Fakultät Informatik
Hochschule Reutlingen
Alteburgstraße 150
D-72762 Reutlingen

Masterthesis

Bestimmung der Dokumentenähnlichkeit basierend auf Bayessche Statistik für eine Big-Data Information Retrieval Lösung

Elisabeth Agnes Mpessa Enangue

Studiengang: Services Computing

Betreuer Hochschule: Prof Dr.-Ing Christian Decker
Betreuer Unternehmen: Steve Strauch
Abgabetermin: 31. Juli 2018

Abstract

In the last years Cloud computing has become popular among IT organizations aiming to reduce its operational costs. Applications can be designed to be run on the Cloud, and utilize its technologies, or can be partially or totally migrated to the Cloud. The application's architecture contains three layers: presentation, business logic, and data layer. The presentation layer provides a user friendly interface, and acts as intermediary between the user and the application logic. The business logic separates the business logic from the underlaying layers of the application. The Data Layer (DL) abstracts the underlaying database storage system from the business layer. It is responsible for storing the application's data. The DL is divided into two sublayers: Data Access Layer (DAL), and Database Layer (DBL). The former provides the abstraction to the business layer of the database operations, while the latter is responsible for the data persistency, and manipulation.

When migrating an application to the Cloud, it can be fully or partially migrated. Each application layer can be hosted using different Cloud deployment models. Possible Cloud deployment models are: Private Cloud, Public Cloud, Community Cloud, and Hybrid Cloud. In this diploma thesis we focus on the database layer, which is one of the most expensive layers to build and maintain in an IT infrastructure. Application data is typically moved to the Cloud because of , e. g. Cloud bursting, data analysis, or backup and archiving. Currently, there is little support and guidance how to enable appropriate data access to the Cloud.

In this diploma thesis the we extend an Open Source Enterprise Service Bus to provide support for enabling transparent data access in the Cloud. After a research in the different protocols used by the Cloud providers to manage and store data, we design and implement the needed components in the Enterprise Service Bus to provide the user transparent access to his data previously migrated to the Cloud.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

List of Listings

1. Einleitung

1.1. Motivation

Im unternehmerischen Umfeld werden immer mehr großen Daten in einer oder mehreren Datenbanken gespeichert, verarbeitet und später ausgewertet, abhängig von dem verbundenen Zweck. Plattner behauptet in [Pla17], dass Big Data ein Synonym für die Bedeutung großer Datenvolumen in verschiedensten Anwendungsbereichen sowie der damit verbundenen Herausforderung diese verarbeiten zu können, ist. So zitieren Fasel und Meier nach [Mer11], dass Big Data definiert wird als Daten, die in ihrer Größe klassische Datenhaltung, Verarbeitung und Analyse auf konventioneller Hardware übersteigen [uAM16]. Es besteht eine im Big Data eine Vielfalt an Daten, die sich voneinander unterscheiden. Mit dem immer wachsenden Datenvolum kann das Finden von bestimmten Daten jedoch mühsam und zeitaufwendig sein.

Die Speicherung von großen Datenvolumen in relationalen Datenbanken kann Schwierigkeiten bereiten, bzw. wenn diese auf mehreren physischen Maschinen erfolgt. Es werden NoSQL-Technologien verwendet bei Unternehmen, die webbasiert sind. Die flexible Gestaltung und schnelle Änderung der Datenformate ist außerdem auch möglich durch die Nutzung von externe Quellen wie Webservices.

Durch digitale Transformation wollen sich Unternehmen entwickeln. Diese Entwicklung läuft über die Speicherung immer großer Datenmenge (Datensee). Jedoch erleichtert nicht dieser Datensee die Analyse und Informationsgewinnung. Die traditionelle Suche und unscharfe Informationsrückgewinnung bieten sich an um dieses Problem zu lösen. Diese sind aber nicht optimal, da sie nur als Basis vorhandenes Wissen haben. Eine Bremse zu der digitalen Transformation von Unternehmen ist das Vorhandensein eines Haufens von unbekannten Informationen, die sich qualitativ und auf Relevanz unterscheiden. Die Strukturierung der Information nach Inhalt und Bedeutung stellt sich als notwendig. Daher kann eine Softwarelösung, die Daten aus unterschiedlichen Datenbanken auf Inhalt vergleicht, eingestellt werden.

Eine Information Retrieval Middleware-Lösung deren Namen BigData4Biz ist, wird von der Firma Dibuco GmbH entwickelt. Diese Softwarelösung beruht auf einem Konzept der mehrdimensionalen Strukturierung eines Datensees unter Verwendung verschiedener Begriffe der Datenähnlichkeit. Wobei die Datenähnlichkeit entweder geschäftsbezogen oder Geschäft agnostisch sein kann und beruht auf linguistische Aspekte, die berechnet werden unter anderem mit Berücksichtigung der Begriffsfrequenz und -bedeutung. Die Berücksichtigung der Begriffsfrequenz und -bedeutung hilft dabei Informationen im Datensee zu gewinnen und zu entdecken.

Eine Programmierschnittstelle ermöglicht es Abfragen in einer GUI zu formulieren um die Informationen zu gewinnen und zu entdecken. Angesichts der großen Menge an Daten eine Datensuche mit Benutzung von Sätze und Schlüsselwörter für eine Abfrage ist unmöglich. Unter Annahme, dass die Benutzer ganz unbewusst sind, was die Anzahl und die Art der Informationen und Daten angeht, BD4B bittet die Option für Benutzer bekannte Informationen zu sehen und abzufragen, sodass alternative und ähnliche Informationen als Ergebnisse angezeigt werden, so nach Relevanzgrad und mit einer immer engen Einschränkung bei der Suche.

Im Kontext von großen Datenmengen, die immer mehr wachsen, ist die Skalierbarkeit sehr wichtig. Die vorliegende Arbeit setzt sich an diesem Punkt an und untersucht sowohl die unterschiedlichen Ansätze zur Ähnlichkeitsbestimmung, die auf Bayesschen Statistik basieren als auch Ihre Skalierbarkeit. Ziel ist es der bestmögliche Ansatz zur Dokumentähnlichkeit zu finden um eine optimale und adaptierte Datensuche in BigData4Biz zu ermöglichen.

1.2. Problemstellung

Der Aufwand bei der Bestimmung der Dokumentähnlichkeit für mehrere Quellen ist in der Regel mit hohem Zeitaufwand verbunden. Vor allem die Bestimmung der Ähnlichkeit in einem Haufen von Daten, die sich entweder im Internet befinden oder in gegebenen Datenspeicher. Hier setzt eine Informationsrückgewinnungslösung an. Diese versucht die Bestimmung der Ähnlichkeit zwischen Dokumente durchzuführen. Dafür gibt es mittlerweile eine Vielzahl an Ansätze. Je mehr die Übereinstimmung sein soll, desto höher wird der Aufwand sein. Darüber hinaus ergibt sich, dass für die Bestimmung der Dokumentähnlichkeit zwei sich gegenseitig gegenüberstehende Herausforderungen beeinflussen, nämlich der Bestimmungsaufwand und den Wunsch Dokumente mit der höchsten Übereinstimmung im gesamten Datensee zu finden. Da dadurch die Nutzer zufrieden sein können. Außerdem kann es passieren, dass Dokumente nicht unbedingt zutreffen, obwohl die als ähnlich bestimmt wurden. Faktoren wie den Kontext, die Semantik sowie die Bedeutung müssen berücksichtigt werden. Um dieses Problem zu lösen, soll einen bestimmten Ansatz basierend auf Bayessche Statistik ausgewählt werden, der diese Faktoren berücksichtigt. Die vorliegende Arbeit untersucht auf Bayessche Statistik basierende Ansätze zur Dokumentähnlichkeitsbestimmung.

1.3. Zielsetzung

Das Ziel dieser Masterarbeit ist es, die Dokumentähnlichkeit zu bestimmen für eine Big Data Informationsrückgewinnungslösung auf Basis von Ansätze, die auf Bayesschen Statistik basieren. Für die zuvor genannten Herausforderungen der Informationsrückgewinnungslösung, sollten in der vorliegenden Arbeit unterschiedliche Ansätze untersucht werden. Die in dieser Arbeit Dokumentähnlichkeitsbestimmung soll so erfolgen, dass die oben genannte Probleme gelöst werden.

Die Bayessche Statistik bietet Ansätze zur Dokumentähnlichkeitsbestimmung im Bereich der Informationsrückgewinnung, die angewendet und adaptiert zu gegebenen Fälle werden. Die Big Data Informationsrückgewinnungslösung kann zudem unterschiedlichen Datenquellen benutzen. Für die vorliegende Arbeit wird als Datenquelle hier ein Datenspeicher benutzt. Aus diesem Grund gilt es Ansätze zur Dokumentähnlichkeitsbestimmung basierend auf Bayessche Statistik zu analysieren Nach der Auswahl optimaler Ansätze wird eine Teilmenge der ausgewählten Ansätze implementiert. So können die realisierten Ansätze evaluiert werden auf Skalierbarkeit, das komplette auf Basis von ausgewählten Stichprobendaten

1.4. Aufbau der Arbeit

Die vorliegende Arbeit besteht aus sieben Kapiteln. Das Kapitel 1 befasst sich mit der Einleitung zum Thema dieser Arbeit, wo die Problemstellung, Zielsetzung sowie der Aufbau der Arbeit erläutert werden.

Das Kapitel 2 behandelt die Grundlagen, die Analyse von Use Cases zur Dokumentähnlichkeitsbestimmung sowie auf Bayessche Statistik basierende Ansätze zur Dokumentähnlichkeitsbestimmung. In Kapitel 2.1 werden Grundlagen der Informationsrückgewinnung gegeben, nämlich die Definition des Begriffs Informationsrückgewinnung. Weitere wichtige Aspekte der Informationsrückgewinnung wie ihr Nutzen, ihr Mechanismus, ihre Modelle sowie ihre Web Variante werden erläutert. Kapitel 2.2 erläutert die Grundlagen der Bayesssschen Statistik mit unteren Punkte über die Definition, der Hintergrund, die Anwendungsgebiete, der Nutzen, Modelle, Parameter, Überzeugungen der Bayessche Statistik, sowie die Wahrscheinlichkeit und der Satz von Bayes. Das Kapitel 2.3 beschäftigt sich mit der Analyse von Use Cases für die Dokumentähnlichkeitsbestimmung. Beim Kapitel 2.4 wird die Analyse existierender Ansätze zur Dokumentähnlichkeitsbestimmung basierend auf Bayesschen Statistik.

Das Kapitel 3 stellt die Informationsrückgewinnung-Middlewäre-Lösung vor. Dabei wird ihr Nutzen, Ihre Anforderungen, Fähigkeiten und Funktionsweise erläutert.

Im Kapitel 4 werden Ansätzen zur Dokumentähnlichkeitsbestimmung in der Informationsrückgewinnung-Middlewäre-Lösung ausgewählt.

Im Kapitel 4.1 werden Auswahlkriterien für Ansätzen zur Dokumentähnlichkeitsbestimmung erläutert und in Kapitel 4.2 wird die Funktionsweise des ausgewählten Ansatzes erläutert.

Im Kapitel 5 wird die Implementierung einer Teilmenge ausgewählten Ansätze erläutert.

Im Kapitel 6 wird die Skalierbarkeit der realisierten Ansätze evaluiert durch die Benutzung von ausgewählten Probedaten. Das Kapitel 6.1 erläutert die Fähigkeiten und Grenzen dieser Ansätze.

Die vorliegende Arbeit wird mit dem Kapitel 7 durch eine Zusammenfassung und einen Ausblick abgeschlossen.

2. Grundlagen, Analyse von Use Cases zur Dokumentähnlichkeitsbestimmung

2.1. Grundlagen der Informationsrückgewinnung

2.1.1. Definition

Die Informationsrückgewinnung ist eine Aktivität bei der Material (Dokumente z. B) in einer unstrukturierten Natur (Text Beispielsweise) gefunden wird, um ein Informationsbedürfnis innerhalb von großen Ansammlungen, die auf einem Speicher gespeichert sind, zu erfüllen [CDM08] . Bei der Informationsrückgewinnung werden Informationsobjekte abgebildet, gespeichert, gestaltet und zugegriffen [RBY99].

2.1.2. Nutzen

Da das Internet immer mehr genutzt wird und die meisten Anwender bei Suchmaschinen oder E-Mail Dokumente abrufen wollen wird die Informationsrückgewinnung immer mehr gebraucht und angewendet. Diese unterscheidet sich von der traditionellen Datenbanksuche und wird beliebter als Form der Informationszugriff. Die Informationsrückgewinnung wird angewendet um Daten- und Informationsprobleme zu lösen. Sie vereinfacht sie die Semi-strukturierte Datensuche wie Z.B das Finden eines Dokuments wo die Überschrift Java enthält und der Inhalt Threading [CDM08]. Die Informationsrückgewinnung wird auch angewendet um Benutzer zu unterstützen bei der Durchsuchung sowie Filterung von Dokumentensammlungen und um den Satz abgerufener Dokumente weiter zu verarbeiten.

2.1.3. Mechanismus

Die Informationsrückgewinnung erfolgt durch eine Software, die für den entsprechenden Zweck hergestellt wurde. Mithilfe einer Software Architektur kann das Mechanismus der Informationsrückgewinnung beschrieben werden. Wichtige Elemente dieser Software Architektur sind eine Datenbank, die die Materialen (Dokumente mit Texten als Inhalt) enthält, ein Datenbank Manager Modul, eine Anwenderschnittstelle, Text Operationen, Abfrageoperationen, die Suche, der Rang und die Indexierung [RBY99]. Der Prozess der Informationsrückgewinnung beginnt sobald der Anwender sein Bedarf an Informationen in Form einer textuellen Abfrage(Schlüsselwörter) spezifiziert hat durch die Anwenderschnittstelle. Dieser textuellen Abfrage werden analysiert und durch die Text Operationen transformiert. Text Operationen generieren daneben eine logische Sicht der textuellen Abfrage. Die textuelle Abfrage wird danach von Datenbank Manager indexiert, bzw. es wird eine sogenannte

invertierte Datei erzeugt. Die vorverarbeitete Abfrage wird von Abfrageoperationen in eine Systemebene Darstellung weiter transformiert. Die Informationsrückgewinnung beginnt dann sobald die textuelle Abfrage indexiert wurde. Es erfolgt dafür die Ausführung der textuellen Abfrage über eine Dokumentenquelle um den Abruf einer Menge relevanter Dokumente. Die Abfrageverarbeitung kann schnell erfolgen mithilfe der zuvor aus den Dokumenten in der Dokumentquelle erstellte Indexstruktur. Die abgerufenen Dokumente werden entsprechend Ihrer Relevanz geordnet bevor sie zum Anwender gesendet werden. Eine Untersuchung des Satzes von rangierten Dokumenten auf nützliche Informationen sowie die Erstellung eines Anwender Feedback können dann vom Anwender durchgeführt werden. Die Abbildung 2.1 zeigt der Mechanismus der Informationsrückgewinnung, wo das Zusammenspiel von Software Architektur Komponente dargestellt ist.

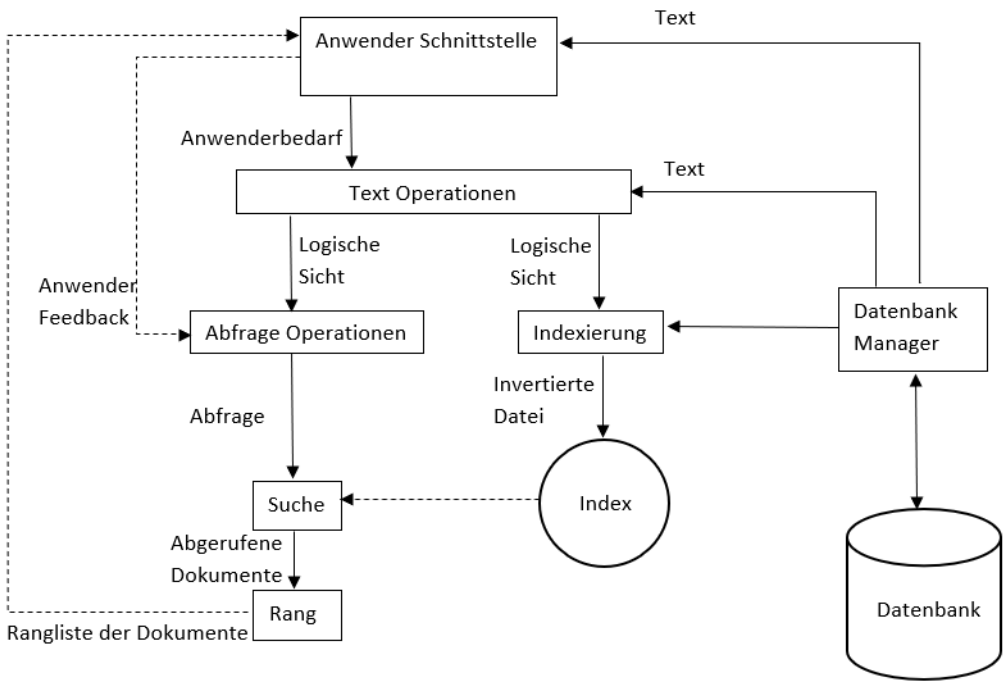

Figure 2.1.: Der Mechanismus der Informationsrückgewinnung [RBY99]

Die Informationsrückgewinnung wird auf Basis von Modelle durchgeführt. Was das Konzept dieser Modelle ist und wie diese funktionieren wird es zunächst erwähnt.

2.1.4. Informationsrückgewinnungsmodelle

Modelle sind charakterisiert durch einen Zweck sowie eine Art und werden definiert als Abbild eines realen Systems oder Problem. Im Fall der Informationsrückgewinnung werden Modelle benutzt um Vereinfachungen zu finden und abstrakt die Zusammenfassung oder die Vernachlässigung von Elemente zu bestimmen. Es bestehen drei Informationsrückgewinnungsmodelle, nämlich das boolesche Modell, das Vektorraummodell und die probabilistischen Modelle.

(1) Das boolesche Modell

Das boolesche Modell, ist ein Modell, das beruht auf Mengenlehre und boolesche Algebra. Detaillierter ausgedrückt, bei dem booleschen Modell Abfragen können in Form eines booleschen Ausdruckes von Termen formuliert werden, bzw. Terme werden mit den Operatoren UND (Konjunktive Abfrage), ODER (Disjunktive Abfrage) oder NICHT (Negative Abfrage) verbunden. Wegen dem intuitiven Charakter des Konzeptes einer Menge, stellt das boolesche Modell einen leicht zu verstehenden Rahmen für einen gewöhnlichen Benutzer. Des Weiteren Abfragen werden als boolesche Ausdrücke mit präzise Semantik beschrieben. Da das boolesche Modell einfach und formell ist, wurde er in den vergangenen Jahren wahrgenommen und von vielen der frühen kommerziellen bibliografischen Systeme angeeignet. In das boolesche Modell werden Dokumente als eine Menge von Indexbegriffe bezeichnet. Außerdem Indexterme Gewichte sind binär. Das boolesche Modell entscheidet ob ein Dokument relevant oder irrelevant ist für eine gegebene Abfrage. Eine teilweise Übereinstimmung mit dem Dokument wird nicht toleriert.

(a) Das boolesche Modell bietet den Vorteil einfach zu sein und ein sauberer Formalismus in seiner Struktur.

(b) Ausdrücke haben eine präzise Semantik, die sie für strukturierte Abfragen geeignet macht, die von "Experten" -Benutzern formuliert werden [SC13].

Es bestehen jedoch auch Nachteile für dieses Modell. Der Hauptnachteil ist, dass die Eigenschaft der totalen Übereinstimmung bei der Informationsrückgewinnung dazu führt, entweder wenige oder zu viele Dokumenten als Ergebnis der Suche zu bekommen. Etwas, das den Anwender die Formulierung guter Abfragen erschwert. Eine Lösung zu dem o. g Nachteil ist die Koordinationsstufe. Mit der Koordinationsstufe können Aussagen aus atomaren Aussagen nichtbinäre Werte haben, bzw. Ranking Dokumente über die reale Linie [Mel15]. Das heißt die binäre Werte des booleschen Modells kein Dokumentenranking ermöglichen bei abwesender Koordinationsstufe.

Die Koordinationsstufe ist ein Maß für den Grad zu dem jedes zurückgegebene Dokument der Abfrage entspricht [Mel15]. Diese stellt eine Punktzahl für das Ranking der Dokumente bereit. Dieses Ranking ermöglicht es dem Benutzer eine Entscheidung über die Menge der zu prüfenden Dokumente zu treffen und gibt dem System die Möglichkeit, den Dokumentenwert, der nach unten gerankt ist, von der Liste abzuschneiden. Die Kalkulation der Koordinationsstufe erfolgt durch die Transkription einer booleschen Abfrage in Konjunktive normale Form (KNF), bzw. durch die Konjunktionsoperator gebundenen Propositionen Liste. Jeder dieser Propositionen präsentiert sich als Disjunktionen von Atom Vorschläge. Die Koordinationsstufe eines Dokuments entspricht die Anzahl der Vorschläge, aus denen eine KNF durch das Dokument zusammengesetzt ist [Mel15].

Bei der booleschen Informationsrückgewinnung wird die Gewichtung von Termen nicht bestimmt. Es ergibt sich infolgedessen zu kleine oder zu große Ausgabe [RBY99]].

Wegen diesem Problem wird das boolesche Model in moderne Informationsrückgewinnung nicht mehr eingesetzt. Eine Alternative ist die Erweiterung des booleschen Modells um die Funktionalität von teilweise Übereinstimmung und Begriffsgewichtung. Um die zukünftige Berücksichtigung der variablen Größe zu ermöglichen, wurde eine Variation der Koordinationsstufe dessen Name gewichtete Koordinationsstufe ist, eingesetzt. Bei der gewichteten Koordinationsstufe erfolgt die Zuweisung eines anderen Gewichts abhängig vom Vorschlag, anstatt der Zuweisung konstantes Gewicht zu jedem vom Dokument wahr gemachter Vorschlag (?Satz Formulierung [Mel15] S8??).

Andere Erweiterungen des booleschen Modells unterstützen die Informations-Nähe und Distanz. Mit diesen Erweiterungen kann spezifiziert werden, ob zwei Begriffe in einer Abfrage in einem Dokument nahe beieinander erscheinen dürfen. Es ist möglich die Nähe zu messen durch Begrenzung der Anzahl von dazwischenliegenden Wörtern oder durch Referenzieren auf eine strukturelle Einheit wie ein Satz oder ein Paragraph (Rock NEAR Roll) [SC13].

(2) Das Vektorraummodell

Das Vektorraummodell wird charakterisiert durch den Begriff Ähnlichkeit. Um den grenzenden Aspekt der binären Gewichtseinheiten zu überwinden, bietet das Vektorraummodell einen Rahmen, bei dem die teilweise Abstimmung toleriert wird. Diese teilweise Übereinstimmung erfolgt durch die Zuweisung nicht binäre Gewichtseinheiten zu Indexbegriffe in Abfragen und Dokumenten. Die Begriffsgewichte berechnen den Ähnlichkeitsgrad von jedem Dokument, das gespeichert ist im System und die Anwenderabfrage. Die Berücksichtigung der nur teilweise mit Abfragebegriffen übereinstimmende Dokumente erfolgt beim Ähnlichkeitsgrad durch das Sortieren in absteigender Reihenfolge der Dokumente, die abgerufen wurden. Was sich daraus ergibt ist eine viel genauere Ranglisten-Antwortmenge als die Dokumentenantwortmenge, die mit dem booleschen Modell abgerufen ist. Das Vektorraummodell bewertet das Ähnlichkeitsgrad eines Dokuments in Bezug auf die Abfrage als die Korrelation zwischen zwei Vektoren. Die Quantifikation dieser Korrelation erfolgt beispielsweise durch den Kosinus des Winkels zwischen diesen zwei Vektoren [RBY99]. Anstatt eine Vorhersage über die Relevanz eines Dokuments durchzuführen, das Vektorraummodell geht mit der Klassifizierung der Dokumente entsprechend ihrem Ähnlichkeitsgrad mit der Abfrage vor. Der Abruf eines Dokuments erfolgt dann nur wenn die Bedingung der teilweisen Übereinstimmung erfüllt ist. Die Festlegung eines Schwellenwertes für Ähnlichkeit ist beispielsweise möglich, und den Abruf der Dokumente mit einem Ähnlichkeitsgrad über diesem Schwellenwert ist machbar. Die Berechnung einer Rangliste ist möglich nur wenn die Indexbegriffsgewichte erhalten sind. Diese Indexbegriffsgewichte werden erhalten durch welche begriffsgewichte Techniken, die einen Bezug auf die Clustering-Techniken unterstützende Grundprinzipien.

Die Clustering-Techniken werden wie folgend beschrieben. Ein einfacher Clusteralgorithmus hat als Ziel die Trennung einer Sammlung S von Objekten in zwei Mengen, mit gegebener Sammlung S von Objekten und einer vagen Beschreibung einer Menge M: eine Menge, die Objekte mit Bezug auf die Menge M enthält und eine andere bestehend

aus nicht mit der Menge M verwandte Objekte. Die Bedeutung von vage Beschreibung ist, dass keine vollständige Informationen vorhanden sind, um die Entscheidung über anwesende Objekte in der Menge M zu treffen. Es ist möglich für anspruchsvollere Cluster-Algorithmen die Trennung von Objekte einer Sammlung in verschiedene Cluster nach ihren Eigenschaften zu bestreben. Bei einem Clusterproblem kommen zwei Hauptprobleme in Frage, nämlich die muss Ermittlung von Merkmale, die besser beschrieben werden von den Objekten, erfolgen und die Ermittlung von Merkmale, die die Objekte in der Menge M besser von den übrigen Objekten in der Sammlung S differenzieren. Das eine Merkmal setzt eine Quantifizierung der Intra-Cluster-Ähnlichkeit frei und das andere Merkmal erlaubt die Quantifizierung der Ungleichheit zwischen den Clustern.

Die Quantifizierung in die Intra-cluster-Ähnlichkeit erfolgt durch die Messung der Rohhäufigkeit eines Ausdrucks innerhalb eines Dokuments. Die Bezeichnung von so eine Begriffshäufigkeit heißt Faktor und die Begriffshäufigkeit misst wie gut die Beschreibung des Dokumenteninhalts ist. Die Quantifizierung der Ungleichheit zwischen den Clustern erfolgt durch Messung der Umkehrungshäufigkeit eines Ausdrucks unter den Dokumenten in der Sammlung. Laut [RBY99] das Vektorraummodell bietet die folgenden Vorteile an:

(a) Sein Begriffsgewichtungsschema verbessert die Suchleistung.

(b) Seine Strategie der teilweisen Übereinstimmung ermöglicht das Abrufen von Dokumenten, die den Abfragebedingungen angenähert sind.

(c) Die Cosinus-Rangliste-Formel sortiert die Dokumente nach ihrem Ähnlichkeitsgrad mit der Suchanfrage

Beim Vektorraummodell besteht auch einen Nachteil, nämlich, dass die Index Begriffe als voneinander unabhängig gelten. Die mögliche Beeinträchtigung der Gesamtleitung erfolgt aufgrund der wahllosen Anwendung von Indexbegriffe auf alle Dokumente in der Sammlung [RBY99].

(3) Die probabilistischen Modelle

Mit den probabilistischen Modellen werden Informationsrückgewinnungsprobleme mithilfe der Wahrscheinlichkeitstheorie bestimmt. Diese werden benötigt, um das Problem der schwierigen Anwendung des booleschen Modells in Informationsrückgewinnungsaufgaben zu überwältigen. Ein Beispiel von Probleme wäre fehlendes Ranking für Forscher oder fehlendes oder überlastetes Output für den Endbenutzer [Mel15]. Mit dem Vektorraummodell konnte durch Ranking die Verbesserung der Benutzererfahrung erfolgen. Jedoch fehlt immerhin die Ermittlung offenen linearen Koeffizienten. Die probabilistischen Modelle unterstützen ein System dabei die Dokumentdarstellung auf Relevanz zu prüfen und hilft mit Prinzipien bei der Bereitstellung und Verwendung von Gewichte der Koordinationsstufe. Probabilistische Modelle umfassen die Wahrscheinlichkeits-Ranking-Prinzip, das binäre Unabhängigkeitsmodell [CDM08], Sprachmodelle, das Relevanz Modell [Mel15]. Bevor das Thema des

Wahrscheinlichkeits-Ranking-Prinzips behandelt wird, ist es wichtig das Thema Wahrscheinlichkeitstheorie zu besprechen.

Wahrscheinlichkeitstheorie [CDM08]: Diese ist ein Feld der Stochastik, wo zufällige Ereignisse beschrieben und modelliert werden. Sie beginnt mit als Mengen aufgefasste und Wahrscheinlichkeiten zugeordnete Ereignisse. Die Wahrscheinlichkeiten in diesem Fall entsprechen reelle Zahlen zwischen 0 und 1. gegeben werden zwei Variablen A und B, die Ereignisse Repräsentieren, wobei die Wahrscheinlichkeit für die jeweiligen Ereignisse $0 \leq$ P (A) ≤ 1 und $0 \leq$ P (B) ≤ 1 abgefragt wird, da es unsicher ist ob, diese Variablen wahr sind in der reellen Welt. Das gemeinsame Ereignis der beiden Ereignisse wird durch die gemeinsame Wahrscheinlichkeit P (A, B) ausgedrückt. Der Ausdruck P(A | B) bedeutet die Wahrscheinlichkeit des Ereignisses A beim Auftritt des Ereignisses B. Die Kettenregel liefert die grundlegen Beziehung zwischen Kettenregel und die bedingte Wahrscheinlichkeit:

Teil mit Formeln muss noch hinzugefügt werden (wurde im Dokument Masterthesis_Mpessa.doc rot markiert am 10.06.2018)

Wahrscheinlichkeits-Ranking-Prinzip (WRP): Wenn die Antwort eines Referenzabrufsystems auf jede Anfrage eine Rangfolge der Dokumente in der Sammlung ist, in der Reihenfolge der abnehmenden Wahrscheinlichkeit der Relevanz für den Benutzer, der die Anfrage eingereicht hat, werden die Wahrscheinlichkeiten so genau wie möglich auf der Grundlage der Daten geschätzt dem System für diesen Zweck zur Verfügung gestellt wird, ist die Gesamteffektivität des Systems für seinen Benutzer die beste, die auf der Grundlage dieser Daten erhältlich ist [MEL15]. In einem einfachsten binären Fall des Wahrscheinlichkeits-Ranking-Prinzips, dessen Name 1/0 Verlust (engl. 1/0 loss) ist, bestehen keine Wiederauffindungskosten oder andere Versorgungssorgen zur unterschiedlichen Gewichtung von Fehler oder Aktionen. Das Prinzip dieses Falls ist einfach so, dass ein Punkt verloren wird, wenn ein nicht relevantes Dokument zurückgegeben wird oder ein relevantes Dokument nicht zurückgegeben wird. Gezielt wird die Rückgabe der bestmöglichen Ergebnisse als oberste Dokumente, die für den Nutzer wählbar sind. Nach dem Wahrscheinlichkeits-Ranking-Prinzip, die Einordnung der Dokumente muss in absteigender Reihenfolge erfolgen. Falls es um die Rückgabe einer Reihe von Abrufergebnisse anstatt einer Bestellung geht, die Bayes Optimale Entscheidungsregel minimiert das Verlustrisiko beim Zurückgeben von eher relevante als nicht relevante Dokumente. Das WRP ist wichtig, dass es verbindet der prinzipielle Ansatz zum Ranking und den Effektivitätsmaßen. Die Risiken, die im WRP sind probabilistische Definitionen von Rückruf oder Abfallquote. Ob die Maximierung des Rückrufs im Falle einer gegebenen maximal tolerierten Abfallquote relevant ist, wird es determiniert bei der Maximierung der Wahrscheinlichkeit Mithilfe dieser probabilistischen Sichtweise. Das WRP verweist auf die Optimierung der Wiedergewinnungseffektivität sobald der Rückruf für jedes feste Abfallquote-Kosten maximal wird. Das WRP kann auch mit Rückholkosten umgesetzt werden, wo die Modellierung der Differenzkosten von Falschpositiven und Falschnegativen durchgeführt wird.

Das binäre Unabhängigkeitsmodell[MRS08]: Traditionell erfolgt die Anwendung die-

ses Modells mit dem WRP. Durch Einführung einfacher Annahmen, ermöglicht das binäre Unabhängigkeitsmodell das Schätzen einer Wahrscheinlichkeitsfunktion zu konkretisie-ren. Binär bedeutet auch boolesch, wobei die Darstellung von Dokumente und Abfragen erfolgt als binäre Begriffsanfall Vektoren. Das heißt die Darstellung eines Dokuments erfolgt durch einen Vektor, der das Wert 1 hat Falls der Begriff im Dokument vorhanden ist oder hat das Wert 0 Falls der Begriff nicht vorhanden ist im Dokument. Das Wort „Unabhängigkeit“ drückt das unabhängige Vorkommen von Begriffe in Dokumenten aus. Die Erkennung einer Assoziation zwischen Begriffe bestehen bei dem Modell nicht. Trotz ihre Unkorrektheit, bietet diese Annahme befriedigende Ergebnisse und entspricht die Annahme von Naive-Bayes-Modellen. Diese Annahme ist auch gleichwertig mit einer Annahme des Vektorraummodells, wo jeder Ausdruck eine zu allen anderen Begriffen orthogonale Dimension entspricht. Die Verfeinerung des Informationsbedarfs von Benutzer erfolgt durch die Anzeige einer Reihe von Ergebnisse. Für eine präzise probabilistische Suchstrategie soll die Abschätzung des Beitrags zur Relevanz von Begriffe in Dokumenten erfolgen. Im binäre Unabhängigkeitsmodell erfolgen die Ableitung einer Ranking-Funktion für Abfragebegriffe, sowie die theoretischen und praktischen Wahrscheinlichkeitsschätzungen. Eine Erweiterung, dessen Name Baumabhängigkeitsmodell ist, versucht die Modellierung von Abhängigkeiten erster Ordnung zwischen Begriffe durch die Verwendung[MET11] .

Sprachmodelle [Mel15]:

Das Relevanz Modell [Mel15]:

Probability theory and ranking, binary independance model

2.1.5. Web Informationsrückgewinnung

2.2. Grundlagen der Bayesschen Statistik

2.2.1. Definition und Hintergrund

2.2.2. Anwendungsgebiete und Nutzen

2.2.3. Modelle, Parameter und Überzeugungen

2.2.4. Die Wahrscheinlichkeit

2.2.5. Der Satz von Bayes

2.3. Analyse von Use Cases für die Dokumentähnlichkeitsbestimmung

Die Dokumentähnlichkeitsbestimmung ist ein Verfahren, bei dem Dokumente auf Ähnlichkeit geprüft werden. Welches Ziel wird von diesem Verfahren verfolgt, hängt stark ab von der Problemstellung, bzw. was gesucht wird und wozu. Es bestehen dafür unter anderem ein Paar Use Case (Anwendungsfälle) und Bereiche bei denen das Verfahren angewendet und gebraucht wird. Unter diesen Use Cases können Finden von Dokumenten mit ähnlichen Inhalten (Duplikate), die Verwendung von a priori Wissen, die Systemübergreifende „Fremdschlüssel“, das Profil Matching und die E-Mail Klassifikation (Spamfilter). Es werden zunächst die oben genannt Use Cases vorgestellt, wobei die aktuelle Situation, Problemstellung sowie Lösung erwähnt werden.

2.3.1. Finden von Dokumenten mit ähnlichen Inhalten(Duplikaten Findung)

In einem Rechner kann abhängig von Speicherkapazität eine Unmenge von Daten gespeichert werden. Diese Speicherung besteht über Jahren im Computer und der Nutzer kann aus Versehen immer Ähnliche Dokumente in dem gleichen Rechner speichern. Es stellt sich danach ein Speicherplatz Problem, da der Rechner voll belegt mit Dateien aller Art ist. Um mehr freier Speicherplatz zu erwerben, wird beim Nutzer den Bedarf bestehen, nach Duplikate zu suchen und zu löschen im Rechner. Der Einsatz einer Information Retrieval Softwarelösung kann dieses Problem lösen. Es werden mit der Software Lösung alle Duplikate gesucht und gelistet, sodass der Nutzer die Möglichkeit hat, Dateien individuell zu löschen.

Eine andere Variante der Findung von Dokumenten mit Ähnlichkeit gibt einem Anwender die Möglichkeit unterschiedliche Dokumente über ein Thema zu finden, die Beispielsweise der gleiche Anfang und das gleiche Ende haben aber ein unterschiedlicher Inhalt. Der Anwender

kann dadurch neue Themen, die zu den Themen seiner aktuellen Dokumente verwandt sind, entdecken.

2.3.2. Verwendung von a priori Wissen

In einer Firma besteht eine Datenbank die die Wetterberichte für jeden Tag enthält und eine andere Datenbank, die die Verkaufszahlen pro Artikeln enthält. Um seine Produkte zu verkaufen und schon Voraus konsequent zu produzieren, möchte diese Firma, die beispielsweise Regenschirme verkauft, im Voraus wissen wann das Wetter günstig ist (schlechtes Wetter), um große Verkaufszahlen zu erhalten. Eine Information Retrieval Softwarelösung könnte dabei helfen ähnlichen Schwankungen der beiden Datenbanken oder von Filesysteme/Intranet der Firma herauszustellen und den Mitarbeiter dieser Firma verfügbar machen.

2.3.3. Systemübergreifende „Fremdschlüssel"

In einer Firma kommt es vor, dass Kundendaten gespeichert werden für unterschiedliche Abteilungen, wobei jede Abteilung eine eigene Datenbank hat, die unterschiedlich sein kann von anderen Abteilungsdatenbanken. Dieser Unterschied besteht auch bei Abfragesprachen dieser Datenbanken, sowie Primärschlüssel. Die Herstellung von Kundenverträge auf Basis einer Kundennummer kann problematisch sein, da Kunden nicht per Fremdschlüssel in beiden Datenbanken identifizier bar sind. Der Einsatz einer Information Retrieval Lösung wäre optimal, indem ähnliche Kundendaten in zwei verschiedene Datenbanken herausgefunden und deshalb zusammengebracht werden. Dadurch wird die Durchführung einer Datenbankmigration, die zeit- und ressourcenaufwändig ist, erspart.

2.3.4. Profil Matching

Bei einer Firma, die nach Fachkräfte sucht, kommt es vor, dass nachdem eine Stellenausschreibung veröffentlicht wird, viele Arbeitssuchenden sich darauf bewerben. Die Bewerbungen werden auf die Firma Datenbank gespeichert. Es geht meistens und Word oder PDF Dateien. Nun ist es so, dass die Firma sich Zeit sparen will und gleichzeitig die passenden Bewerber haben möchte. Eine Information Retrieval Softwarelösung löst dieses Problem, indem bestimmten Schlüsselwörter in Bewerberprofile durchgesucht werden um einen Match mit den Anforderungen der Ausschreibung zu finden.

Dieses Match kann auch genutzt werden um intern bei einer Firma geeignete Mitarbeiter für ein gegebenes Projekt zu finden.

2.3.5. Email Klassifikation (Spamfilter)

Spam-Mail sind E-Mails, die gesendet werden, ohne vorher verlangt zu sein. Diese können unerwünschte Werbungsemail sein, worauf Anwender kein Interesse haben. Die E-Mail Spam werden von besonderen Programme gesendet, wobei die gezielte Email Adressen von einer Software zufälligerweise erzeugt werden, oder aus Sammelprogramme. Eine Lösung für Spam E-Mail wäre ein Spam Filter oder eine Information Retrieval Lösung, die Techniken zur Erkennung von Spam nutzen wie die IP-Adresse, der Inhalt oder die Filterlisten [COM18].

IP-Adresse: Da die Spam-Emails vom Computer mit bestimmten IP-Adresse gesendet werden, können durch Erkennung dieser IP-Adresse die E-Mails als Spam bezeichnet und sortiert werden.

Inhalt: Die Spam E-Mail können auf Inhalt geprüft werden. Dafür werden die Betreffzeile und die Nachricht in sich auf bestimmte Schlagwörter überprüft.

Filterlisten: Spam Filter sind mit Schwarze und Weiße Listen versehen, bei denen Spamfiltermerkmale Schlagwörter und IP-Adresse sind. Die E-Mails, die Merkmale aus der Schwarze Liste ausstellen werden gelöscht und in einem Spam-Ordner verlagert oder als Spam markiert. Anderenfalls (Weiße Liste) werden sie im Posteingang verschoben. E-Mails mit Absender dessen Adresse in Adressenbuch gespeichert sind werden automatisch im Posteingangsordner verschoben, da diese darauf hinweist, dass die Absender und Empfänger gerne kommunizieren wollen.

2.4. Analyse existierender Ansätze zur Dokumentähnlichkeitsbestimmung basierend auf Bayesscher Statistik

2.4.1. More Like This

Der More Like This (MLT) Algorithmus ist, ein Algorithmus, der es ermöglicht Dokumente zu finden, die ähnlich sind zu den Dokumenten einer Ergebnisliste. Das Ganze läuft durch Relevanz-Scoring, wo Dokumente auf Relevanz geprüft werden. Um eine optimale Gestaltung des Relevanz-Scoring zu erzielen, das Algorithmus verringert die Anzahl an Kandidaten Dokumente durch einen Boolean Test, bei dem das gesuchte Dokument vergleicht wird mit der Abfrage. Die durch Boolean Test selektierte Dokumente werden Punkte erhalten. Die Bestimmung des Rangs für die Relevanz wird auf Basis diesen Punkten erfolgen. Ein Dokument kann trotz eine Übereinstimmung irrelevant sein für die Abfrage. Deswegen ist die Einstellung einer Filterung von Übereinstimmungen nach Index, dokumenttyp oder durch kontextueller Logik wichtig zur Lösung dieses Problems. Die Bewertungsfunktion der Informationsabrufsoftware-Bibliothek Lucene wird bei dem MLT Algorithmus genutzt. Diese entspricht ein auf der Begriffsfrequenz und der Inverse Dokumenthäufigkeit basierende Ähnlichkeitsmodell, das das Vektorraummodell für mehrere Begriffe Abfragen benutzt. Es sind unterschiedliche Ähnlichkeitsalgorithmen verfügbar:

(1) BM25 okapi: diese basiert auf die Begriffsfrequenz/inverse Dokumenthäufigkeit. Die Steuerung sowohl von nichtlineare Begriffshäufigkeitsnormierung als auch des Normalisierungsgrades von Begriffsfrequenz-Werte durch die Dokumentlänge.

(2) die klassische Ähnlichkeit: basiert auch auf Begriffsfrequenz/inverse Dokumenthäufigkeit. Sie bestimmt, ob Überlagerungstoken bei der Berechnung der Norm ignoriert werden[ELA18].

(3) die Abweichung von Zufälligkeitsähnlichkeit

(4) Abweichung von der Ähnlichkeit der Unabhängigkeit,

(5) Informationsbasierte Ähnlichkeit

(6) Skriptähnlich Ähnlichkeit: die Verwendung eines Skriptes zur Angabe wie die Ergebnisse berechnet werden sollen [ELA18].

(7) LM Jelinek Mercer Ähnlichkeit: die Erfassung von wichtigen Muster im Text mit Rücksichtlosigkeit von Rauschen.

(8) LM Dirichlet Ähnlichkeit

2.4.2. Naive Bayes

2.4.3. BayesLSH

3. Die Informationsrückgewinnung-Middleware-Lösung

3.1. Vorstellung

BigData4Biz wird von der Firma Dibuco GmbH seit April 2017 entwickelt. Diese ist für die Informationsrückgewinnung versehen und ermöglicht es den Nutzer sowohl relevante Informationen zu erkennen als auch gebrauchte Informationen zu gewinnen in einem Datensee. BigData4Biz sollte zukünftig die Möglichkeit anbieten, Ähnliche Dokumente zu bestimmen auf Basis von Ansätze, die auf Bayesscher Statistik basieren. Um dies zu ermöglichen sollten erstmal Anforderungen an diese Softwarelösung erläutert werden.

3.2. Anforderungen

BigData4Biz muss bestimmte Eigenschaften haben und eine bestimmte Leistung erbringen. Für BigData4Biz es bestehen unterschiedliche Anforderungen, nämlich funktionelle Anforderungen, nicht funktionelle Anforderungen, Plattform Anforderungen, Ähnlichkeitsanforderungen, Operationsanforderungen und Geschäftsanforderungen [Dib18].

3.2.1. Funktionelle Anforderungen

Entitäten entsprechen strukturierte Geschäftsobjekte von Metadaten und Eigenschaften, die assoziiert sind mit weiteren technischen Attributen. Im Falle von gemeinsamen Formate müssen einige Anforderungen erfüllt werden: die Eigenschaftsnamen müssen so vergeben werden, dass die genau gleich sind wie die entsprechenden Werten in der Datenquelle. Außerdem die Lokalisierung von den ursprünglichen Wert in den Daten einer Datenquelle (Datensatz, Datei usw.) aus dem Namen der Eigenschaft müsste möglich sein.

Was Entitätstechnischen Attributen angeht, die technische Attributen müssen vom Entitätsmodell abgetrennt werden um erstens sowohl eine Reduzierung der Komplexität zu erzielen als auch der Geschäftsteil vom internen technischen Teil der gesamten Entitätsdaten zu unterscheiden und zweitens die Behandlung der Persistenz technischer Attribute getrennt von der Entität zu erzielen. Die Werte von relationalen Attribute müssen eindeutig sein für alle Entitäten um eine genaue Zuordnung von den relationalen Services zu erzielen [Dib18].

Was die Beschaffenheit angeht, der gegenseitige Überlauf von zwei schnell auftretende Entitätsaktualisierung oder -löschung in der Lastverarbeitung muss vermieden werden. Die Vergabe von Ladezeitstempel dient zur Vermeidung solcher Situationen. Im Falle von mehreren Instanzen desselben Agenten für Lastausgleichs- oder Hochverfügbarkeitsgründe muss dann

garantiert werden, dass eine zeitlich bestmögliche Synchronisierung dieser Agenten durchgeführt wird. Außerdem muss die Anwendung einer Entitätsaktualisierung auf eine zuvor gelöschte Entität muss vermieden werden.

Nachdem die funktionellen Anforderungen erläutert wurden, gilt es zunächst die nicht funktionelle Anforderungen an BigData4Biz zu nennen.

3.2.2. Nicht funktionelle Anforderungen

Unter den funktionellen Anforderungen können Eigenschaften wie Beharrlichkeit, Belastbarkeit, Skalierbarkeit berücksichtig werden.

Was die Beharrlichkeitstechnologie einer Entität an geht, es müssen einige Anforderungen erfüllt werden. Erstens müsste eine effiziente Rohspeicherung von Entitäten nach Entitäten ID geben, zweitens müssen Daten konfigurierbar repliziert werden, drittens muss es Optionen für den Betrieb mehrerer Datencenter mit automatischer Replikation geben. Weiterhin eine robust hohe Verfügbarkeit, Betriebsunterstützung, Überwachung, Backup sowie Fehlerkorrektur müssen möglich sein.

Alle Dienste müssen bei Bedarf hochverfügbar sein. Alle erkannten Dienste müssen auf Integrität geprüft werden, und diese Integritätsprüfung muss sowohl intern als auch extern für Überwachungszwecke verfügbar sein [Dib18]. Ein Neustart muss durchgeführt werden für fehlgeschlagene Dienste oder die Verschiebung von neuen Instanzen auf den fehlerfreien Server muss erfolgen. Außerdem die Bereitstellung einer Programmierschnittstelle für den internen Status des Knotens durch jeden Dienst muss erfolgen. Durch einen einzelnen Bezugspunkt muss es eine strukturierte Zusammenfassung vom Status der gesamten Plattform bereitgestellt werden, mit einer optionalen Auswahl eines Themas wie Verfügbarkeit, Gesundheit und Leistungsüberwachung. Das Verhalten der Plattform sollte wie ein gigantisches Cluster sein aus der Sicht der Außenwelt.

Was die Skalierbarkeit angeht, die Bereitstellung der manuellen horizontalen Skalierbarkeit muss so einfach wie möglich erfolgen durch den Start und die automatische Entdeckung von neuen Instanzen, sowie die Einbeziehung des Routings.

3.2.3. Plattformanforderungen

Bei der Plattform wird sowohl von der Architektur als auch von der Entitätsverarbeitung ausgegangen.

Die genaue Einhaltung der Prinzipien einer nativen Cloud-Architektur sollte erfolgen um es der Plattform zu ermöglichen, so viel Möglichkeiten für Architektur- und Betriebsentscheidungen wie möglich auszuwählen[DIB18]. Im Falle einer nicht gewählten Ausführungsplattform-Technologie, die sich nicht zu den Microservices eignet, sollte ein Microservice-Design von Komponenten als Norm dienen. Alle Dienste müssen eigenständig sein, bzw. zentrale

Dienste, die von den anderen Diensten abhängig sind, müssen gemieden werden. Ein Dienstentdeckungsdienst muss vorgesehen werden, mit dem das automatisierte Routing ohne manuelle Konfiguration sowie die automatische Registrierung aller Plattformdienste möglich ist. Die Erkennung von neuen Dienste wie benutzerdefinierte Umwandlungen ist erforderlich. Außerdem die Übermittlung von routing- oder ausführungsrelevante Informationen über den Dienst zum Zeitpunkt der Dienstermittlung ist auch erforderlich.

Die Über die Lade-API empfangenen Entitäten müssen mindestens eine Verarbeitung erlebt haben und wenn Fehler bestehen kann die Verarbeitung bis zur maximalen Anzahl von Wiederholungen erfolgen. Entitätsverarbeitende Dienste sollen idempotent sein und sollten sowohl die Konsistenz als auch die Qualität der Ergebnisse nicht beeinflussen, im Falle von mehrfach Wiederholung einer Entitätslast. Die Konfiguration einer beliebigen Zahl von Versionen der Transformationsdienste, dessen Anwendung auf alle Entitäten erfolgt, muss möglich sein. Eine Teilreihenfolge von Umwandlungsdiensten muss an die Umwandlungskonfiguration teilnehmen.

In BigData4Biz soll eine linguistische Berechnung erfolgen. Daher werden folgend Anforderungen in Bezug auf diese genannt. Da die linguistische Berechnung auf neuen linguistischen Daten zugehen, die eindeutige IDs als eine raumhaltende Darstellung der Textdarstellung auf der gesamten Plattform brauchen, müssen diese IDs so erstellt werden, dass das Sperren gemieden wird. Angesichts des zeitaufwendigen Aspekts der linguistischen Berechnung, seine Ausführung sollte gleichzeitig für gerade verarbeitete Entitäten erfolgen. Die Konfiguration einer beliebigen Anzahl von Versionen der Ähnlichkeitsdienste, deren Anwendung auf alle Entitäten erfolgt, sollte möglich sein. Die Erkennung des Endes von gleichzeitigen Ausführung der Ähnlichkeitsdienste sollte robust sein. Für die Registrierung von Datenquellen und Agenten, liefert die Management-API einen Dienst zur Registrierung eines Agententyps, seine konfigurierte Datenquelle nach Namen und anderen Identifikationsinformationen. Die zusätzliche Identifikationsinformation muss genügend sein zur Identifizierung der tatsächlichen physikalischen Datenquelle sowie des von der Datenquelle abgedeckten Aspekts.

Außer eine Plattform BigData4Biz enthält auch Komponente, die bestimmte Anforderungen haben.

3.2.4. Komponentenanforderungen

BigData4Biz enthält folgende Komponente: Agenten, der Ladedienst, Transformationsdienste, der Entitätsdienst, der linguistische Dienst, der Ähnlichkeitsdienst, der Lebenszyklusdienst, der Benachrichtigungsdienst, der Löschdienst und das Kundenprofil.

Agenten sind Microservices, die zur Ermittlung von Informationen an die Plattform dienen. Diese können konfiguriert werden für eine Datenquelle, haben die Aufgabe Daten in einem einheitlichen Format (Entität) zu bringen und an die Plattform zu senden. Darüber hinaus sollten diese Agenten autonome Dienste sein, die sich um die Überwachung und Sendung der Datenquelleninhalte an die Lade-API kümmern. Eine Interaktion mit der Plattform sollte nicht zwangsläufig sein zur Ermittlung eines Entitätsstatus oder zur Erstellung einer Entität

ID. Die Verwaltung der eigenen Datenbank muss von Agenten durchgeführt werden um die Überprüfung der als Entität ID übertragenen Daten. Die Überwachung von jeder Datenquelle muss durch eine einzelne Agenteninstanz erfolgen. Es müsste eine Interaktion zwischen Agenten und Datenquelle geben, so dass die Ermittlung von neuen oder geänderten Daten möglich ist. Für neue Entitäten wird nach Möglichkeit eine Backlink-Information vergeben, derer Wahl datenspezifisch erfolgt. Weiterhin sollte es möglich sein mit dem Backlink der Ursprung von Entitäten in der physischen Quelle in Kombination mit der registrierten Datenquelle zu lokalisieren. Die Extraktion vom Textkörper einer Entität erfolgt nur dann, wenn die Daten eine Dateienart mit einem Textteil besitzen. Der eigentliche Text muss von allen technischen und strukturellen Elementen wie Sonderzeichen, Formatierung von Meta-Informationen befreit werden. Der aggregierte Text der Entität muss alle Textdaten in den Daten der Datenquelle enthalten. Der Verzicht auf Textinformationen in den Originaldaten und umgekehrt muss vermieden werden, sowie die Duplizierung von Textinformationen aus den Originaldaten im aggregierten Text. Strukturell muss eine bestimmte Reihenfolge erfolgen, nämlich erstmal den Textkörper und dann die Eigenschaftstexte. Wobei das Einfügen von Eigenschaftstexten muss als Satz erfolgen zur Erleichterung des Auftritts von Satzkooperationen. Da die Klassifizierung jeder Entität durch eine Reihe von Wörtern erfolgt, sollte die Haltung dieser Klassifizierung allgemein bleiben. Während die erste Klassifizierung die Benennung von Entität erstellende Agenten durchführt, die zweite Klassifizierung dient als „Typ“ der Entität, die immer vorhanden sein sollte. Die Kombinierung von Tabellen einer relationalen Datenquelle in einer Entität sollte möglich sein zur Vermeidung von atomare Verbindungen und Denormalisieren eines normalisierten Datenbankschemas. Die Identifizierung aller tatsächlichen Fremdschlüssel des Datensatzes muss vom Entität-Backlink durchgeführt werden zur Ermöglichung einer Teilung von Eigenschaften einer Entität auf ursprünglichen Tabellen. Da die relationalen Datenquellen Typen über ihr Schema bereitstellen, soll die Zuordnung übereinstimmenden Entitätstyps mit dem relationalen Datentyp erfolgen. Falls der Neustart eines Agenten erforderlich ist, muss er während seinem Stillstand Informationen über Datenänderungen oder Löschungen für die entsprechenden Entitäten liefern zur erneuten Überwachung der Datenquelle. Wenn für einen Agenten die Übergabe seiner Entität an die Lade-API unmöglich ist, sollte er die Speicherung und die Wiederaufnahme der Entität durchführen, sobald das Lade-API wieder operativ ist. Im Falle einer Ablehnung von einer Entität aus Verifizierungsgründen, muss für diese Entität ein separates Protokoll durchgeführt und einen erneuten Versuch muss vermieden werden. Technische Attribute dienen zur Speicherung von nicht auf der Client-Seite gegenüberliegenden Eigenschaften. Die Verwendung von strukturellem Wissen wie Eltern-Kind-Beziehungen oder Fremdschlüsselbeziehungen ist möglich zum Einfügen von technischen Attribute. Zur Erkennung des Attributes durch den jeweiligen Ähnlichkeitsdienst müssen die Eltern/Kind-Attributnamen festgelegt werden. Es bestehen Agenten für CSV Dateien, das Dateisystem, die RDB Dateien, Web Dateien und XML Dateien. Was die CSV Datei betrifft, kommt es häufig vor, dass diese leeren Spaltenwerte haben, die nicht berücksichtigt sein können. Die Implementierung sollte flexibel genug sein, dass diese leeren Spaltenwerte in CSV-Dateien nicht berücksichtigt werden. Die Extraktion von so viele Datei-Metadaten wie möglich als Eigenschaften muss erfolgen. Die Extraktion von Metadaten und des Textkörpers aus den Dateitypen docx, xlsx, pptx, rtf, txt, html, und pdf, muss durch den Dateisystemagenten möglich sein. Es müsste eine

Aufteilung des XML-Baumes unter Verwendung einfacher kundenfreundlicher Konfigurationswerte geben. Die Trennung von untergeordneten Elemente von einem Vorgängerelement muss möglich sein zur Erstellung von beiden separaten Entitäten. Für jede Unterteilung von Vorfahr und Kind muss ein global technischer Attributwert verfügbar sein, der den Ausdruck dieser aufgeteilten strukturellen Beziehung ermöglicht. Die XML-Elemente in roh Text sollten umgewandelt werden durch eine XSLT-Transformation, konfigurierbar durch einen Dateinamen entsprechenden regulären Ausdruck. Die Bereitstellung einer Standardumsetzung für nicht mit den konfigurierbaren Mustern übereinstimmende Dateien muss erfolgen. Diese Standardumwandlung müsste in der Lage sein, die Extraktion aller Elementinhalte als Nur-Text durchzuführen um deren Nutzbarkeit für die Textähnlichkeit zu ermöglichen. Zur Extraktion von bestimmten Elemente als Eigenschaften, ist es möglich, dass jede XML-Datei-XSLT-Transformation weitere Vorlagen umfassen. Diese Eigenschaften müssen als Teil der Extraktionsvorlage eingegeben werden. Eine mit einer konfigurierbaren Liste von Eigenschaftenextraktionsinformationen kombinierte generische XSLT-Transformation wird geliefert zur Bereitstellung der Extraktion von Nur-Text und Eigenschaften. Die vollständige XSLT-Transformation sollte nur in speziellen Fällen nötig sein. Die Zuordnung eines mit den Dateinamen übereinstimmenden regulären Ausdrucks für jede generische Transformation muss erfolgen.

BigData4Biz wird mithilfe des Frameworks Spring Boot implementiert. Zur Erleichterung der Implementierung von benutzerdefinierte Entitätstransformationsdienste muss die Bereitstellung eines Vorlagenprojekts mit Spring Boot erfolgen. Die Vorlage sollte eine Standardimplementierung für die Dienstintegration in die Plattform, die Standardkonfiguration und die generische Transformationslogik zur Modifikation von Einheiten, ermöglichen. Die Verwendung von entsprechenden Entwurfsmuster wie Vorlagenmuster ist erforderlich zur Vereinfachung der Fertigstellung und Anpassung von diesem Vorlagenprojekt. Die Implementierung von nur Entitäten lesende und neue synthetische Eigenschaften berechnende Standardtransformationen durch die Implementierung von nur eine Methode für die Berechnung der neuen Eigenschaften, muss möglich sein. Die Abfangung von jeder weiteren Verarbeitung ermöglichende Ausnahme muss durch einzelnen Transformationsdienste erfolgen.

Eine API wird vom Entitätsdienst geliefert für die Speicherung und den Abruf von Entitäten über ihre Entität ID. Ein optionaler partieller Abruf zur Vermeidung des Textkörperabrufs muss geliefert werden. Den Abruf oder die Speicherung des aggregierten Textes muss nicht nötig sein, weil dieser zur Verarbeitung vorgesehen sein sollte und die Beibehaltung der Zwischenergebnisse von Sprach- oder Ähnlichkeitsdiensten erforderlich ist. Die Wahl der Persistenz-Technologie ist erforderlich zur Optimierung der Rohspeicherung von Entitäten nach ID. Die Verwendung des Entitätsdienstes sollte nur durch den Abfragedienst erfolgen, um tatsächliche Entitäten bereitzustellen. Ein Zugriff auf die Entitäten wird erforderlich sein. Es müsste keine Abhängigkeit bestehen zwischen den Ähnlichkeitsalgorithmen und dem Entitätsdienst aus Skalierungsgründen. Für die Suche sollte der Indexzugriff nur vom Abfragedienst gebraucht werden. Falls kontextabhängige Informationen über Entitäten vom Ähnlichkeitsalgorithmus gebraucht werden, die lokale Beibehaltung dieser Information ist erforderlich. Die unveränderliche Behandlung von Entitäten über die Lade-API bis

zur nächsten Aktualisierung ist erforderlich. Speziell muss eine separate Speicherung der Informationen über Entitäten wie Lebenszyklusstatus stattfinden. Die Speicherung der Entitäten muss so erfolgen, dass eine Optimierung des häufigsten Abrufs von Entitäten ohne den Textkörper erfolgt. Eine Entität und ihres Indexes müssen konsistent gespeichert werden. Falls der Ausgleich des Ausfalls von einer der Ausdauer erfolglos ist, musste die Plattform benachrichtigt werden über den inkonsistenten Zustand der Entität.

Zur Vermeidung einer übermäßigen Belastung vom Müllsammler muss eine Zuweisung einer raumbewahrende ID an verschiedenen durch den linguistischen Dienst erzeugten Begriffen und Phrasen erfolgen. Die nicht Verwendung von Begriffe als externe Repräsentationen ist erforderlich.

Die Autonomie der Ähnlichkeitsdienste sollte so hoch wie möglich sein und die Wiederspiegelung dieser Autonomie sollte in der Beständigkeit erkennbar sein. Das Implementierungsdesign muss die Wiederspiegelung des Faktes, dass die verschiedenen Instanzen der Ähnlichkeitsdienste sowohl zur Ladezeit als auch zur Abfragezeit Konkurrent laufen, liefern. Die Ablehnung der Verarbeitung von einer Entität durch jede Ähnlichkeitsdienstversion ist möglich im Falle von ungeeignetem Ähnlichkeitsalgorithmus für diese Entität. Der Lebenszyklusstatus der Entität muss wiedergeben, dass die Verarbeitung der Entität durch die Ähnlichkeit stattgefunden hat. Die Unterscheidung dieses Status von einem Status, der den Hinweis auf einen nicht verarbeiteten Ähnlichkeitsdienst gibt, ist erforderlich. Es besteht bei der Abfrage-API die Annahme, dass die Strömung über große Listen ähnlicher Entitäten erfolgt. Die Lieferung der bestmöglichen Unterstützung durch den Ähnlichkeitsdienst ist erforderlich zur effizienten Unterstützung der Berechnung von solchen großen Listen ähnlicher Entitäten und zur Vermeidung von riesigen Ressourcenkosten. Die Berechnung von allen Ähnlichen Entitäten muss durch Algorithmen erfolgen zur Ermöglichung einer Sortierung nach Rangfolge. Die Darstellung dieser Ergebnisliste im Hinblick auf den Speicherverbrauch muss erfolgen. Alle Ähnlichkeiten sollten optional eine Zwischenspeicherung der neuesten berechneten Ähnlichkeiten, die raumlimitiert ist, fördern. Die Einstellung dieser Zwischenspeicherung sollte so erfolgen, dass es nur die Speicherung von bestimmten Ähnlichkeiten im Cache, deren Auswahl anhand der Datenquelle oder Klassifikation des Betreffs erfolgt, erfolgt. Die Entfernung von Zwischenspeicherzeilen ist erforderlich sobald das Löschen von Objekt- und Subjektentitäten erfolgt. Die Bereitstellung eines Vorlagenähnlichkeitsprojekts unter Verwendung von Spring Boot ist erforderlich zur Erleichterung der Erstellung von neuen Ähnlichkeitsdienste.

Die Beibehaltung des Lebenszyklusstatus „gelöscht“ von Entitäten muss sein zur Sicherstellung der richtigen Verwendung von Verweise auf diese Entitäts-ID. Es muss keine Abhängigkeit bestehen zwischen den Lebenszyklusdienst und irgendeine Art der Synchronisierung von Statusaktualisierungen. Die Berücksichtigung von Designs wie Ereignisbeschaffung ist umso erforderlich.

Es muss auch einen Benachrichtigungsdienst bestehen in BigData4Biz und dieser hat auch Anforderungen. Die asynchrone Bereitstellung von Statusinformationen um die Ladeverarbeitung einer Entität zu beendigen durch den Benachrichtigungsdienst ist erforderlich. Eine geeignete Benachrichtigungs technologie für Unternehmensabläufe ist auch erforderlich. Das

Abonnement des Benachrichtigungsdienstes sollte mit geringem Aufwand implementiert werden und über eine REST-API erfolgen.

3.2.5. Anforderungen an dem Ähnlichkeitsalgorithmus

Ein Ähnlichkeitsalgorithmus wird von einem spezifischen Ähnlichkeitsdienst implementiert und wird genutzt sowohl zur effektiven Berechnung von Ähnlichkeiten als auch zur Berechnung der aktuellen Ähnlichkeit für eine abgefragte Entität. Dieser hat auch besondere Anforderungen was sein Inhalt, Struktur und Funktionsweise betrifft. Bei der Berechnung von Ähnlichkeiten und Ihre Gewichtungen ist die Beachtung eines optionalen Kundenprofils erforderlich. Eine Optimierung der Speicherung von Ähnlichkeiten muss für den Speicherplatzverbrauch und für den schnellen Abruf erfolgen. Die Vorbereitung aller Ähnlichkeitsalgorithmen muss zur Erleichterung einer begrenzte Zwischenspeicherung von neuesten berechneten Ähnlichkeiten erfolgen.

3.3. High-Level Architektur

BigData4Biz wird hergestellt um gerade und zukünftig bei namhaften Kunden eingesetzt zu werden. Um diese Software langlebig und nachhaltig zu betreiben, ist die Herstellung einer Architektur nötig. [GS11] behauptet, dass die Architektur eines Systems die Strukturen des Systems, dessen Bausteine, Schnittstellen und deren Zusammenspiel beschreibt. Dies Bedeutet, dass eine Architektur die Systembausteine sowie ihre Beziehungen zueinander zum einen vorstellt und zum anderen zeigt die Gruppierung von diesen Bausteinen sowie die Bausteine gehörenden Schnittstellen. Eine Architektur hilft dabei nicht nur einen Überblick über die Struktur eines Systems zu haben durch die grobe Anzeige von nur wichtige Aspekte des Systems, sondern auch zur wartbare, flexible, verständliche und langfristige Konstruktion von diesem.

In Abbildung **??** wird die Architektur von BigData4Biz gezeigt. Hier wird keine konkrete Plattformkommunikationstechnologie vorgestellt, sondern die generellen Elemente, die wichtig sind für die Funktionsweise.

1. Die Datenquelle (1) ist ein wichtiger Teil der Architektur, der sich aber nicht in BigData4Biz befindet, sondern bei der Kundenseite. Diese entspricht eine physische Instanz, an der Daten generiert werden. Eine relationale Datenbank, ein Dateisystem, eine Ereignisquelle oder eine Webseite sind Beispiele von Datenquellen. Das Ziel der Datenquelle ist die Sammlung aller technischen Informationen, die einen Zugriff auf Informationen ermöglichen. Die Existenz von mehreren Datenquellen für eine physische Quelle zur Unterscheidung der logischen Partitionierung von Daten an der physischen Quelle, ist möglich. Die Datenquelle ist mit einer standardisierten Übertragungsschnittstelle versehen und enthält unterschiedliche Entitäten (2) und Agenten (5), die wichtig sind für die spätere Ausnutzung von Daten in BigData4Biz. Für die

Datenquelle die Ausgabe von Entitäten verschiedener Klassifizierungen und Strukturen ist möglich.

2. Die Entität (2) stellt die primär verknüpften Daten in BigData4Biz und ist ähnlich zu einer Ressource in der Terminologie von verknüpften offenen Daten und Resource Description Framework (RDF) [DIB18]. Es besteht jedoch eine starke Unterscheidung zwischen die Datenquellen der Entitäten, ihre Verwendung und die verknüpften offenen Daten. Die Datenquelle liefert die Darstellung eines beliebigen Datenelementes aus einer Datenquelle und entspricht ein relationaler Datensatz, ein Join von Beziehungsdatensätzen, eine Datei, eine Webseite, ein Text oder allgemein strukturierte Daten aller Art. BigData4Biz kann die Berechnung der Beziehung zwischen eine Subjektentität und eine Objektentität durchführen. Es sei denn eine Verknüpfung der Subjektentität über ein Prädikat mit einer Objektentität erfolgt und die Begründung von verschiedenen Arten von Ähnlichkeiten zwischen Subjekt und Objekt ist durch das Prädikat möglich. Daten werden in einer Entität (2) durch Agenten (3) umgewandelt um später in BigData4Biz über eine Lade-API (5) geladen zu werden zur Durchführung der Dokumentähnlichkeitsbestimmung. Sobald eine Entität in BigData4Biz empfangen und gespeichert wird, besitzt diese Informationen wie die Entitätsbezeichnung, die Datenquellenbezeichnung, der Agentenname, das Backlink, Metadaten und Eigenschaften.

3. Die Übertragung der Entitäten (2) von einer Datenquelle (1) zu BigData4Biz erfolgt in der Extract-Transform-Load (ETL)- Methode. Jedoch gibt es keine festgelegte Implementierung von diesem ETL-Schritt, wo es ausnahmsweise die Lade-API (5) gibt, deren erwarteten Aufgabe den Empfang von Entitäten (2) ist. Die Entitätsagenten (3) werden benutzt von der gebräuchlichsten Implementierung des ETL. Die Entitätsagenten entsprechen kleine Programme mit Zugriff auf eine Datenquelle, erhöhten Rechte, und die die Ermittlung von neuen oder geänderten Daten durchführen. Der Entitätsagent (3) führt die Umwandlung der extrahierten Daten in die Entitätsform selbst durch. Der Entitätsagent (3) befindet sich in der Sicherheitszone der Datenquelle (1) zur effektiven Erkennung der Datenänderungen und einfachere Sendung von Daten. Es bestehen schon definierte Standardagenten für allgemeine physische Datenquelle (1), die den vollständigen ETL-Schritt liefern. Die Entitätsagenten (3) sind Buchhalter aller aus ihrer Datenquelle (1) extrahierte Entitäten. Damit können Datenänderungen an Entitäten in BigData4Biz verfolgt werden und die Entität kann zum Zeitpunkt des Ladens aktualisiert werden.

4. Abfrage-API (4) dafür da eine Interaktion zwischen BigData4Biz und den Nutzer zu ermöglichen und entspricht einen REST-fähiger Dienst, der dazu dient verbundene Einblicke in die geladenen Entitäten abzufragen. Es können hier traditionelle Suche nach Entitäten mit Phrasen oder Textausschnitten durchgeführt werden im Rahmen von Abfragen. Die Umwandlung eines gefundenen Interessenbereichs in einen neuen Bereich, wo Ähnlichkeitsbeziehungen, zusätzliche Ausdrücke oder eine Auswahl signifikanter Ausdrücke des Geltungsbereichs benutzt werden, ist möglich. In einem Bereich befinden sich einige Sehenswürdigkeit darstellende dedizierte Entitäten und einen Kontext von zum Definieren von Ähnlichkeitsbeschränkungen verwendete Entitäten.

5. Die Lade-API (5) entspricht ein REST-konformer Dienst von BigData4Biz, die für den Empfang und die Ladung von Entitäten (2) zur weiteren Verarbeitung in BigData4Biz dient.

6. Die Entitätsverarbeitung (6) ist der Teil der Architektur, der sich um die Entitätsversorgung kümmert in BigData4Biz nachdem diese über die Lade-API (5) geladen wurde. Die Entität wird verarbeitet um passend zu werden für die verschiedenen Algorithmen, die anwesend in BigData4Biz sind. Bei der Entitätsverarbeitung erfolgen die Entitättransformation (5.a), die Entitätsspeicherung (5.b) und die Entitätsindizierung (5.c). Die Entitätsindizierung (5.c) zum Beispiel ist ein wichtiger Prozess, der die Assoziation eines Vokabulars aus Schlüsselwörtern und allen Dokumenten eines Textkorpus durchführt.

7. Die Linguistik (7) benutzt statistische Informationen über die Texte der Entitäten erstens zur Unterscheidung der signifikante von nicht signifikanten Teilen des Textes und zweitens sowohl zur Bestimmung der Beziehungen von Text auf der lexikalischen Ebene als auch zur Strukturierung von Texten in Gruppen ähnlicher Themen. Dieses Service stellt eine gute grobe Klassifizierung bereit. Dabei wird das Service als Basis statistische Zahlen bezüglich der Häufigkeit und des Auftritts von Begriffen in Dokumente haben und nicht das Verständnis der Bedeutung von Texten. Die Linguistik befindet sich in der Abfrage-API (4), wo ein konkreter Benutzer Suchbegriffe eingeben kann, die später zum Informationenvergleich ausgenutzt werden. Die Verwendung von einer begrenzten Teilmenge der semantischen Analyse wie Teil der Sprachmarkierung für verbesserte Ergebnisse durch signifikante Phrasenextraktion und Begriff gemeinsames Auftreten ist möglich.

8. Die Ähnlichkeitsdienste (8) berechnen die Ähnlichkeiten zwischen den Dokumenten. Es besteht bei Entitäten eine Subjekt-Prädikat-Objekt-Beziehung (SPO-Beziehung). Die Bestimmung des Prädikates erfolgt nicht durch manuelle Zuweisung oder Berechnung unter Verwendung einer Ontologie. Ähnlichkeitsdienste (8) verfügen über Ähnlichkeitsalgorithmen, die die Berechnung von durch Prädikaten ausgedruckte Ähnlichkeiten durchführen. Die Kernähnlichkeitsalgorithmen von BigData4Biz haben als Basis linguistischen Statistiken (7.a) wie TF-IDF. Die Ähnlichkeitsdienste (8) sind unabhängig von den anderen Diensten und benutzen ihre eigene Persistenz zur effektiven Berechnung von SPO-Beziehungen [Dib18].

9. Das Entitätslebenszyklusdienst (9) ist für die Überwachung von jeden schritten der Entitätsverarbeitung (6) zuständig. (Kompletter Teil noch zu ändert: die Nummerzuweisungen wurden geändert [Siehe Abbildung Word Datei])

4. Auswahl von Ansätzen zur Dokumentähnlichkeitsbestimmung in der Information-Retrieval-Middleware-Lösung

In this chapter we first provide an overview of the system and its components which provide support for accessing on-premise and off-premise Cloud data stores after migrating the data to the Cloud. In the second part of this chapter we specify the functional and non-functional requirement the system must fulfill, and provide a list of the use cases, which extend the use cases description provided in [**?**] and [**?**].

4.1. Kriterien der Auswahl

To provide transparent access support for migrated data, we present in this section an overview of the system, and its components. As we can see in Figure 4.1, we divide the system into two main parts: the *Cloud Data Migration Application*, and the Cloud data access subsystem, which we name in this diploma thesis CDASMix (Cloud Data Access Support in ServiceMix-mt). However, in this diploma thesis we do not focus on the *Cloud Data Migration Application*, but include it in the system's overview in order to explain the role of CDASMix in the context of the migration of the DL to the Cloud. We consider the different tenant's applications hosted in their environment not as part of our system, but as consumers of the services provided by it. We must specify that the system overview described in Figures 4.1 and 4.2 shows the state after the data migration, when the data is already hosted in the backend Cloud provider. However, we include the migration process explanation in this section.

In the first part of our system, the *Cloud Data Migration Application* provides support for the data migration process, from an traditional to a Cloud data store, or between Cloud data stores [**?**]. After the tenant provides the required source and target data store configuration, the application calculates possible incompatibilities between data sources, and presents them to the tenant. If they exist, the tenant must resolve the incompatibilities before migrating the data. In the end phase of the migration process data can be easily migrated to the Cloud by providing the application with the **DBMS!** (**DBMS!**) access credentials.

From the point in time where the data migration process is terminated, either the application or the tenant must choose if he directly connects to his data source in the Cloud, or if he prefers to transparently access his data in the Cloud utilizing our Cloud-enabled data bus. If the latter is chosen, either the application or the tenant must register which communication protocol is required and register access and configuration data in our registry, e.g. database type, database URL, access credentials, etc. For this purpose, we enhance the administration and

management system's (JBIMulti2) Web service **API!** (**API!**) with Cloud data access registering capabilities, as described in the following sections.

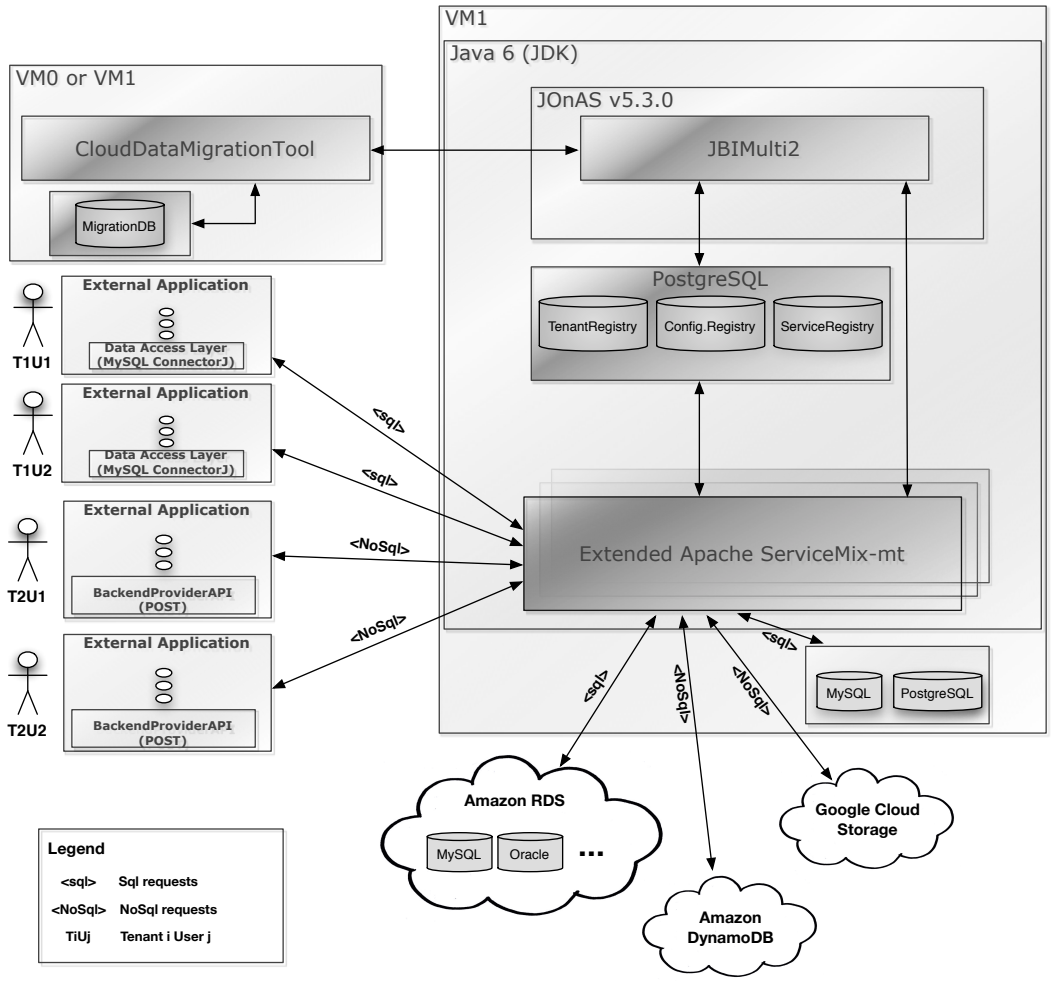

Figure 4.1.: Transparent Cloud data access system overview, including the *Cloud Data Migration Application* [**?**]. Note: represents the system in the post-migration phase

The transparent Cloud data access access support is achieved by the interaction of three main components: JBIMulti2, registries containing tenant-aware information, and an extended version of ServiceMix-mt (see Figure 4.1). JBIMulti2 deploys in ServiceMix-mt the **SA!** (**SA!**)s containing the endpoint and routing configurations selected by the tenant, which support two different communication protocols: MySQL and **HTTP!** (**HTTP!**). From this point the DAL of the tenant's application can retrieve and modify data in his data container in the Cloud through the **ESB!** (**ESB!**) connecting to a single logical endpoint which connects to multiple physical backend data stores. In our approach we provide also the possibility, either to configure a connection to the traditional database, e.g. when a hybrid model is pursued, or to utilize a **DBMS!** provided in our system, which is described in the following subsection.

4.2. Funktionsweise des Ansatzes

In Figure 4.2 we specify the main components which build the subsystem mentioned in Section **??**. We highlight the components which require an extension, and the new components which are implemented and included in the system.

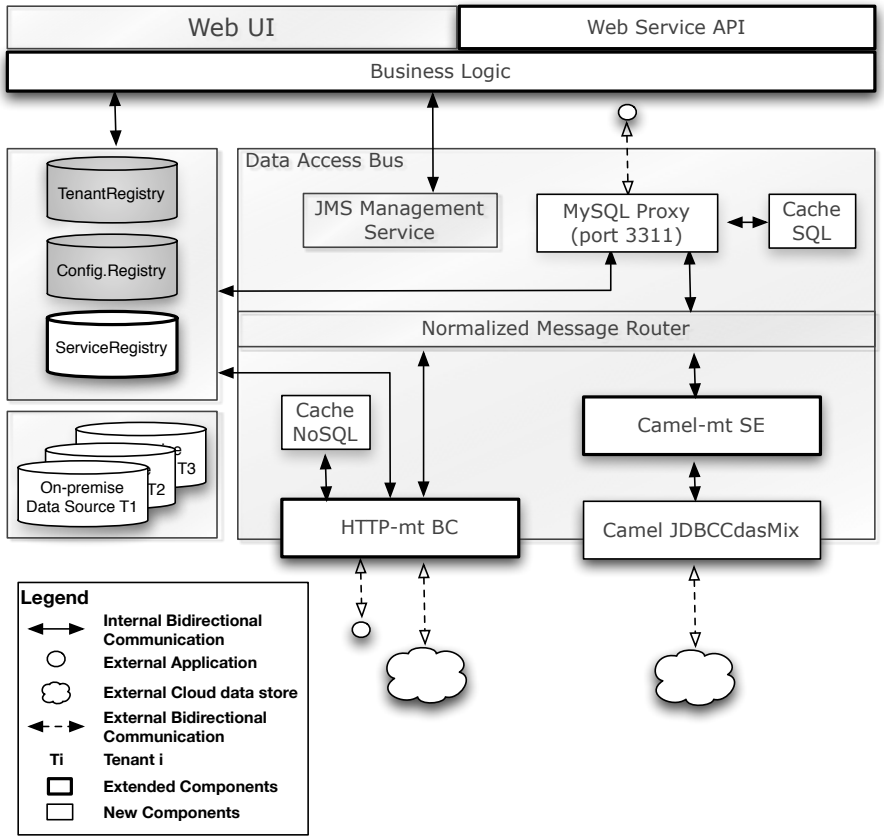

Figure 4.2.: Transparent Cloud data access components overview.

As described in Section **??**, we extend the JBIMulti2 Web service **API!** and its associated business logic. The operations we include perform access and modifications to one particular registry: the Service Registry. This registry persists information about services, its policies, and **SA!**s deployed by one tenant. The last kind of information is the one we mainly focus on, due to the information which is contained in it: the tenant-aware endpoint configuration in ServiceMix-mt. Therefore, we extend this registry to persist the configuration about the data stores. We make a differentiation between data stores and name it source and target data sources, to be able to relate the one that the tenant physically accesses and the one which the tenant logically accesses, which is the one that the system physically accesses. To support transparent access to two different database types, we divide our architecture and implementation into the the communication protocols they support: MySQL for MySQL databases, and HTTP for **NoSQL!** (**NoSQL!**) databases (see Figure 4.2).

For the first one, the single access physical endpoint is provided through a **TCP!** (**TCP!**) port,

which then forwards the message to the appropriate tenant's endpoint in the multi-tenant Servicemix Camel component, and this to the component which physically connects to the backend **SQL!** (**SQL!**) **DBMS!**. For the second one, we extend the Servicemix-http-mt in order to physically connect to the backend **NoSQL!** data stores.

The possibility of migrating a database to the VM instance where the **ESB!** runs is also supported. However, we do not provide a multi-tenant **DBMS!** where a database or table is shared between multiple tenants, since it is not a requirement in this diploma thesis. The ensured isolation in this case is the one provided by a **DBMS!** between different databases. Furthermore, backup, restoration, administration, and horizontal scalability services are not supported. We provide in the VM instance where CDASMix runs a MySQL database system. The number and types of databases systems supported in the VM instance relies on the administrator, and the PostgreSQL database system where the system registries are stored must be independent from the PostgreSQL instance which hosts the migrated tenant's databases.

As represented in Figure 4.2, we enhance ServiceMix-mt with cashing support, in particular for the two different types of databases we support. The cashing mechanism supports storage of access control data, as well as retrieved data from the backend data store, e.g. a bucket, or a set of rows. The reasons for using a divided cashing system instead of a single one is explained in Chapter **??**.

5. Implementierung einer Teilmenge ausgewählten Ansätzen

In this chapter we present the architectural solution taken into account to build the system which fulfills the requirements specified in Chapter **??**. Due to the required communication support for **SQL!** and **NoSQL!** databases, we separate the architectural approaches and provide them separately. JBIMulti2 and ServiceMix-mt are the subsystems we must reengineer in order to aggregate transparent and dynamic routing functionalities. Therefore, we also provide in this chapter the needed extensions in the components conforming the system, e.g. service registry in JBIMulti2, and **NMF!** (**NMF!**) in ServiceMix-mt.

6. Evaluierung der Skalierbarkeit von den realisierten Ansätzen basierend auf ausgewählten Probendaten

In this chapter we describe the challenges and problems during the implementation phase to fulfill the requirements specified in Chapter **??** and the design presented in Chapter **??** of the system. Furthermore, we discuss the incompatibilities found with components we must extend. We divide, as in the previous chapters, the implementation phase into the **SQL!** and **NoSQL!** databases support, and provide a separate section for the extensions made to JBIMulti2 and the Cache.

6.1. Fähigkeiten und Grenzen ausgewählter Ansätze

7. zusammenfassung und Ausblick

In this chapter we provide the validation, and evaluation of the system. We must ensure that the requirements specified in Chapter **??** are fulfilled in the design and implementation phases. In Section **??** we describe the steps which should be followed to initialize the system, and the testing scenarios. After the initialization we execute the test cases in Section **??**, and monitor the incoming requests to ServiceMix-mt, and the outgoing requests to the backend Cloud data store. Due to the extensions implemented on the **ESB!**, we evaluate in Section **??** its behavior, and the impact that our modifications have on the original ServiceMix-mt.

Appendix A.

Components

A.1. CDASMix MySQL Proxy

The MySQL proxy **OSGi!** (**OSGi!**) bundle is implemented on the Continuent Tungsten Connector [**?**], which is a Java MySQL proxy which directly connects with the backend MySQL database system. We extend and adapt this proxy in order to integrate it with ServiceMix, aggregate transparency, multi-tenant awareness, cashing, and dynamic connection with the backend Cloud data sources.

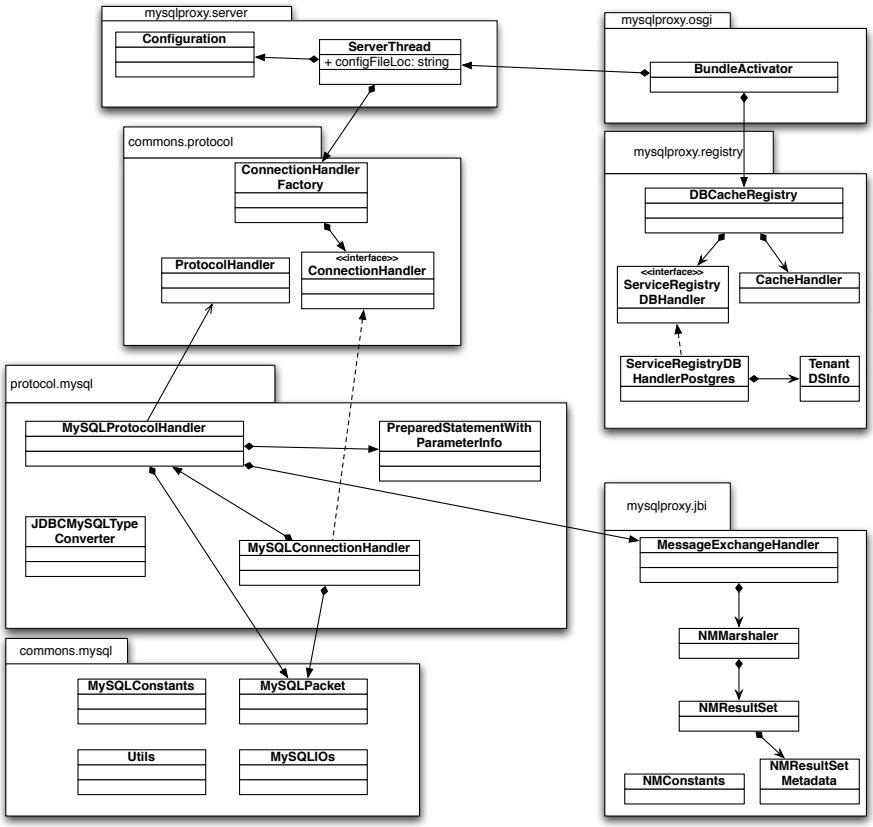

Figure A.1.: OSGi bundle providing MySQL support in ServiceMix-mt

A.2. CDASMix Camel JDBC

The *cdasmixjdbc* component is a custom component which is built and deployed as an **OSGi!** bundle in ServiceMix-mt. It provides support for connections with backend **SQL!** Cloud data stores, and message marshaling and demarshaling.

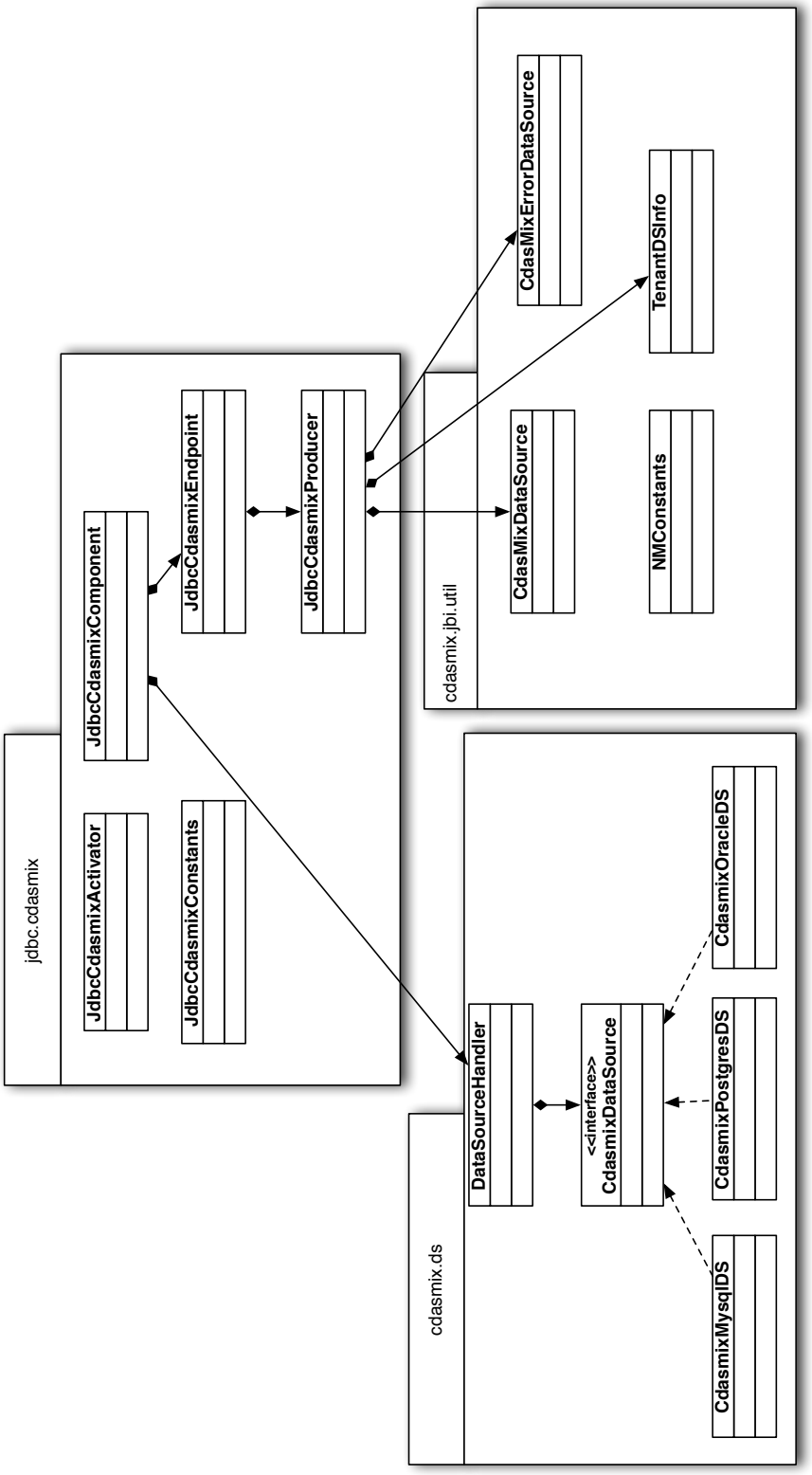

Figure A.2.: OSGi bundle and Camel component providing JDBC support in ServiceMix-mt

Appendix B.

Messages

In this chapter we provide an overview of the requests which are sent to, and received from the extended ServiceMix-mt. For MySQL requests we provide the **TCP!** packets which are transferred between the application, ServiceMix-mt, and the backend MySQL database system. For NoSQL requests we present messages samples which are in **JSON!** (**JSON!**) format, but its content varies among the different backend Cloud data store providers.

B.1. Normalized Message Format Content Description

In this section we provide an overview of the data structures which are sent in the **NMF!**. The sections of the **NMF!** where the data and meta-data are sent are the *properties*, and the *attachment*. In the Listing B.1 we detail the contents sent in each of the sections, and the data structures in which the data and meta-data are stored.

```
1  ########## Normalized Message Adaptation for CDASMix ###########
2
3  Properties for the backend connections are stored in this format. At this time, only joins
       which involve tables in the same backend db are supported:
4
5  ////////////////////////////////////////////////////////////////////
6
7    - MySQL || NoSql -> NMF (Request Message)
8      // Main properties
9      - target_data_sources : number of target data sources. This number will set the length
           of the properties vector and will set the length of the vector queries
10     - tenantId : string (UUID)
11     - userId : string (UUID)
12     if (mysql)
13       - mysqlServerPropsQueries:vector<string>      // server configuration queries that the
             proxy has received from the jdbc driver, e.g. SET names, SET character_set_*
14
15     // Backend datasource dependent properties : stored as Properties in vector<properties>.
16     - Tenant configuration data
17       - source & target dataSource name : string
18       - source & target dataSource type : family-mainInfoStructure-secondaryInfoStructure-
             version
19       - source & target dataSource protocol : mysql | http.[xml|rest]
20       - target datasource endpoint url: string
21       - target datasource user and password : string, string
```

```
22        - source & target endpoint type: endpoint.type.jdbc | endpoint.type.http | endpoint.
              type.jms
23        - targetJBIEndpointURI: QName
24        - source and target information structure information:
25                if source is sql
26                  - src_main_info_structure_name : string
27                  - target_main_info_structure_name : string
28                if source is nosql
29                  - src_main_info_structure_name : string
30                  - target_main_info_structure_name : string
31                  - src_secondary_info_structure_name : string
32                  - target_secondary_info_structure_name : string
33
34        - if the target protocol == mysql
35          - target native driver name : native driver name, e.g. "com.mysql.jdbc.driver"
36          - bean DS name (if exists)
37          - escapeProcessing : true | false               //statement property of mysql
38          - fetchSize : int (statement property of mysql)
39          - returnGeneratedKeys : true | false            //statement property of mysql
40
41      // Attachment
42      if (SQL)
43        - Query/ies go/es in the NMF body as a vector<String>, with query data (for insert
              queries).
44      else
45        - NoSQL JSON payload
46
47
48  ////////////////////////////////////////////////////////////////////////
49
50    - NMF -> MySQL || NoSql (Response Message)
51      // Main properties
52      - target_data_sources : number of target data sources. This number will set the length
            of the properties vector and the length of the result
53      - tenantId : string (UUID)
54      - userId : string (UUID)
55      - target_op_status = ok | error
56
57      - if source protocol == mysql
58
59        - updateCount = int           // the current result as an update count
60        - resultSetMetadataNumber: int      // number of result set in the vector of response
              (index in the result set metadata vector)
61
62      // Backend datasource dependent properties : stored as Properties in the vector<
            properties>.
63      - source & target dataSource name : string
64      - source & target dataSource type : family-mainInfoStructure-secondaryInfoStructure-
            version
65      - source & target dataSource protocol : mysql | http.[xml|rest]
66      - source & target endpoint type: endpoint.type.jdbc | endpoint.type.http | endpoint.type
            .jms
67      - source and target information structure information:
68              if source is sql
69                - src_main_info_structure_name : string
```

```
70              - target_main_info_structure_name : string
71            if source is nosql
72              - src_main_info_structure_name : string
73              - target_main_info_structure_name : string
74              - src_secondary_info_structure_name : string
75              - target_secondary_info_structure_name : string
76
77    // Attachment
78    - if target_op_status == error
79      - target_op_error : vector<Hashmap<String,String>>            // map of error code and
            error message per backend db
80      else
81
82
83      - if source protocol == mysql
84
85        - vector<arraylist[HashMap<columnName,value>]]>           // result sets are
            inserted as arraylist in the body
86
87        - ResultSetMetadata: Vector<HashMap<String,Object>>
88          - columncount : int                        // number of columns in this ResultSet
              object.
89          - columntype : arraylist<int>                 // the designated column's SQL type.
90          - isnullable : arraylist<int>                 // if the column can contain null
              values
91          - isAutoIncrement : arraylist<int>               // if the column values
              increment when the rows do
92          - tableName : arraylist<string>                // name of the table where a
              column is contained
93          - stringcolumnlabel : arraylist<string>             // label which sql proposes
              for printing reasons
94          - stringcolumnname : arraylist<string>              // column name
95          - ColumnDisplaySize : arraylist<int>              // Indicates the designated
              column's normal maximum width in characters.
96          - Scale : arraylist<int>                    // column's number of digits to right
              of the decimal point
97          - RowCount : int                         // number of rows
98          - ColumnMapper : Hashmap<int,String> (column index, name)   // mapping of the
              column index with the column name
99          - ColumnSigned : arraylist<int>                 // if the values of the columns
              are signed or unsigned
100       else
101       - versionControl
102       - nosql metadata
103       - nosql JSON payload
104
105 ////////////////////////////////////////////////////////////////////
106
107   Note: response to the user -> distributed atomic transaction mechanism used. If one of
          the backend datasources reports an error as a response, the final response to the
          tenant is an error response:
108
109     - Error response: string of the error + the datasource/s which gave the error.
110
111     - OK response: the full result set, this means, the vector<arraylist[HashMap<
```

```
        columnName,value>]]> created in the data transformation SE or jdbccdasmix
        component.
112
113
114 ////////////////////////////////////////////////////////////////////
```

Listing B.1: Detail of the content and data structures used to send the requests' data and meta-data.

B.2. MySQL TCP Stream

In this section we provide two **TCP!** streams which are captured with the *ngrep* program for UNIX [**?**]. The first stream captures the **TCP!** packets on port 3311, where the MySQL component in ServiceMix-mt listens for incoming connections (see Listing B.2). The second stream captures the **TCP!** packets on port 3306, where the a locally deployed MySQL server listens for incoming connections (see Listing B.3).

```
interface: eth3 (109.231.70.232/255.255.255.248)
filter: (ip or ip6) and ( port 3311 )

T 109.231.70.234:3311 -> 129.69.214.249:52190 [AP]
  45 00 00 00 0a 35 2e 31    2e 31 2d 53 65 72 76 69    E....5.1.1-Servi
  63 65 4d 69 78 2d 34 2e    33 2e 30 00 2d 02 00 00    ceMix-4.3.0.-...
  53 28 72 4d 51 30 6c 61    00 00 0d a2 02 00 00 00    S(rMQ0la........
  00 00 00 00 00 00 00 00    00 00 00 32 2f 54 6f 44    ...........2/ToD
  4d 4a 39 2a 70 69 49 00    00                         MJ9*piI..

  ...

T 129.69.214.249:52190 -> 109.231.70.234:3311 [AP]
  0f 00 00 00 03 53 45 54    20 4e 41 4d 45 53 20 75    .....SET NAMES u
  74 66 38                                              tf8

T 109.231.70.234:3311 -> 129.69.214.249:52190 [AP]
  07 00 00 01 00 00 00 02    00 00 00                   ...........

  ...

T 129.69.214.249:52190 -> 109.231.70.234:3311 [AP]
  1d 00 00 00 03 73 65 6c    65 63 74 20 2a 20 66 72    .....select * fr
  6f 6d 20 6d 61 69 6e 49    6e 66 6f 54 65 73 74 31    om mainInfoTest1
  3b                                                    ;

T 109.231.70.234:3311 -> 129.69.214.249:52190 [AP]
  01 00 00 01 02 46 00 00    02 03 64 65 66 12 69 6e    .....F....def.in
  66 6f 72 6d 61 74 69 6f    6e 5f 73 63 68 65 6d 61    formation_schema
  0d 6d 61 69 6e 49 6e 66    6f 54 65 73 74 31 0d 6d    .mainInfoTest1.m
  61 69 6e 49 6e 66 6f 54    65 73 74 31 02 49 44 02    ainInfoTest1.ID.
  49 44 0c 21 00 0b 00 00    00 03 01 02 00 00 00 4a    ID.!...........J
  00 00 03 03 64 65 66 12    69 6e 66 6f 72 6d 61 74    ....def.informat
  69 6f 6e 5f 73 63 68 65    6d 61 0d 6d 61 69 6e 49    ion_schema.mainI
  6e 66 6f 54 65 73 74 31    0d 6d 61 69 6e 49 6e 66    nfoTest1.mainInf
  6f 54 65 73 74 31 04 6e    61 6d 65 04 6e 61 6d 65    oTest1.name.name
  0c 21 00 ff ff 00 00 fc    01 00 00 00 00 05 00 00    .!..............
  04 fe 00 00 02 00 13 00    00 05 01 33 10 54 68 69    ...........3.Thi
  73 69 73 41 4e 65 77 4e    61 6d 65 33 33 08 00 00    sisANewName33...

  ...

T 129.69.214.249:52190 -> 109.231.70.234:3311 [AP]
  01 00 00 00 01                                        .....

T 109.231.70.234:3311 -> 129.69.214.249:52190 [AP]
  07 00 00 01 00 00 00 02    00 00 00                   ...........

T 129.69.214.249:52190 -> 109.231.70.234:3311 [R]
  00 00 00 00 00 00                                     ......

T 129.69.214.249:52190 -> 109.231.70.234:3311 [R]
  00 00 00 00 00 00                                     ......
```

Listing B.2: TCP Stream for a MySQL communication captured on port 3311 with the program *ngrep* [**?**].

```
interface: lo (127.0.0.0/255.0.0.0)
filter: (ip or ip6) and ( port 3306 )

  ...

T 127.0.0.1:46409 -> 127.0.0.1:3306 [AP]
  0f 00 00 00 03 53 45 54    20 4e 41 4d 45 53 20 75    .....SET NAMES u
  74 66 38                                              tf8

T 127.0.0.1:3306 -> 127.0.0.1:46409 [AP]
  07 00 00 01 00 00 00 02    00 00 00                   ...........

  ...

T 127.0.0.1:46409 -> 127.0.0.1:3306 [AP]
  50 00 00 00 03 75 70 64    61 74 65 20 6d 61 69 6e    P....update main
  49 6e 66 6f 54 65 73 74    31 20 73 65 74 20 6e 61    InfoTest1 set na
  6d 65 3d 27 54 68 69 73    69 73 41 4e 65 77 4e 61    me='ThisisANewNa
  6d 65 33 33 27 20 77 68    65 72 65 20 6e 61 6d 65    me33' where name
  3d 27 54 68 69 73 69 73    41 4e 65 77 4e 61 6d 65    ='ThisisANewName
  32 32 27 3b                                           22';

T 127.0.0.1:3306 -> 127.0.0.1:46410 [AP]
  45 00 00 00 0a 35 2e 31    2e 36 37 2d 30 75 62 75    E....5.1.67-0ubu
  6e 74 75 30 2e 31 30 2e    30 34 2e 31 00 c0 02 00    ntu0.10.04.1....
  00 5f 67 6b 63 2f 5e 6a    7b 00 ff f7 08 02 00 00    ._gkc/^j{.......
  00 00 00 00 00 00 00 00    00 00 00 00 4e 53 35 4e    ............NS5N
  68 44 57 6a 2f 48 5e 21    00                         hDWj/H^!.

T 127.0.0.1:46410 -> 127.0.0.1:3306 [AP]
  4a 00 00 01 8f a2 02 00    ff ff ff 00 21 00 00 00    J...........!...
  00 00 00 00 00 00 00 00    00 00 00 00 00 00 00 00    ................
  00 00 00 00 72 6f 6f 74    00 14 f3 54 d0 fa f3 56    ....root...T...V
  e9 c0 43 2c 4c 78 18 88    ac de 4c 5e aa d1 64 61    ..C,Lx....L^..da
  74 61 53 6f 75 72 63 65    54 65 73 74 31 00          taSourceTest1.

T 127.0.0.1:3306 -> 127.0.0.1:46410 [AP]
  07 00 00 02 00 00 00 02    00 00 00                   ...........

  ...

T 127.0.0.1:46410 -> 127.0.0.1:3306 [AP]
  1d 00 00 00 03 73 65 6c    65 63 74 20 2a 20 66 72    .....select * fr
  6f 6d 20 6d 61 69 6e 49    6e 66 6f 54 65 73 74 31    om mainInfoTest1
  3b                                                    ;
```

Listing B.3: TCP Stream for a MySQL communication captured on port 3306 with the program *ngrep* [**?**].

bibliographyliteratur/literatur

Bibliography

[CDM08] H. S. Christopher D. Manning, Prabhakar Raghavan. *Introduction to Information Retrieval*. Cambridge University Press, New York, 2008.

[COM18] So schuetzen Sie sich vor der E-Mail Spam. *Computer Bild*, February 2018.

[Dib18] M. Dibuco. Connected Insight, 2018.

[ELA18] Elasticsearch reference, 2018.

[GS11] P. H. G. Starke. *Software-Architektur kompakt*. Spektrum Akademischer Verlag, Heidelberg, 2011.

[Mel15] M. Melucci. *Introduction to Information Retrieval and Quantum Mechanics*. Springer-Verlag GmbH, Berlin Heidelberg, 2015.

[Mer11] A. Merv. It's going mainstream, and it's your next opportunity. *Teradata Magazine*, 2011.

[Pla17] P. D. H. Plattner. Big Data. 2017.

[RBY99] B. R.-N. R. Baeza-Yates. *Modern Information Retrieval*. ACM Press, New York, 1999.

[SC13] M. B. E. D. V. P. F. S. Q. Stefano Ceri, Alessandro Bozzon. *Web Information Retrieval*. Springer-Verlag, Berlin Heidelberg, 2013.

[uAM16] D. F. und Andreas Meier. *Big Data*. Springer Fachmedien, hmd edition, 2016.

All links were last followed on March 21, 2013.

Acknowledgement

I am heartily thankful to my supervisor Steve Strauch from the University of Stuttgart for his encouragement, guidance and support in all the phases of this diploma thesis. I am also grateful to Dr. Vasilios Andrikopoulos for his advices and useful tips. Special thanks to my family, friends and girlfriend for their moral support.

Santiago Gómez Sáez

Declaration

All the work contained within this thesis, except where otherwise acknowledged, was solely the effort of the author. At no stage was any collaboration entered into with any other party.

Stuttgart, 22nd March 2013 _____

(Santiago Gómez Sáez)