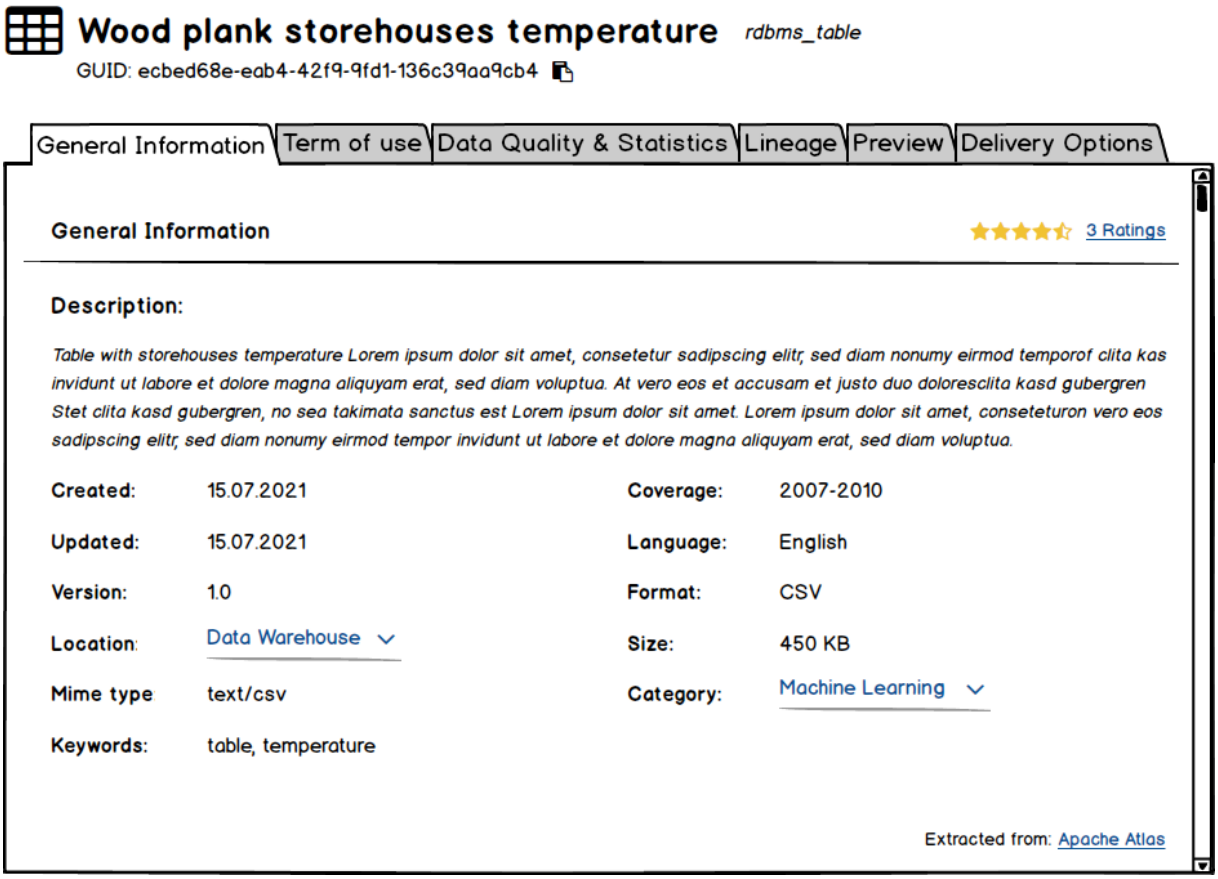
12. Template für Datenkollektionen

Template 69. <data_collection>

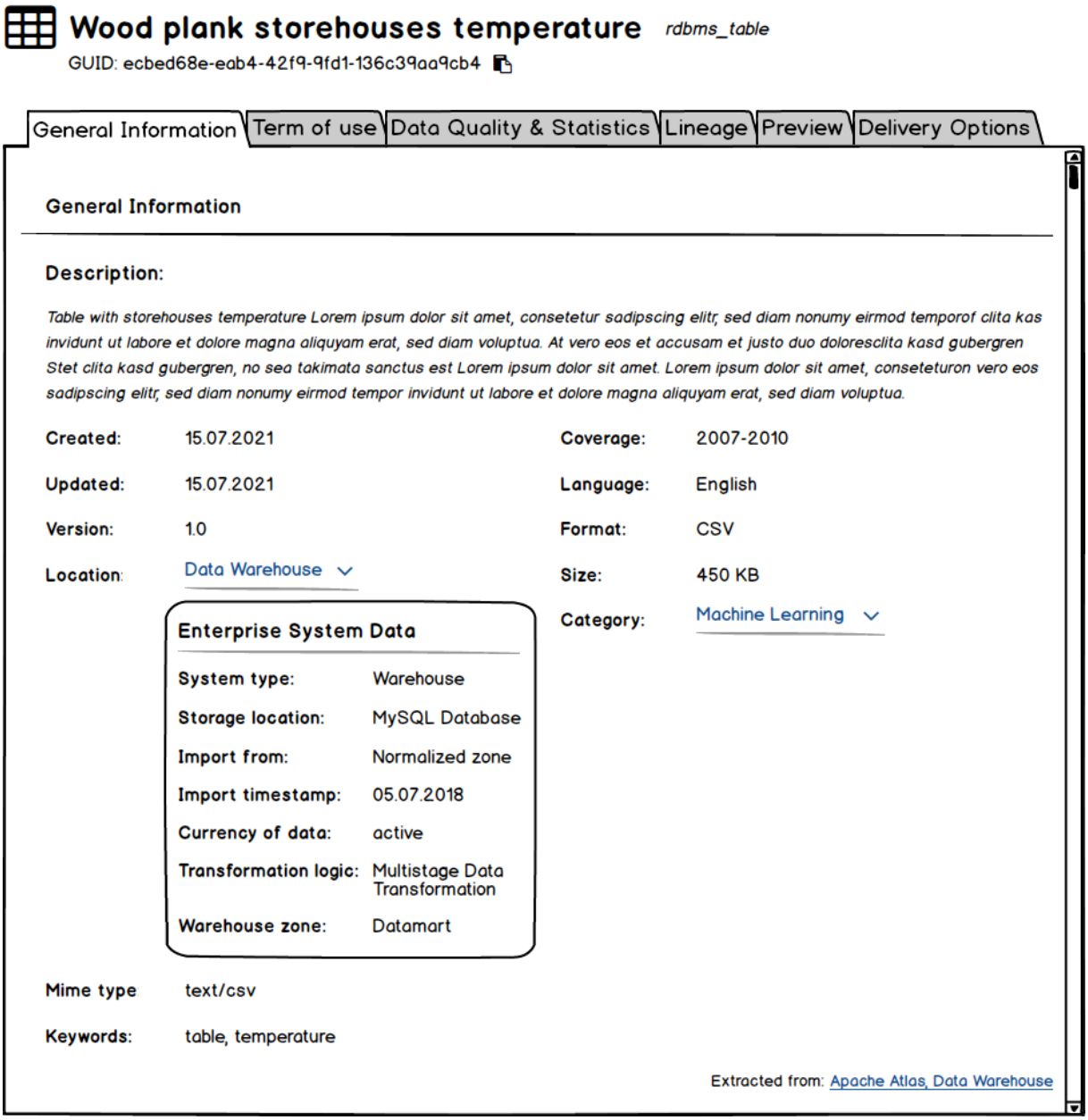
Attribut name	Value type
element_count	number
collection_compression	number, string
list_of_elements	array_link<dmp_data>, array<string>

Anhang 2. Mockups

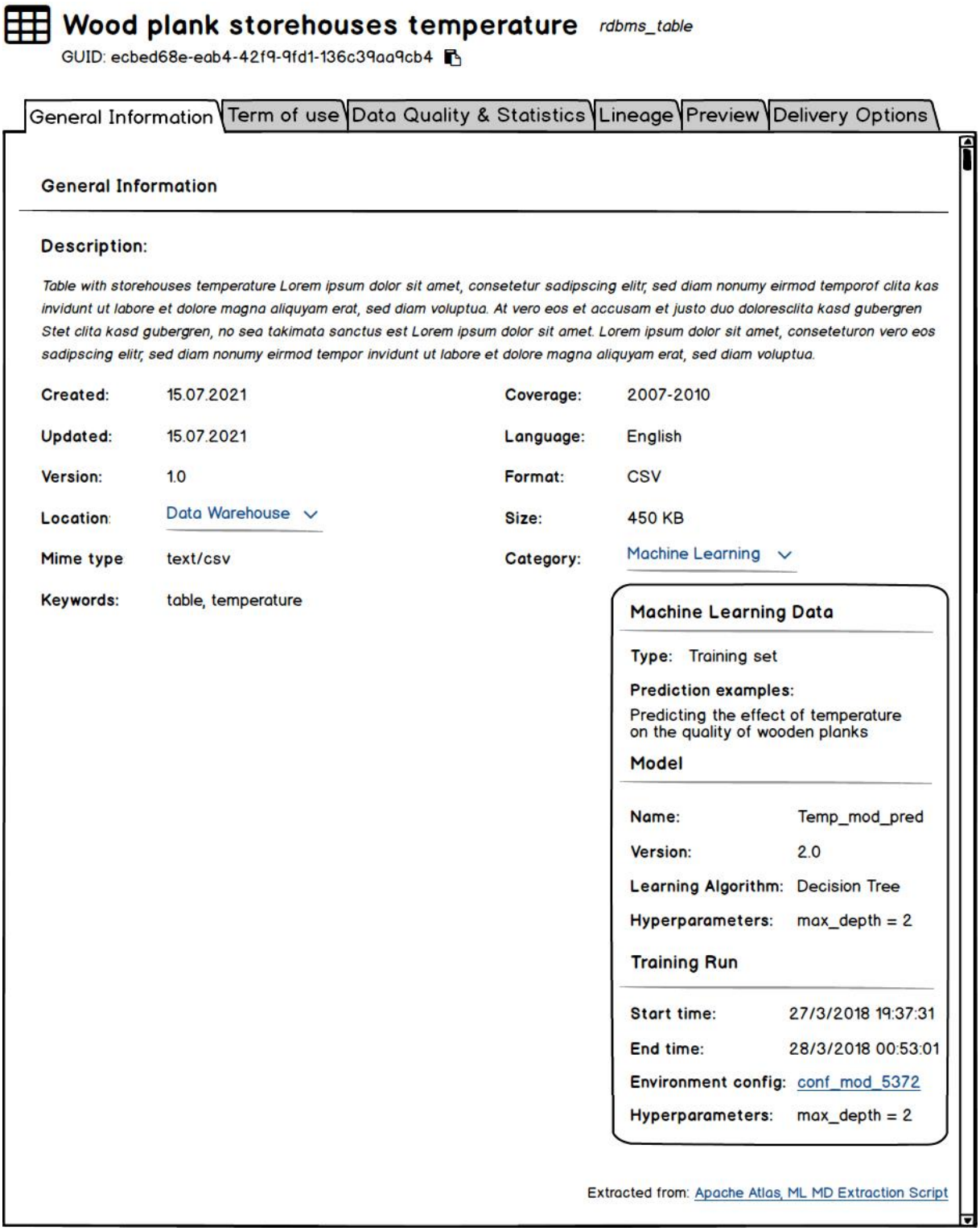
Mockup 1. General Information



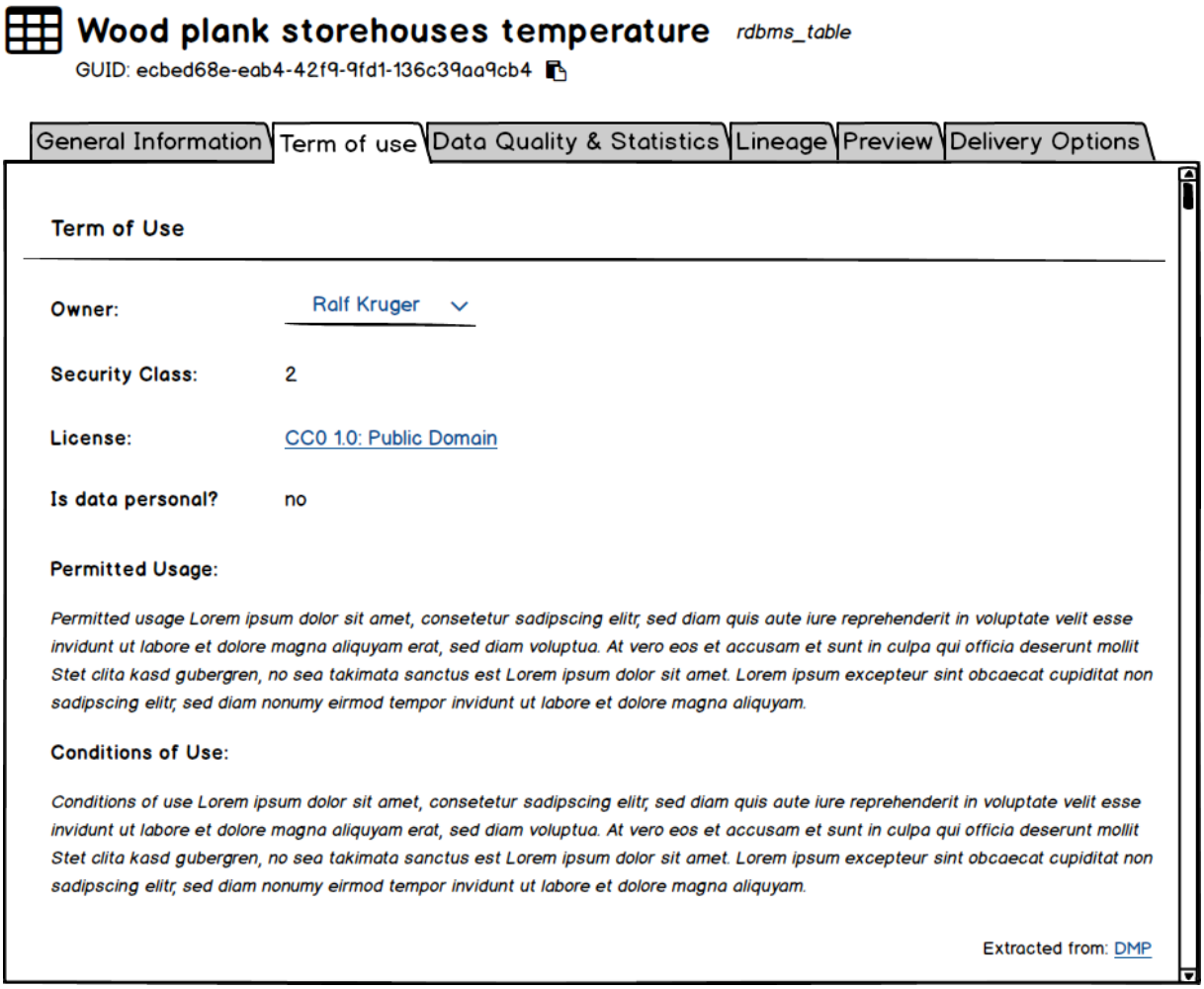
Mockup 2. Data Warehouse Metadaten



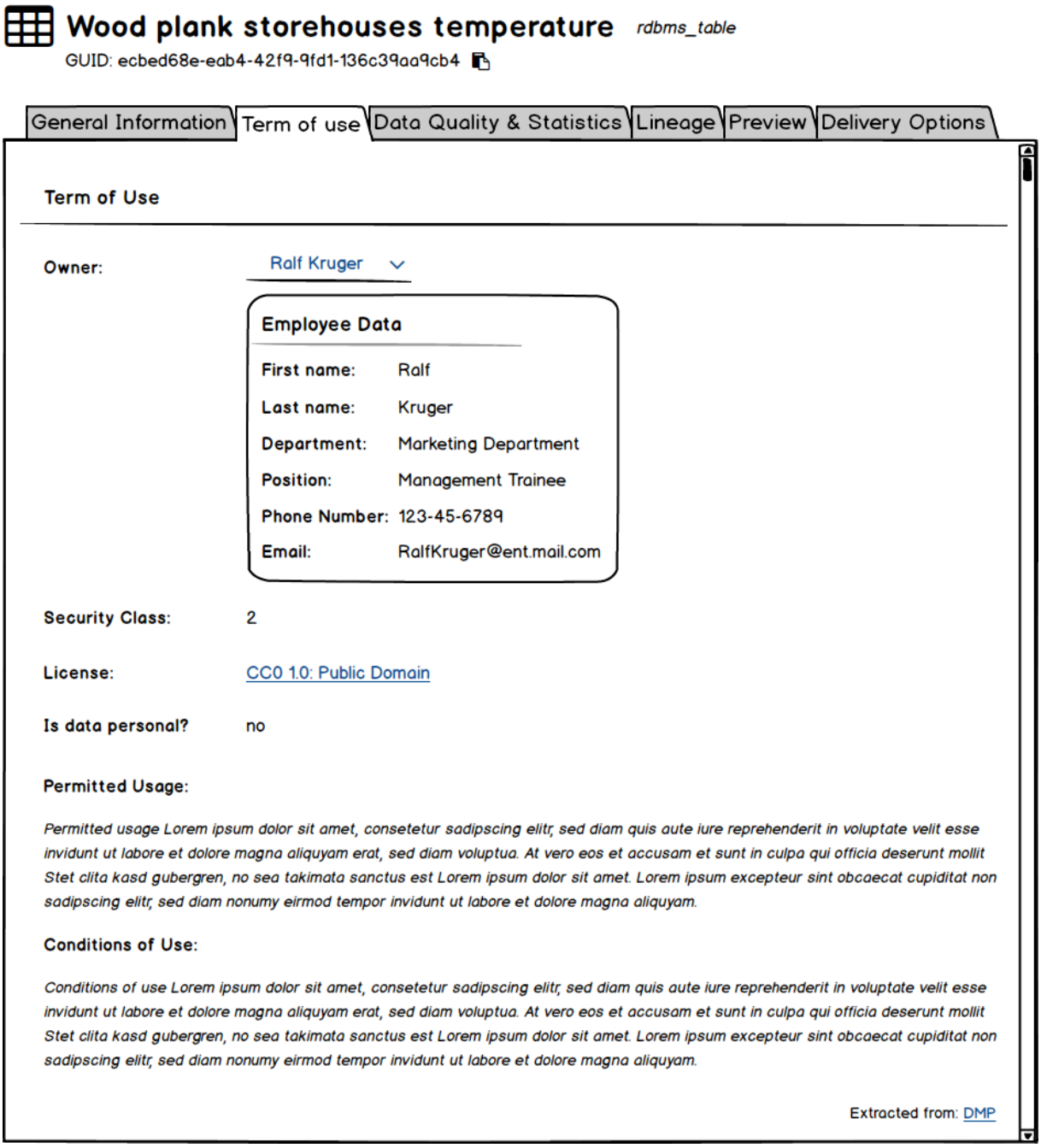
Mockup 3. Machine Learning Data



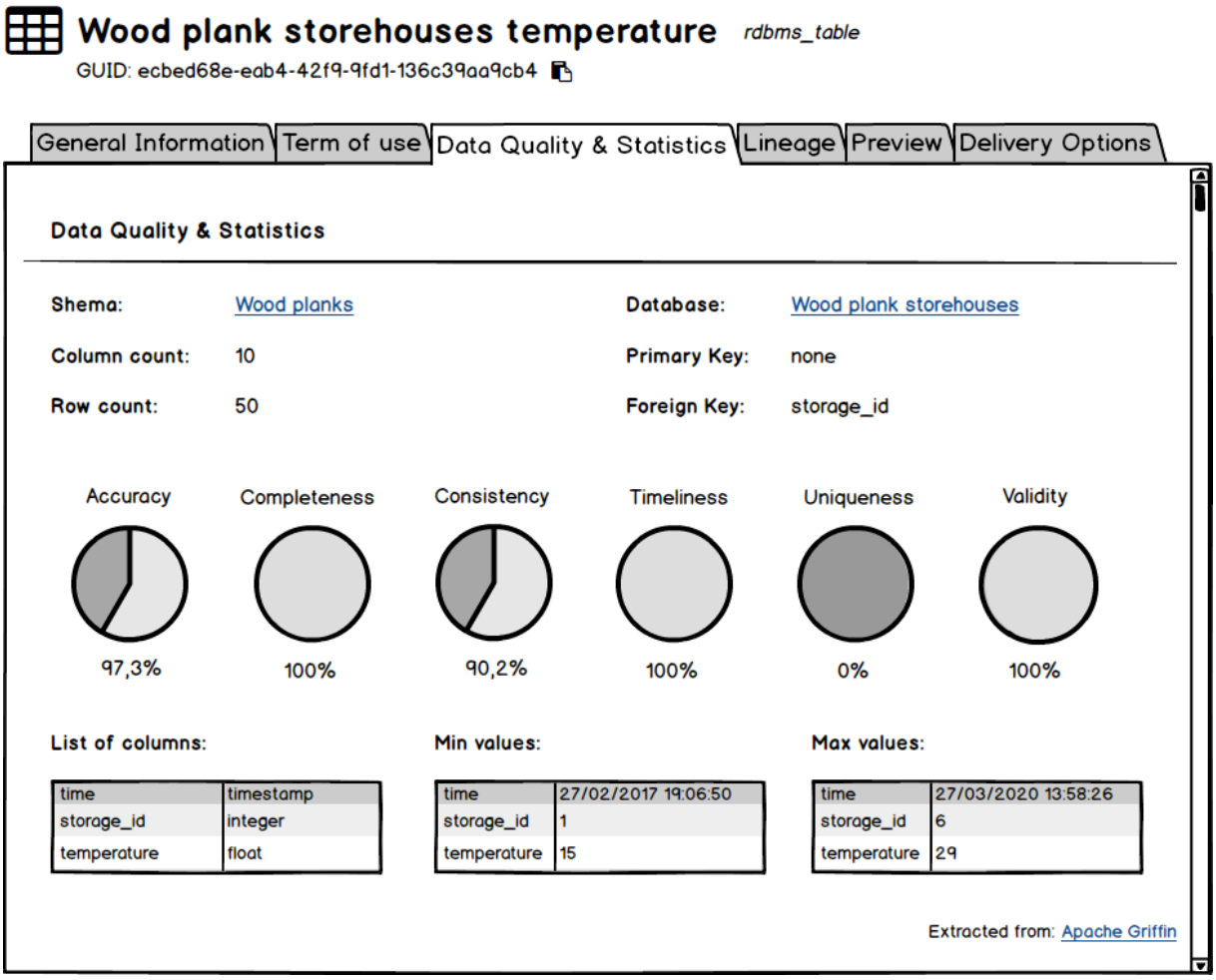
Mockup 4. Term of Use



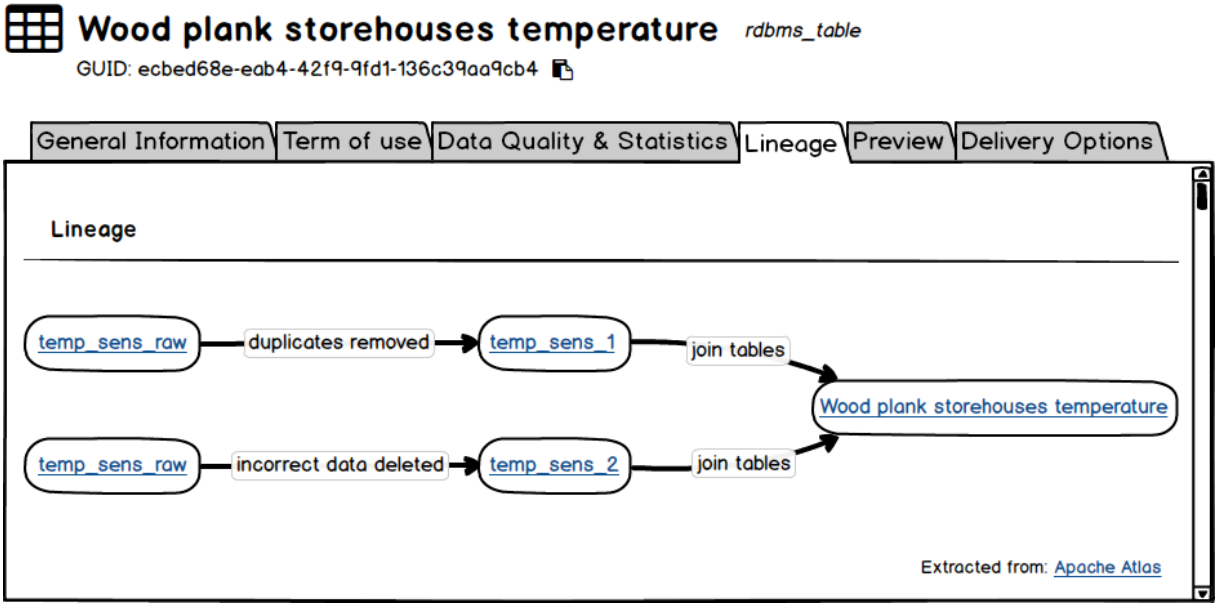
Mockup 5. Employee Data



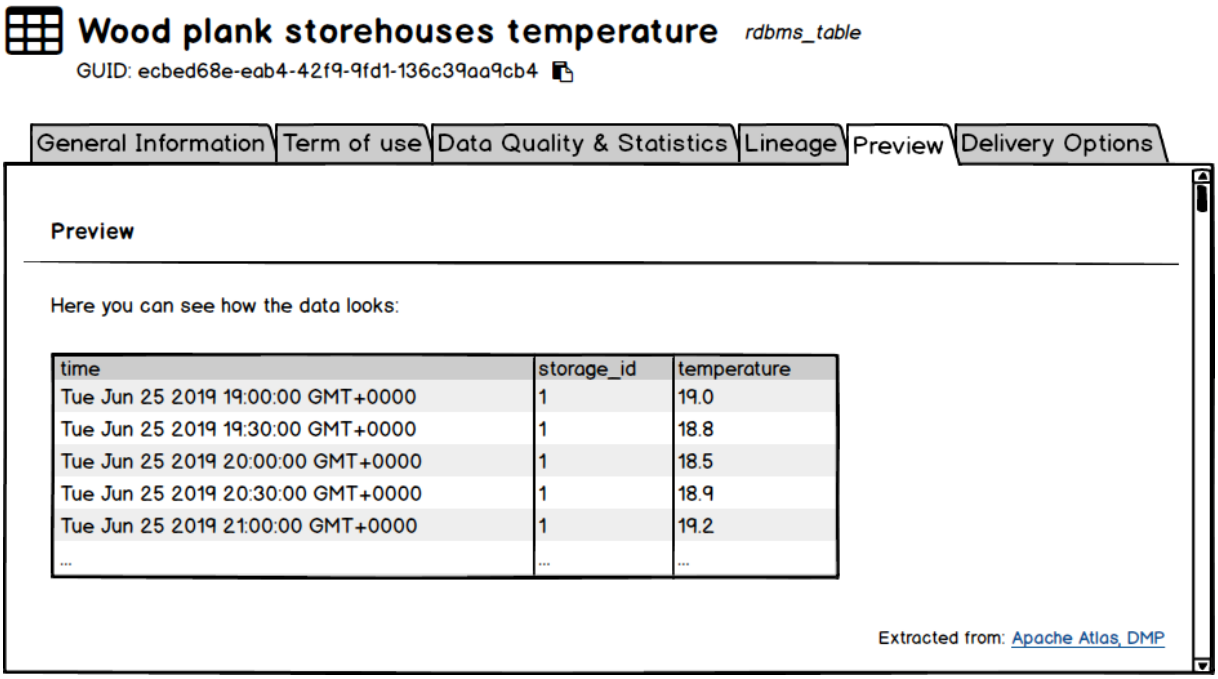
Mockup 6. Data Quality & Statistics



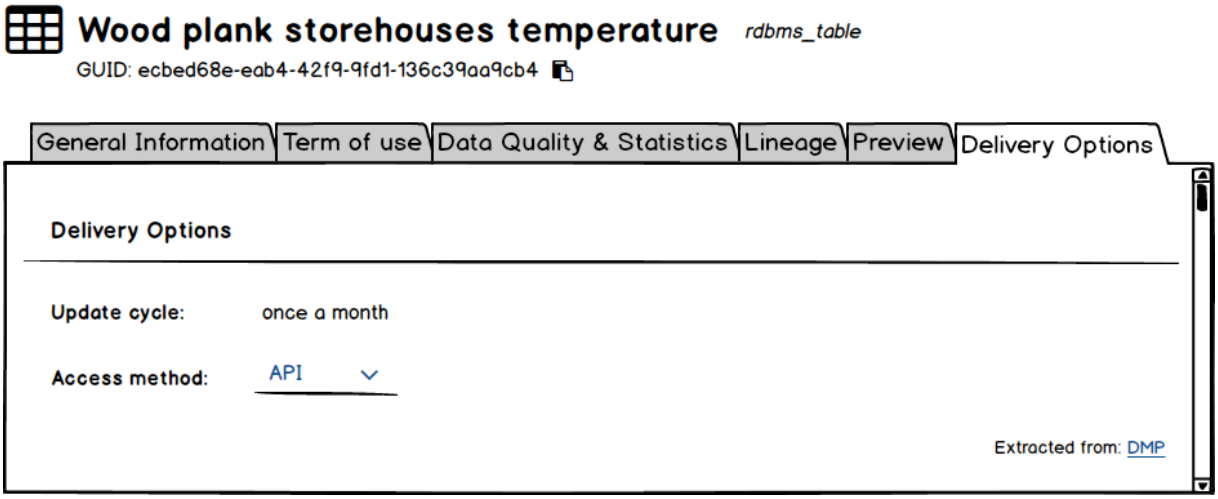
Mockup 7. Lineage



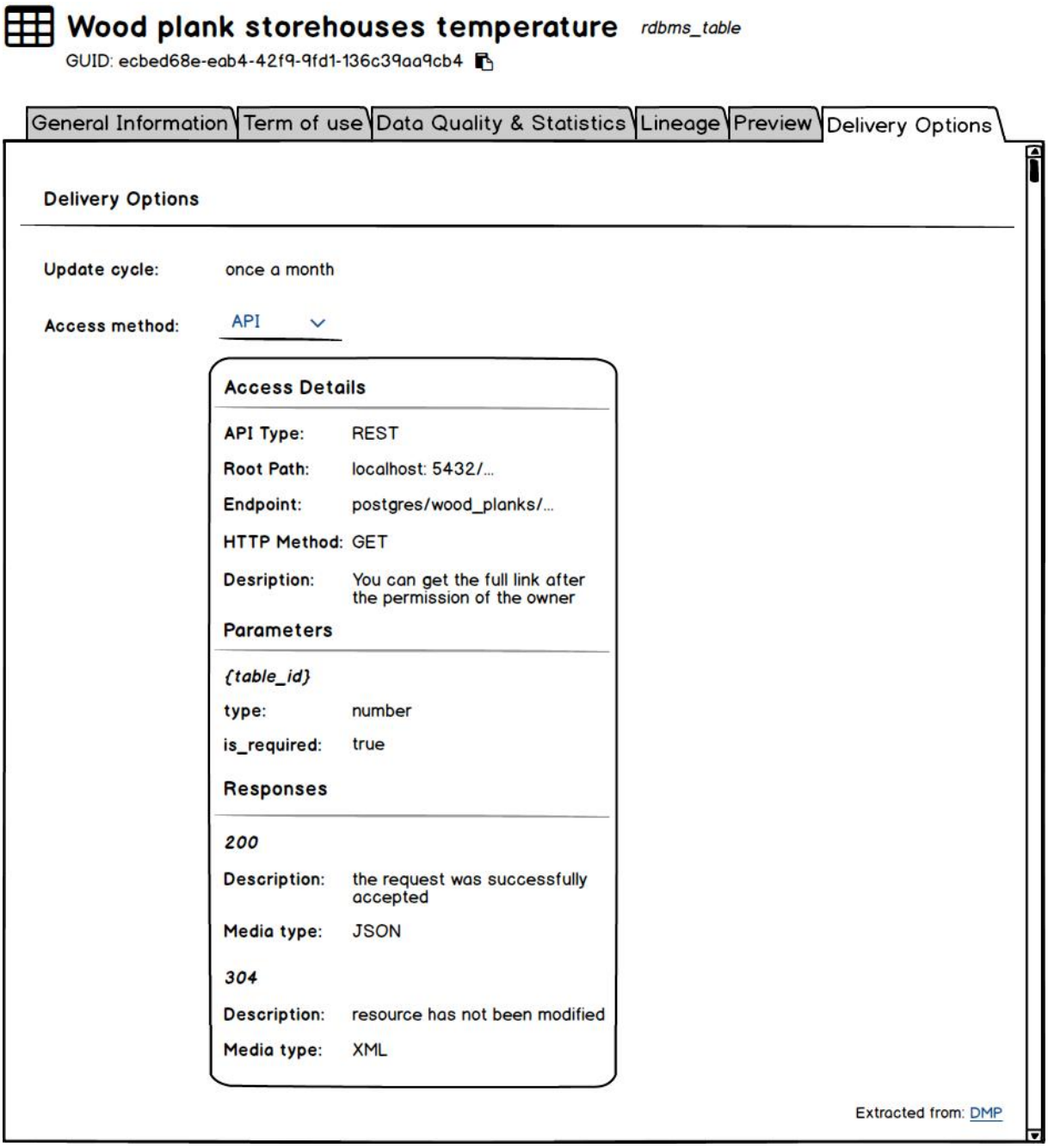
Mockup 8. Preview



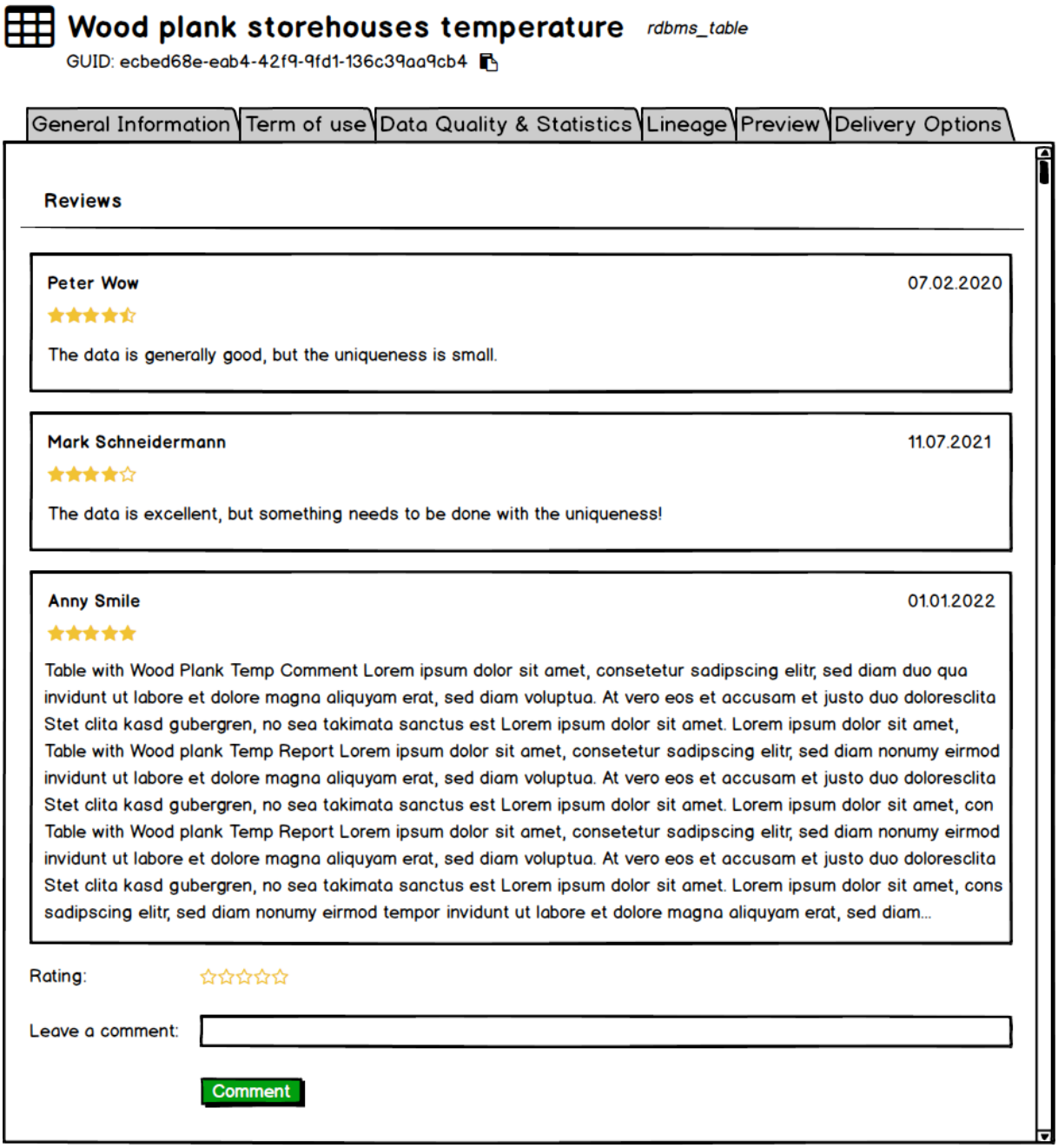
Mockup 9. Delivery Options



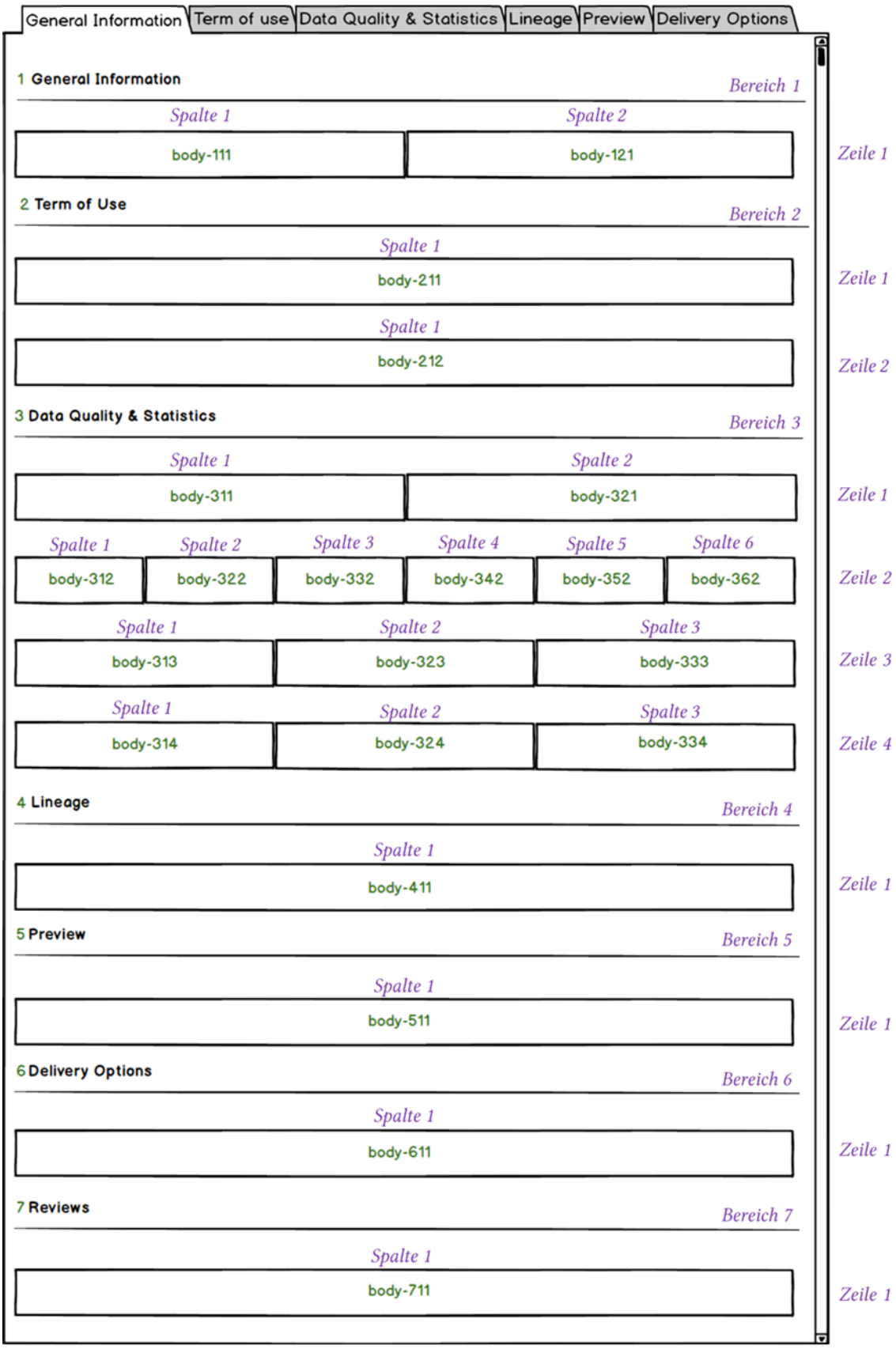
Mockup 10. API-Metadaten



Mockup 11. Reviews



Mockup 12. Layout für die Anordnung der Elemente im Frontend



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