Hochschule für Technik, Wirtschaft und Kultur Leipzig

Fakultät Informatik, Mathematik und Naturwissenschaften

Masterstudiengang Informatik

Masterarbeit

zur Erlangung der akademischen Grades

Master of Science (M.Sc.)

Untersuchung und Optimierung verteilter Geografischer Informationssysteme zur Verarbeitung Agrartechnischer Kennzahlen

Eingereicht von: Kurt Junghanns

Matrikelnummer: 59886

Leipzig 17. Februar 2015

Erstprüfer: Prof. Dr. rer. nat. Thomas Riechert

Zweitprüfer: M. Sc. Volkmar Herbst

Abstrakt

Danksagung

Vorwort

Glossar

Bonitur landwirtschaftliche Beurteilung des Ackers und der Pflanzen zum Zwecke der Planung des Einsatzes von Dünger, Pestiziden, Fungiziden und Herbiziden.

Cascading Das Java Framework Cascading steht unter der Apache License dient der Erstellung komplexer Datenverarbeitungsabläufe. Dafür wird MapReduce indirekt in vereinfachter Form zugänglich gemacht.

GeoServer freier OGC konformer Mapserver der Open Source Geospatial Foundation, geschrieben in Java.

Pig Als Apache Projekt dient Pig zur Abstraktion von Java MapReduce Jobs in der Sprache Pig Latin. Ziel ist eine Vereinfachung von MapReduce mit der gleichzeitigen Einbindung externer Funktionen.

R Die plattformunabhängige Programmiersprache R steht unter der GNU General Public License ist wird für statistisches Rechnen und dessen grafische Aufbereitung verwendet.

Scala Scala ist eine objektorientierte funktionale Programmiersprache mit einem statischen Typsystem und ist auf der JVM und LLVM lauffähig.

Spark Apache Spark steht unter der Apache License 2.0 und ist ein Framework zur Datenverarbeitung in Clustersystemen. Es tritt mit Hadoop in Konkurrenz und arbeitet mit HDFS, Apache Cassandra, OpenStack Swift, Amazon S3 und Accumulo zusammen.

Storm Apache Storm ist ein Framework speziell für Stapelverarbeitung von Datenströmen durch verteilte Prozesse. Es steht unter der Apache License 2.0

UMN MapServer Mapserver des OGC unter MIT Lizenz, Erstentwicklung durch Universität von Minnesota, welcher als CGI Modul und für verschiedene Sprachen bereitsteht.

Abkürzungsverzeichnis

ACID Atomicity, Consistency, Isolation und Durability

BASE Basically Available, Soft state, Eventual consistency

BSON Binary JSON

CAP Consistency, Availability and Partition Tolerance

DBMS Datenbankmanagementsystem

DBS Datenbanksystem

GIS Geoinformationssystem

GPL General Public License

HDFS Hadoop File System

JSON JavaScript Object Notation

JTS Java Topology Suite

LGPL Lesser General Public License

MPP Massively Parallel Processing

MTTF Mean Time to Failure

MVCC Multi Version Currency Control

ODBMS objektorientiertes Datenbankmanagementsystem

OGC Open Geospatial Consortium

TPC Transaction Processing Performance Council

Abbildungsverzeichnis

Tabellenverzeichnis

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Die Agri Con GmbH verwaltet als Akteur im Bereich „Precision Farming“ täglich mehrere Millionen geografische Punktdaten. Diese Daten werden von aktiven Landwirtschaftsmaschinen und durch die Verarbeitung durch firmeninterne und firmenexterne Mitarbeiter sowie Systeme erzeugt. Weiterhin fallen dadurch indirekt Vektor- und Rasterdaten an, welche gespeichert und anschließend verarbeitet werden müssen. Aus den Quelldaten werden Vektordaten für beispielsweise Verteilung der Grunddüngung erzeugt. Rasterdaten werden für „N-Düngung“ verwendet, was unter anderem die Biomasse, die Nährstoffaufnahme und die Nährstoffverteilung beinhaltet. Diese Menge an Daten ist essentiell für den Betrieb, weshalb diese strukturiert gespeichert und kostengünstig verarbeitet werden müssen. Nicht nur Agri Con steht vor dieser Notwendigkeit, sondern der Großteil der Unternehmen, die sich mit komplexen Geodaten beschäftigen, wie Monsanto, Google, Facebook, ESRI, OpenGEO, etc.

1.2 Zielsetzung

Die aktuellen Werkzeuge kommen an ihre Grenzen wenn große Datenmengen zur Laufzeit bearbeitet werden müssen. Es ist zu untersuchen welche Vorteile andere Datenhaltungssysteme bieten bzw. welche alternativen Herangehensweisen wie NoSQL, die Verwendung von caching und die verteilte Datenhaltung verwendet werden können. Dafür sind existierende Geoinformationssysteme (GISs) zu untersuchen und deren Eignung für den in Kapitel 4 beschriebenen Anwendungsfall festzustellen. Die Schwerpunkte

der Untersuchung sind die Möglichkeiten und die Leistungsfähigkeit der räumlichen Datenverarbeitung und nicht die Formen der Datendarstellung. Dabei werden NoSQL und Open-Source Systeme bevorzugt behandelt. Aus geeigneten Frameworks wird eines ausgewählt. Dieses wird speziell untersucht und eine prototypische Installation[1] erstellt. Schlussendlich soll eine Entscheidungsgrundlage anhand von Qualitätsmerkmalen für die teilweise Ersetzung des Ist-Standes gegeben werden.

1.3 Aufbau der Arbeit

Zu Beginn werden theoretischen Grundlagen zu Datenbanken, geographischer Datenverarbeitung, NoSQL und Leistungstests festgehalten. Anschließend definiert Kapitel 3 das Ausgangsszenario, für welches die Systeme analysiert und getestet werden sollen. Darauf folgend bewertet eine Nutzwertanalyse ausgewählte Frameworks nach den Anforderungen des Anwendungsfalles. Das vorletzte Kapitel stellt das ausgewählte Framework unter den Punkten Aufbau, Installation, Datenimport, Verarbeitung, Schnittstelle und Leistungstest dar. Die Thesis endet mit einer Zusammenfassung, einer Empfehlung bzw. Wertung der Ergebnisse und einem Ausblick auf die zukünftige Handhabung der räumlichen Daten bei Agri Con.

[1] Dabei kann eine Installation aus mehreren Frameworks bestehen und eigens implementierte Funktionalitäten enthalten

2 Grundlagen

Framework

Ralph E. Johnson in [AD14] definiert ein Framework wie folgt:

> *Ein Framework ist eine semi-vollständige Applikation. Es stellt für Applikationen eine wiederverwendbare, gemeinsame Struktur zur Verfügung. Die Entwickler bauen das Framework in ihre eigene Applikation ein, und erweitern es derart, dass es ihren spezifischen Anforderungen entspricht. Frameworks unterscheiden sich von Toolkits dahingehend, dass sie eine kohärente Struktur zur Verfügung stellen, anstatt einer einfachen Menge von Hilfsklassen.*

In dieser Arbeit dienen Frameworks oder auch Ordnungsrahmen zur Lösung spezieller Aufgaben und sind somit domänenspezifische Frameworks. Das heißt, dass notwendige Funktionen und Strukturen zur Lösung von speziellen Aufgaben bereits vorhanden sind, die konkreten Lösungen müssen jedoch mit Hilfe des Frameworks erstellt werden.

2.1 Datenbank

2.1.1 Begriffsdefinitionen

ACID

Die bekanntesten Vertreter von relationalen Datenbanksystemen wie Oracle, MySQL und PostgreSQL arbeiten transaktional nach Atomicity, Consistency, Isolation und Durability (ACID). ACID kann nach [Kud07, S.262] wie folgt definiert werden:

Atomarität

Transaktionen sind atomar, wodurch ein Abbruch einer Transaktion deren enthaltenen Operationen rückgängig macht.

Konsistenz

Das Ende oder der Abbruch einer Transaktion geht immer mit Nachbedingung aller Intergritätsbedingungen einher.

Isolation

Transaktionen verschiedener Benutzer beeinflussen sich nicht gegenseitig.

Dauerhaftigkeit

Jede Änderung einer Transaktion ist nach Ende dieser auf die Festplatte geschrieben und nicht mehr im Puffer vorhanden.

Die Definition dieses anerkannten Begriffes ist für Kapitel 2.3 notwendig.

MVCC

In grundlegenden relationalen Systemen werden Transaktionen verzögert oder sogar gesperrt, um Konsistenz und Isolation zu gewährleisten. Multi Version Currency Control (MVCC) erhöht die Effizienz des blockierenden Verhaltens. Dabei werden von jedem Objekt mehrere Versionen verwaltet. Neue Versionen entstehen durch Änderungen einer anderen. Eine Transaktion verwendet die zu Transaktionsbeginn aktuelle Version. Dadurch werden die allgemeinen Sperrverfahren (siehe [Kud07, S.266 ff.]) verbessert, indem lesende Transaktionen sich nicht gegenseitig blockieren und schreibende- gegen lesende Transaktionen nicht mehr synchronisiert werden müssen. (vgl. [Kud07, S.270])

BASE

Basically Available, Soft state, Eventual consistency (BASE) ist ein optimistischer und sperrenfreier Ansatz mit fließender Konsistenz. [Edl11]

CAP

Consistency, Availability and Partition Tolerance (CAP)
TODO

2.1.2 Indexstrukturen

Indexstrukturen oder allgemein Zugriffsstrukturen dienen dem effizienten Zugriff auf Dateneinträge. Ein Index ist nach [Kud07, S.284] wie folgt definiert:

> *Ein Index ist ein Verzeichnis von Dateneinträgen der Form (k, k*), das den effizienten Zugriff auf allen Einträgen mit einem Suchschlüsselwert k erlaubt. Dabei bezeichnet k den Wert eines Suchschlüssels (auch Zugriffsattribut) und k* den Verweis auf den Datensatz in der Datei, der k als Wert des Suchschlüssels enthält.*

Zugriffsstrukturen haben je nach Art und Umfang der Daten sowie entsprechend den Anforderungen an das Datenbanksystem (DBS) unterschiedliche Strukturen. In der einfachsten Struktur unterscheidet man nach Indexen die direkt die Daten beinhalten, auf die Daten zeigen oder eine Menge von Adressen beinhalten. (siehe [Kud07, S.284])

Im folgenden werden spezielle Indexstrukturen vorgestellt, da dieses Wissen zur Bewertung von DBS herangezogen werden müssen.

B-Baum

Nach [Kud07, S.288] ist ein B-Baum ein dynamisch balancierter Indexbaum, bei dem jeder Indexeintrag auf eine Seite der Hauptdatei zeigt. Der Baum besitzt die Höhe h und die Ordnung m sowie die folgenden Eigenschaften:

> *1. Jeder Weg von der Wurzel zum Blatt hat die Länge h (balanciert)*
> *2. Jeder Knoten enthält mindestens m Elemente (außer der Wurzel) und höchstens 2m Elemente (mindestens halbvolle Belegung)*
> *3. Jeder Knoten ist entweder eine Blattseite oder hat höchstens 2m + 1 Kinder (maximale Belegung)*[1]

[1] ebenda

Diese Struktur garantiert eine Belegung von 50%. Weiterhin beschreibt h die Anzahl der Seitenzugriffe als relevantes Maß für die Zugriffskosten und Datensätze n bedingen den Zugriff in maximal logm(n) Seitenzugriffen. (vgl. [Kud07, S.288])

Eine Spezialisierung stellt der B+-Baum dar. Hierbei befinden sich die Dateneinträge ausschließlich in den Blattknoten. Die Blattknoten sind unidirektional verkettet.

LSM-Baum

Log structured merge tree
TODO

R-Baum

R-Bäume sind balancierte Bäume und „organisieren k-dimensionale Rechtecke mithilfe überlappender Blockregionen“ [Kud07, S. 523] Diese Struktur wird folglich zur räumlichen Datenhaltung eingesetzt, da die Indexierung anhand räumlicher Informationen der Daten erfolgt. Ein Verzeichnisknoten besteht aus einem Tupel (ref, mur). ref steht für den Verweis auf den direkten Nachfahren und mur für das minimal umgebende Rechteck der Kindknoten. Datenknoten enthalten dagegen nur mur als eigentliches Geoobjekt. (vgl. [Kud07, S.523 ff.])

Geohash

Bei Bedarf

2.1.3 Mehrrechner-Datenbanksystem

Nach [Kud07, S.394] wird bei einem Mehrrechner-Datenbanksystem (MDBS) die Datenbankverwaltungsfunktionen auf mehreren Prozessoren bzw. Rechnern ausgeführt. Kudraß ergänzt dies durch folgende Unterscheidungen:

shared everything Datenbankmanagementsystem (DBMS) befindet sich auf eng gekoppelter Multiprozessor-Umgebung.

shared nothing Die Verarbeitung erfolgt durch mehrere Rechner mit jeweils einem DBMS, dabei ist der Externspeicher unter den beteiligten Rechnern partitioniert.

shared disk Hierbei handelt es sich um mehrere lokal angeordnete, lose oder nah gekoppelte Rechner mit je einem DBMS und einer gemeinsamen Speicherzuordnung. Lokal verteilte Systeme werden als parallele Datenbanksysteme bezeichnet.

2.1.4 Verteiltes Datenbanksystem

Verteilte Datenbanksysteme (VDBS) sind geografisch verteilte Shared-Nothing Systeme mit homogenen lokalen DBMS, die gemeinsam ein globales konzeptionelles DB-Schema unterstützen. Förderierte Datenbanksysteme (FDBS) sind ebenfalls geografisch verteilte Sgated nothing systeme, wobei die beteiligten lokalen DBMS eine höhere Autonomie aufweisen, d.h. dass jeweils eine eigene lokale Datenbank mit lokalem DB-schema vorliegt.[2]

2.1.5 Replikationsverfahren

Synchron

Bei Bedarf

Asynchron

Bei Bedarf

[2] [Kud07, S.398]

Kaskadiert

Bei Bedarf

2.2 geografische Datenverarbeitung

2.2.1 Bezugssysteme

> *Räumliche Bezugssysteme (spatial reference systems) erlauben die Interpretation der gespeicherten Koordinaten als Beschreibung von Lage- und Ausdehnungsinformationen in einem (realen) Datenraum. Ein räumliches Bezugssystem besteht aus einem Koordinatensystem (coordinate system), einem Geltungsbereich und Angaben, die es erlauben, Daten aus unterschiedlichen Koordinatensystemen auf ein globales System abzubilden.*[3]

Kudraß allgemeine Definition wird durch [Lan13, S.141 ff.] mit folgendem ergänzt: Man unterscheidet Koordinatensysteme nach kartesisch, homogen, Kugeltransformation und Ellipsoidentransformation, wobei den kartesischen einer hoher Stellenwert zugegordnet wird.

Allen Bezugssystemen wird zur Identifikation ein weltweit eindeutiger Code zugeordnet. Dieser ist ein von der European Petroleum Survey Group Geodesy vergebener so genannter EPSG Code.

Das auf einem Ellipsoiden basierende Bezugssystem World Geodetic System von 1984[4] wird von der Agri Con GmbH verwendet.

2.2.2 Datenformate

> *Geoobjekte sind räumliche Elemente, die zusätzlich zu Sachinformationen geometrische und topologische Eigenschaften besitzen und zeitlichen Veränderungen unterliegen können. Kennzeichnend für Geoobjekte sind somit Geometrie, Topologie, Thematik und Dynamik.*[5]

De Lange definiert räumliche Objekte bzw. Geoobjekte ausreichend. Ein Geoobjekt enthält als Geometrie eine oder mehrere zwei- oder dreidimensionale Koordinaten, was

[3] [Kud07, S.506]
[4] EPSG:4326
[5] [Lan13, S.133]

die Lage, den Umfang und die Ausdehnung beschreibt. Zur Topologie zählt de Lange Umgebungen, Nachbarschaften, Teilmengen und Überlagerungen. Weiterhin werden Geoobjekte mit Sachinformationen gespeichert und je nach Anwendungsfall versioniert.[vgl. [Lan13, S.133]]

einfache Geoobjekte

Ein Punkt besteht aus einer zwei- oder dreidimensionalen Koordinate und beliebigen Sach-, Topologie- und Dynamikinformationen. Mehrere Punkte bilden Linien. Bildet eine Linie eine geschlossene Fläche, handelt es sich um ein Polygon.

Vektorenmodell

Es besteht die Möglichkeit eine Menge von Punkten als Vektoren aufzufassen und daraus topologische Objekte entstehen zu lassen. Um damit geografisch zu modellieren, ist eine Diskretisierung d.h. eine Zuordnung der Vektoren notwendig.

Rastermodell

Ein Raster löst einen rechteckigen Bereich mit in einem Koordinatensystem gleichmäßig angeordneten quadratischen Bildelementen bzw. Pixeln fester Größe auf. Geodaten werden ergo mit einer indizierten Matrix abgebildet. Ein dreidimensionales Raster heißt Voxel.

> *Ein Punkt wird näherungsweise durch ein einzelnes Pixel dargestellt. Ein Linienzug wird durch entsprechende Anordnungen zusammenhängender Pixel angenähert erfasst. Linienzüge können dann z.B. durch Folgen von Indexpaaren (Zeile, Spalte) der zugehörigen Pixel beschrieben werden. Eine Fläche ist ebenfalls durch zusammenhängende Pixel darstellbar. Somit sind keine weiteren Zusatzinformationen zur Modellierung von Flächen wie im Vektormodell notwendig [...].*[6]

[6] [Lan13, S.136]

Shapefile

Bei Bedarf

2.2.3 Operationen

Aggregation

TODO

Filterung

TODO

Geostatistik

TODO

Interpolation

TODO

Resampling

Bei Bedarf

2.2.4 GIS

Ein GIS ist wie folgt definiert:

> *Ein System, das auf einen Datenbestand zurückgreift und Auswertungen dieser Daten zulässt, so dass Informationen abgeleitet und wiedergegeben werden können, kann allgemein als ein Informationssystem bezeichnet werden. [...]*
>
> *Im Mittelpunkt der Geoinformatik stehen mit den Geoinformationssystemen raumbezogene Informationssysteme, die im Gegensatz zu den übrigen Informationssystemen Geoobjekte der realen Welt modellieren und diese in ein digitales Informationssystem abbilden [...]. Die Gegenstände eines Geoinformationssystems besitzen wie auch bei allen anderen Informationssystemen eine Thematik (und Dynamik). Das Besondere bei Geoinformationssystemen ist, dass Geoobjekte darüber hinaus Geometrie und Topologie als implizite und untrennbare Bestandteile aufweisen! Die Verarbeitung derartiger raumbezogener Informationen erfordert spezielle Werkzeuge bzw. Funktionen, die von den übrigen Informationssystemen nicht bereitgestellt werden [...].*[7]

2.2.5 GeoTools

GeoTools ist eine in Java geschriebene Open Source Bibliothek welche Standardkonforme Operationen zur Verarbeitung von geografischen Daten bereitstellt. Die Implementation erfolgte nach Anforderungen des Open Geospatial Consortium (OGC), worauf beispielsweise Geometrien des Java Topology Suite (JTS) unterstützt werden und die OGC Filter Encoding Spezifikation von Attributen und räumlichen Filtern verwendet wird.(vgl. [Geo15]) [Geo15] listet die wichtigsten Funktionalitäten wie folgt auf:

- *A clean data access API supporting feature access, transaction support and locking between threads*
  - *Access GIS data in many file formats and spatial databases*
  - *Coordinate reference system and transformation support*
  - *Work with an extensive range of map projections*

[7] [Lan13, S. 337]

  - *filter and analyze data in terms of spatial and non-spatial attributes*

- *A stateless, low memory renderer, particularly useful in server-side environments.*

  - *compose and display maps with complex styling*

  - *vendor extensions for fine control of text labels and color blending*

- *Powerful schema asisted parsing technology using XML Schema to bind to GML content*
  The parsing / encoding technology is provided with bindings for many OGC standards including GML, Filter, KML, SLD, and SE.

Unterstützte Formate sind nach der selben Quelle:

- *raster formats and data access*
  arcsde, arcgrid, geotiff, grassraster, gtopo30, image (JPEG, TIFF, GIF, PNG), imageio-ext-gdal, imagemoasaic, imagepyramid, JP2K, matlab

- *Database "jdbc-ng" support*
  db2, h2, mysql, oracle, postgis, spatialite, sqlserver

- *Vector formats and data access*
  app-schema, arcsde, csv, dxf, edigeo, excel, geojson, org, property, shapefile, wfs

- *XML Bindings*
  Java data structures and bindings provided for the following: xsd-core (xml simple types), fes, filter, gml2, gml3, kml, ows, sld, wcs, wfs, wms, wps, vpf. Additional Geometry, Filter and Style parser/encoders available for DOM and SAX applications.

2.2.6 PostGIS

PostGIS ist eine geografische Erweiterung der Objekt-relationalen Datenbank PostgreSQL. PostgreSQL wird dabei um geografische Datentypen, geografische Indizes und Funktionen erweitert. Konkret wird der Simple Feature Access Standard verwendet und

um den Datentyp raster und weitere Funktionen zur Datenverarbeitung erweitert. (siehe [OSG15b]) Somit kann mit SQL direkt mit geografischen Daten gearbeitet werden. PostGIS steh unter der General Public License (GPL)v2.

2.3 NoSQL

2.3.1 Definition

NoSQL steht für eine Bewegung des letzten Jahrzehnts, in welcher die Abkehr von klassischen relationalen Systemen gefordert oder zumindest ein Umdenken bestehender Strukturen, Vorgehen und Grundsätze angestrebt wird. Dies wird durch andere Abfragesprachen, nicht relationale Datenbanksysteme oder Neudefinitionen von Begriffen wie der Konsistenz zum Ausdruck gebracht. Der Ursprung wird in der Literatur verschieden hergeleitet, jedoch wird immer zu den ersten Vertretern der NoSQL Bewegung Systeme mit einer anderen Abfragesprache und einfache Schlüssel-Hash Datenbanken gezählt. Auf einer Messe zu aktuellen Trends im Datenbankbereich wurde der Begriff NoSQL zuerst öffentlich für Lösungen dieser Bewegung verwendet (vgl. [Eva09]) und ist seitdem ein Sammelbegriff für eine hohe Anzahl an Systemen.

NoSQL GIS

In Bezug auf NoSQL kann GIS wie in 2.2.4 definiert werden, jedoch muss das zugrunde liegende System nicht relational sein. Im Rahmen dieser Arbeit ist mit GIS ein System oder die Teilsysteme zur räumlichen Datenhaltung, Datenverarbeitung und Bereitstellung gemeint.

2.3.2 Kategorisierung

Edlich unterscheidet, wie andere Autoren, NoSQL Datenbanken nach vier Kategorien. Jedoch kann eine eindeutige Zuteilung nicht für jedes System erfolgen, da Prinzipien verschiedener Kategorien auf eines zutreffen können. Unter `http://nosql-database.`

org/ ist eine persönliche Übersicht der NoSQL Datenbanken von Herrn Edlich dargestellt. Für dieses Kapitel diente wesentlich [Jun13] als Quelle.

Key Value Datenbank

Key Value Datenbanken speichern Daten in Tupeln aus Schlüssel und Wert. Der Key ist eine Zeichenkette oder ein Hashwert und der Datentyp von Value ist beliebig im Rahmen der Datentypen der Datenbank. Datenzugriff erfolgt über Key. Es existiert keine einheitliche Abfragesprache.

Erste Datenbanken die zu NoSQL zugeordnet werden sind Key Value Datenbanken. Konkret DBM und BerkleyDB. Aktuelle Vertreter sind Amazon Dynamo, Riak, Voldemort und Redis.

Diese Datenbanken eignen sich für heterogene Daten, horizontale Skalierung und Schemafreiheit, da diese einfach strukturierten Daten sich in keiner Relation zueinander befinden.

Dokumentenbasierende Datenbank

Hierbei werden strukturierte Daten, hier Dokumente, unter einem Hash abgelegt und sind über diesen abrufbar. Diese Dokumente sind im großteil der dokumentenbasierten Datenbanken versioniert. Häufige Formate sind JavaScript Object Notation (JSON), Binary JSON (BSON) und YAML.

Ziel ist hier schemafreie Daten zu speichern und den Zugriff zu skalieren. Dabei können zumeist keine Joins verwendbar.

Bekannte Vertreter sind MongoDB, CouchDB und Terrastore.

Spaltenorientierte Datenbank

Im Gegensatz zu zeilenorientierten Datenbanken legen spalteniorientierte Datenbanken ihre Werte, hier Attribute einer Tabelle, spaltenweise ab.

Dies eignet sich für OLAP und Data Warehouse, da Spalteneinfügungen kostengünstig und Garbage Collection effektiv ist. Dagegen besteht ein hoher Aufwand beim Lesen und Schreiben von zusammengehörigen Spaltendaten.

Googles Big Table erweitert diesen Ansatz und beschreibt es in dessen Paper wie folgt:

> *A Bigtable is a sparse, distributed, persistent multi-dimensional sorted map. The map is indexed by a row key, column key, and a timestamp; each value in the map is an uninterpreted array of bytes.*[8]

Die mehrdimensionalen Tabellen oder Maps sind vom Format:
$n * [Domain/Keyspace]\ x\ [item/Column\ Family]\ x\ [Key\ x] * n * [key + Value]$

Googles Ansatz wurde OpenSource in HBase und Cassandra umgesetzt. Die konkrete Implementierung von Google wurde jedoch nicht veröffentlicht. HBase verwendet folgendes Format: Pro Table Zugriff auf Zeilen per Rowkey, diese enthalten Column Familys oder Spalten welche wiederum eine Map namens Column Qualifier mit Tupeln aus der Version als Schlüssel und ein Byte-Array als Wert besitzt.(vgl. [Hä13, S.13])

Graphenbasierte Datenbank

Der bekannteste Vertreter der graphenbasierten Datenbanken ist Neo4J. Alle Daten und deren Beziehungen werden in Form von Graphen persistiert. Ein Graph besteht dabei aus Knoten und gerichteten Kanten. Knoten sind dabei strukturierte Objekte und Kanten Beziehungen zwischen den Objekten. Diese strukturierten Objekte sind Key Value Tupel. Kanten können typisiert sein. Somit lassen sich direkt Beziehungen zwischen Daten definieren, was sich für semantic web, social network, Bioinformatik und Internetrouting eignet.

Diese Datenbanken sind nur optional mit einem Schema versehen und besitzen keine einheitliche Abfragesprache. Auch sind im allgemeinen keine Joins vorgesehen.

[8] [FC06, S.1]

2.3.3 Hadoop

Hadoop ist ein unter der Apache Lizenz 2.0 stehendes Java-Framework zur Datenhaltung und Verarbeitung von großen Datenmengen auf einem Verbund von mehrerern Computern. Es basiert auf MapReduce und dem Dateisystem HDFS.

Das Hadoop File System (HDFS) ist ein verteiltes Dateisystem, welches keine besonderen Anforderungen an die Hardware stellt und für die Verwendung von mehreren hundert bis tausend Computern[9] ausgelegt ist. Es besitzt eine hohe Fehlertoleranz und ist für den Einsatz auf kostengünstiger Hardware ausgelegt. Hoher Datendurchsatz und die Verwendung großer Dateien[10] sind wesentliche Merkmale.(vgl. [Bor07, S.3]) Die Datei-Blöcke werden redundant auf die Knoten verteilt und sind mit Hilfe des Name-Node abrufbar.(vgl. [Hä13, S.7])

Die verteilte Verarbeitung übernimmt MapReduce. Entsprechend dem Namen entspringt der Name MapReduce aus der funktionalen Programmierung, in welcher die Funktionen „map“ und „reduce“ zum Einsatz kommen. So werden hier die Daten mit einer map-Funktion verändert und mit reduce-Funktion aggregiert. Ein Master weist die Daten und Funktionen den Slaves[11] zu. Die Slaves führen die Funktionen mit den ihnen zugewiesenen Daten aus und speichern ihre Ergebnisse auf deren Festplatte ab. MapReduce wurde von Google definiert. In Abbildung 2.1 ist der beschriebe Ablauf dargestellt. Auch hier werden keine besonderen Anforderungen an die Hardware gestellt.(vgl. [JD04, S.3])

Hadoop besitzt eine Master-Slave Architektur, wobei der Name-Node[12] ankommende Anfragen bearbeitet und die Slave-Knoten organisiert. Hadoop ist per API verwendbar und bietet sich somit zur Stapelverarbeitung an. Es wird meist nur als Grundgerüst verwendet und mit Datenbanken wie HBase, MongoDB oder PostgreSQL sowie mit Frameworks für die Nutzung wie Hive, Pig, Spark oder Scalding erweitert.

[9] die in einem verteilten System teilnehmenden Computer heißen Knoten
[10] eine Datei kann mehrere Gigabyte bis mehrere Terrabyte groß sein und wird in Blöcke gleicher Länge aufgeteilt
[11] In diesem Zusammenhang auch Worker genannt
[12] damit ist der Master-Knoten gemeint, auch Jobtracker genannt

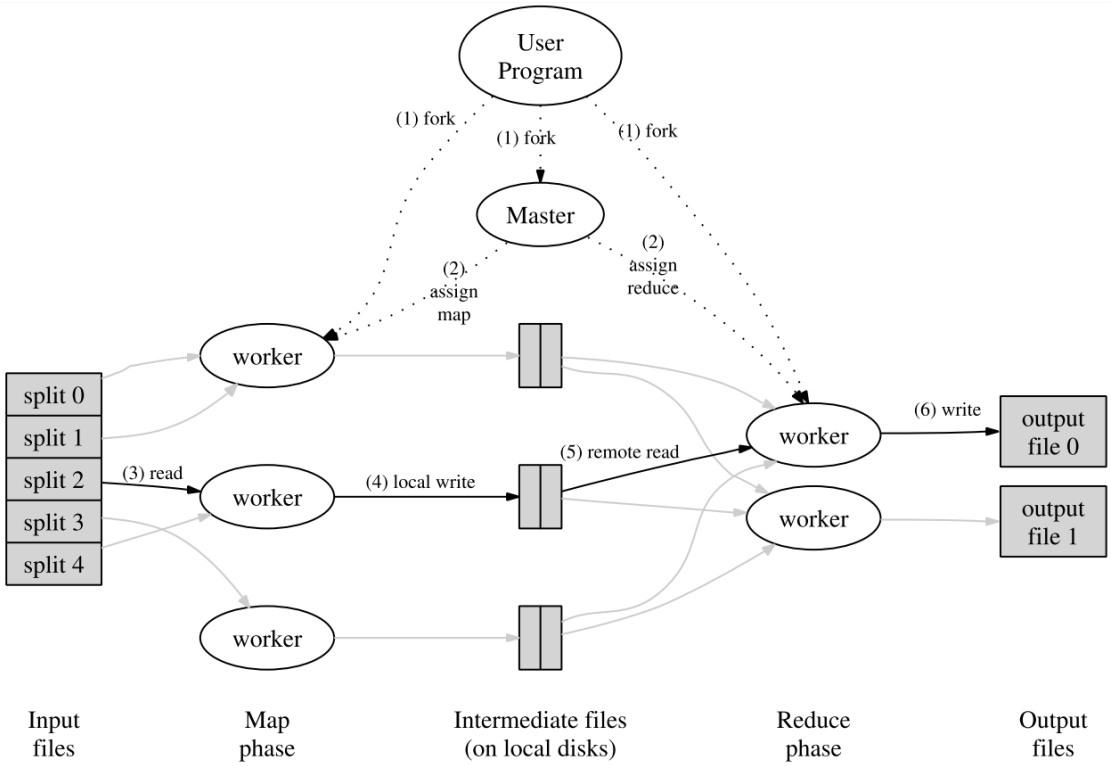

Abbildung 2.1: Übersicht der Ausführung von Googles MapReduce, Quelle: [JD04] S. 3

2.3.4 ZooKeeper

Das Apache Projekt ermöglicht verteilten Prozessen über ZNodes miteinander zu kommunizieren. Es wird häufig gleichzeit mit Hadoop[13] eingesetzt. Ziel ist dabei ein hoher Durchsatz, geringe Latenzen, Hochverfügbarkeit und effektiver Zugriff durch die Prozesse. Dabei verwaltet ZooKeeper eine geringe Datenmenge von einigen Kilobyte, da einzig Metainformationen von Interesse sind.[14]

2.3.5 Thrift

> *The Apache Thrift software framework, for scalable cross-language services development, combines a software stack with a code generation engine to build services*

[13] siehe 2.3.3
[14] siehe [Apab]

that work efficiently and seamlessly between C++, Java, Python, PHP, Ruby, Erlang, Perl, Haskell, C#, Cocoa, JavaScript, Node.js, Smalltalk, OCaml and Delphi and other languages.[15]

2.3.6 Accumulo

Hierbei handelt es sich um ein Level-1-Apache Projekt, es ist eine Java Open-Source Implementation des BigTable Ansatzes von Google und wird seit 2008 entwickelt. Es verwendet Hadoop[16], ZooKeeper[17] und Thift[18]. Der BigTable Ansatz wird um Iteratoren, Zellenbezeichnungen, Constraints, Fragmentierungsmöglichkeiten einer dokumentbasierten Datenbank und die Unterstützung der gleichzeitigen Verwendung mehrerer HDFS namenodes erweitert. Weitere Funktionen sind folgende:

- Verwendung mehrerer Master
- Verwendung einer eigenen Zeitsynchronisation
- Eingebaute temporäre Datenhaltung im Arbeitsspeicher
- Bereitstellung von Testimplementierungen per API

Es existieren weiterhin verschiedene Erweiterungen zum Datenmanagement und Änderung des Ordnungsrahmens zur Verfügung. [Apa14a]

2.3.7 GeoMesa

GeoMesa ist eine freie[19] geografische Datenbank der Firma LocationTech mit den Möglichkeiten der verteilten Verarbeitung und Versionierung von geografischen Daten. Dieses Framework ist in Scala geschrieben. Es erweitert Accumulo[20], unterstützt die GeoTools API und bietet ein Plugin für den Mapserver GeoServer an. Die Daten werden

[15] [Apaa]
[16] siehe2.3.3
[17] siehe 2.3.4
[18] siehe 2.3.5
[19] Apache License Version 2.0
[20] siehe 2.3.6

nach Geohash[21] verwaltet. (vgl. [Fox14])
GeoMesa wird in Verbindung mit stream processing[22] und batch processing[23] verwendet. Zur räumlichen Datenverarbeitung werden Scala Bibliotheken wie JTS und GeoTools eingesetzt. Vorrangig werden Vektordaten von GeoMesa verarbeitet, durch eine optionale Erweiterung sind auch Rasterdaten verwendbar. Datenimport wird ingest genannt, erfolgt über die Kommandazeile und unterstützt die Datenformate CSV, TSV und SHP. CSV, TSV, Shapefile, GeoJSON, and GML können dagegen über den selben Weg exportiert werden. Weiterhin erfolgt der Datenexport und -import über Scala. GeoMesa ist gedacht, um initial große Datenmengen per Ingest zu laden und anschließend diese mit Spark und dafür vorgesehenen Frameworks sowie Bibliotheken zu verarbeiten.

2.3.8 Postgres-XL

Postgres-XL ist ein frei verfügbares Clustersystem für PostgreSQL unter der Mozilla Public License. XL steht dabei für eXtensible Lattice, erweiterbarer Verbund. Damit soll es ermöglicht werden, mit PostgreSQL verteilt Schreiboperationen zu skalieren sowie parallele Datenverarbeitung auf mehreren physischen und virtuellen Systemen gleichzeitig zu betreiben. Dafür wird zur verteilten Datenhaltung ACID mit MVCC und zur parallelen Verarbeitung ein Massively Parallel Processing (MPP) Mechanismus eingesetzt. (siehe [Tra15b]) Die Postgres-XL Umgebung nutzt mehrere PostgreSQL Instanzen oder Partitionen und bietet eine Schnittstelle für alle Instanzen an.
Abbildung 2.2 verbildlicht den Aufbau. Laut Abbildung wird als erstes ein Load-Balancer angesprochen. Dies wird in der Dokumentation nicht belegt. Die anderen Elemente sind nach [Tra15b] wie folgt beschrieben:

Global Transaction Manager Dient als Verwaltungselement der Transaktionen und realisiert MVCC über das System. Laut Dokumentation existiert für jedes PostgreSQL Element ein GTM, um MVCC mit je einem globalen Kontext realisieren zu können.

[21] siehe 2.1.2
[22] bspw. Spark oder Storm
[23] bspw. Pig oder Cascading

Coordinator „The Coordinator manages the user sessions and interacts with GTM and the data nodes. The Coordinator parses and plans queries, and sends down a serialized global plan to each of the components involved in a statement.“ ([Tra15a])

Data Node Diese Elemente enthalten PostgreSQL. Diese müssen sich nicht eine Datenbank replizieren, sondern können auch eine Datenbank über partitioning teilen. Anfragen können von verschiedenen Coordinators gleichzeitig in unterschiedlichen Sitzungen erfolgen. Auf Grund der Kapselung besitzt jeder Data Node seinen eigenen Kontext zur Transaktion.

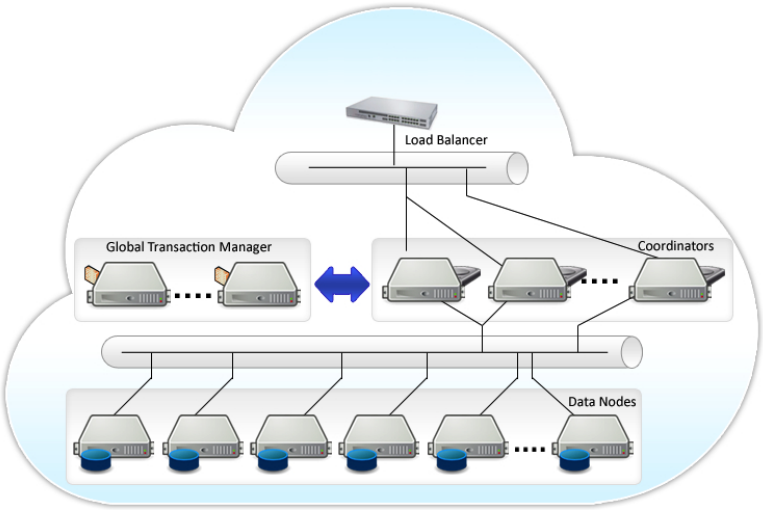

Abbildung 2.2: Aufbau Postgres-XL, Quelle: http://www.postgres-xl.org/wp-content/uploads/2014/04/xl_cluster_architecture1.jpg

Das System wird analog einer PostgreSQL Instanz angesprochen. Ebenso ist PostGIS verwendbar.

2.3.9 Rasdaman

Rasdaman ist ein Array-Datenbanksystem speziell zum speichern und verarbeiten von Rasterdaten. Es erweitert eine relationale Datenbank und wird mit multi-dimensionalität der Daten, einer eigenen SQL ähnlichen Abfragesprache, Parallelisierung und Skalierbarkeit in beliebigen Maßstab sowie OGC konformen Diensten beworben. Es ist als Client bzw. API unter der Lesser General Public License (LGPL) 3 und als Server unter

der GPL 3 für Linux, MacOS und Solaris verfügbar. Als OGC konforme Dienste werden WMS 1.3, WCS 2.0, WCS-T 1.4, WCPS 1.0 und WPS 1.0 bereitgestellt. Die API kann in Java, C++ und über die eigene Abfragesprache rasql verwendet werden. (vgl. [Liv14]) Der beschriebene Aufbau ist unter Abbildung 2.3 dargestellt.

Es besteht die Möglichkeit, Rasdaman zu einer bestehenden PostgreSQL zu installieren und direkten Datenaustausch zwischen den beiden Systemen zu ermöglichen. Weiterhin kann Rasdaman in Verbindung mit GDAL verwendet werden. Momentan existiert eine Community und eine Enterprise Variante. Dabei verfügt die Enterprise Variante über mehr Features wie beispielsweise Datenkomprimierung, Serververwaltung per Webbrowser, Laufzeitoptimierungen und verschiedene Datenbankschnittstellen. Von der verwendeten Datenbank wird BLOB als Datenbankinterner Datentyp verwendet. (vgl. [Ras14])

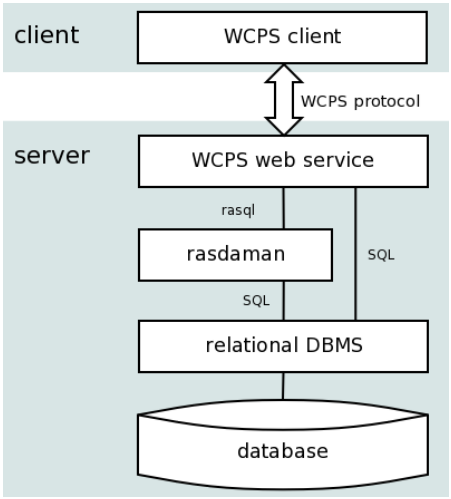

Abbildung 2.3: Aufbau Rasdaman, Quelle: http://www.rasdaman.org/raw-attachment/wiki/Technology/wcps-stack.png

2.3.10 Spacebase

Spacebase ist eine in Java programmierte geografische Datenbank mit Betonung auf Echtzeit und geringe Latenzen des Unternehmens Parallel Universe. Die Datenbank wird ausschließlich im Arbeitsspeicher gehalten und ausgeführt. Außerdem ist sie für

die verteilte Nutzung auf mehreren Computern konzipiert. Räumliche Daten werden in 2D oder 3D mit einem dazugehörigen begrenzenden Rechteck im R-Baum gespeichert. SpaceBase ist für eine sehr große Anzahl an Abrufen und Veränderungen der Daten in Echtzeit geeignet. Für die räumliche Verarbeitung stehen Bereichsabfragen, Überschneidungsabfragen und allgemeine Join Abfragen zur Verfügung. Diese beschränken sich aber auf das begrenzende Rechteck, da die eigentliche Geometrie nur über eine Referenz im Datenobjekt abrufbar ist. Ergänzend besteht die Möglichkeit eigene Abfragen zu formulieren und dabei die referenzierte Geometrie zu verwenden. Es wird jedoch darauf hingewiesen, dass complexe Operationen auf diese Geometrie die Laufzeit der Berechnungen wesentlich erhöht. Zum Erreichen der geringen Latenzen werden räumliche Berechnungen parallelisiert und auf die Instanzen verteilt. Es stehen APIs für Java, C++, Erlang, Python, Ruby und Node.js zur Verfügung. (vgl. [Uni14])

2.3.11 ESRI GIS Tools for Hadoop

Das Unternehmen ESRI stellt eine Umgebung zum verteilten Verarbeiten geografischer Daten auf einem Hadoop Cluster bereit. Diese Umgebung nennt sich GIS Tools for Hadoop und besteht aus vier Teilen:

Esri Geometry API for Java Java API für Entwickler, um mit Hadoop geografische Datenverarbeitung durchzuführen. JSON, WKT und Shapefile sind Importformate. Die Datentypen werden als einfache Geometrien bezeichnet und setzen sich laut Quellcode aus Punkt, Punktsammlung, Linie, Polygon und PolyLine zusammen. Geografische Funktionen sind „union, difference, intersect, clip, cut, and buffer“ sowie „equals, within, contains, crosses, and touches“. (vgl. [ESR14a])

Spatial Framework for Hadoop Erweiternte Bibliothek für Hadoop, um mit HQL geografisch zu arbeiten.

Geoprocessing Tools for Hadoop Hadoop Anbindung an ArcGIS.

GIS Tools for Hadoop Projekt zum testen der Hadoop Umgebung mit ArcGIS.

ArcGIS ist ein proprietäres GIS, die beschriebene Umgebung steht dagegen unter Apache Lizenz Version 2.0 zur Verfügung. (vgl. [ESR14b])

2.4 Tests

Die zu analysierenden Systeme sind mit Hilfe von Tests zu untersuchen und zu vergleichen. Diese Tests müssen einerseits vergleichend sein, ergo bei unterschiedlichen Systemen die gleichen Merkmale testen und wiederholbar sein, andererseits die relevanten Merkmale testen.

Es handelt es sich somit um systematische Tests:

> *Ein systematischer Test ist ein Test, bei dem*
> *die Randbedingungen definiert oder präzise erfasst sind,*
> *die Eingaben systematisch ausgewählt wurden,*
> *die Ergebnisse dokumentiert und nach Kriterien beurteilt werden, die vor dem Test festgelegt wurden.*[24]

Nachfolgend werden die Randbedingungen definiert und in Kapitel 4.1 die Ein- und Ausgaben dargestellt.

2.4.1 Funktionstests

Um die Systeme auf Softwarequalität, beschrieben auf Seite 31, zu testen, sind Funktionstests notwendig. [Lud07] verweist auf Funktionstests auf Seite 455 wie folgt:

> *Werden die Testfälle auf Basis der in der Spezifikation geforderten Eigenschaften des Prüflings ausgewählt (z.B. Funktionalität, Antwortzeit), dann spricht man von einem Block-Box-Test oder auch von einem Funktionstest (19.5).*

Dazu wird auf Seite 471 der Umfang des Funktionstestes wie folgt umrissen:

> *Ein umfassender Black-Box-Test sollte*
> *alle Funktionen des Programms aktivieren (Funktionsüberdeckung),*
> *alle möglichen Eingaben bearbeiten (Eingabeüberdeckung),*
> *alle möglichen Ausgabeformen erzeugen (Ausgabeüberdeckung),*
> *die Leistungsgrenzen ausloten,*
> *die spezifizierten Mengengrenzen ausschöpfen,*
> *alle definierten Fehlersituationen herbeiführen.*

[24] [Lud07, S.446]

Im Rahmen dieser Arbeit soll dieser Umfang wie folgt begrenzt werden:

Funktionsüberdeckung Zur Einschätzung der Eignung eines Systems für den Anwendungsfall sind ausgewählte Funktionen zu testen, Im allgemeinen besitzen die ausgewählten Systeme Funktionen die in diesem Rahmen nicht genutzt werden sollen, jedoch auch Funktionen die von Hand ergänzt werden müssen. Die notwendige Menge an Funktionen soll einzig getestet und somit abgedeckt werden.

Eingabeüberdeckung Auch hierbei stehen verschiedene von den Systemen bereitgestellte Schnittstellen und Austauschformate zur Verfügung, aber es sollen nur die relevanten untersucht werden.

Ausgabeüberdeckung siehe Eingabeüberdeckung

Leistungsgrenzen Dafür werden eigene Tests definiert und verwendet.[25]

Mengengrenzen Die zu untersuchenden Systeme eignen sich zum Speichern großer Datenmengen die mehrere Terrabyte bis Petabyte umfassen. Da die vorhandenen Datenmengen diese Größe unterschreitet, müssen die Mengengrenzen nicht wesentlich getestet werden.

Fehlersituationen Es besteht nicht das Ziel die Fehleranfälligkeit als einzelnes Merkmal zu untersuchen, weshalb dafür keine Testfälle erstellt oder durchgeführt werden. Einzig die Korrektheit der Berechnungen wird überprüft, indem die Ergebnisse des aktuellen ist-Standes zum Vergleich herangezogen werden.

2.4.2 Leistungstests

- Lasttests zur Persistierung von Eingangsdaten - Lasttests zur Bereitstellung von verarbeiteten und eingabedaten (an UMN[plugin, WMS, Bilder oder über pgsql] oder allgemein) - Überwachung der Systemauslastung ist notwendig

Ein Kriterium der Untersuchung in dieser Arbeit ist die Leistung. Nach Kesselman in [Han95, S.20] ist Leistung die gewichtete Summation von Leistungsindizes, wobei ein

[25] siehe 2.4.2

Leistungsindex die Quantifizierung einer Eigenschaft eines Systems ist. Wesentliche Leistungsindizes sind Laufzeiten von einfachen oder komplexen Operationen. Es wird hier nur die spezielle Leistung gemessen, da ausgewählte Eigenschaften betrachtet werden und somit nicht die Leistung für einen allgemeinen Anwendungsfall.

Der in diesem Zusammenhang in der Literatur verwendete Begriff Benchmark ist hier jedoch ungeeignet, da die Software und nicht die Hardware untersucht werden soll:

> *Benchmarking ist eine Methode der Analyse des Leistungsverhaltens von Rechensystemen anhand von Referenzprogrammen oder Sätzen von Referenzprogrammen (Benchmarks). Dabei wird das Leistungsverhalten verschiedener Rechensysteme in Relation gesetzt, um so Vergleichskriterien für Rechensysteme zu erhalten.*[26]

Eine hier betrachtete Leistungs- und Laufzeitmessung ist dabei wie folgt definiert:

> *Unter Leistungsmessung versteht man die Beobachtung des Ablaufverhaltens eines Programms bei der Ausführung auf einem realen System. Das Ziel das dabei verfolgt wird, ist die Gewinnung von Erkenntnissen, die zur Optimierung eines Programms genutzt werden können.*[27]

Die aus der Leistungsmessung gewonnenen Erkenntnisse dienen hierbei als Qualitätsmerkmale und werden nach definierten Metriken gewichtet.

Es existieren vordefinierte Leistungstests, dabei sind jene des Transaction Processing Performance Council (TPC) sowie die Benchmarks 001, 007, HyperModel und Justitia zu nennen. Diese sind jedoch nicht für den zu untersuchenden Anwendungsfall geeignet, da sie die allgemeine Leistungsfähigkeit und Effektivität messen und somit nicht die gesuchten Werte der GIS ermitteln.

[26] [Han95, S.24]
[27] [Han95, S.28]

3 methodisches Vorgehen

Das Thema "Untersuchung und Optimierung von verteilten geografischen Informationssystemen zur Verarbeitung agrartechnischer Kennzahlen" besteht aus vier Unteraufgaben:

Untersuchung bestehender Frameworks anhand von Qualitätsmerkmalen
Die erste Hälfte dieser Arbeit ist eine Softwareauswahl im Bereich der Datenverarbeitung. Es handelt sich dabei nicht im Anwendersoftware, wodurch keine empirischen Studien der Benutzung der Frameworks durch Anwender und keine Kopplung mit der Unternehmensstruktur notwendig ist. In diesem speziellen technischen Kontext sind wissenschaftliche Vorgehensweisen zur Softwareauswahl notwendig. Dafür geeignet ist eine Nutzwertanalyse. Darin soll mit Mitteln des Softwarequalitätsmanagement und allgemein Prinzipien der Softwareentwicklung eine nachvollziehbare, auch auf ähnliche Projekte übertragbare und wissenschaftlich korrekte Vorgehensweise erreicht werden.
Aus gegebenen Anforderungen[1] sind Qualitätsmerkmale zu erstellen. Davon ist die Mindestmenge welche zur Eignung eines Frameworks notwendig ist bzw. die notwendige Qualität zu definieren. Darauf aufbauend sind Qualitätsmetriken zu erstellen, welche die einzelnen Qualitäten messbar machen. Eine Menge von scheinbar geeigneten Frameworks ist anhand der definierten Metriken zu untersuchen. Dabei sind jedoch nur die wesentlichen Qualitäten zu untersuchen. Aus bekannten GIS Frameworks werden vielversprechende ausgewählt und somit die Menge der zu untersuchenden Frameworks erstellt. Diese Vorauswahl erfolgt anhand der Tabelle „Table of free systems especially for spatial data processing“ in [AT15]. Wikpedia dient dabei als Quelle, da diese Website von Arbeitnehmern genutzt wird und somit diese Community die Tabelle aktuell

[1] siehe 4.1

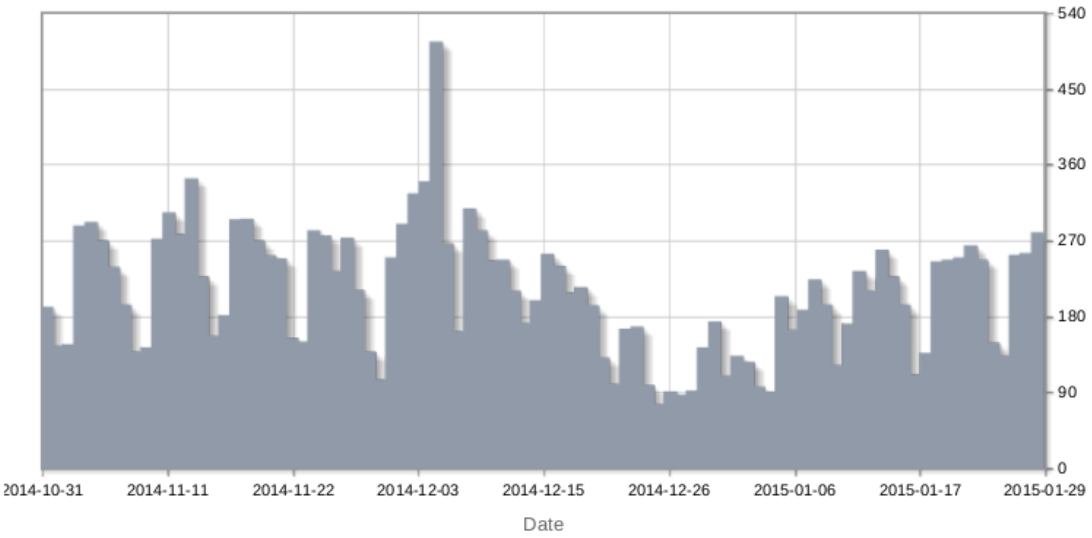

Abbildung 3.1: Nutzungsstatistik von 90 Tagen der Wikipedia Seite „Spatial database“ nach [Wik15c]

und ausreichend vollständig hält. Dies ist in der Aufrufstatistik begründet, welche in Abbildung 3.1 dargestellt ist. So wurde die Website am Mittwoch dem 21.1.2015 250 mal, dagegen am Samstag dem 17.1.2015 111 mal aufgerufen. Allgemein ist an Tagen an Wochenenden etwa die Hälfte der Aufrufe pro Tag gegenüber Montag bis Freitag zu beobachten. Weiterhin war zu den Weihnachtsfeiertagen eine wesentlich geringere Anzahl an Aufrufen zu verzeichnen: 1650 Aufrufe zwischen dem 22.12.2014 und 4.1.2015 im Gegensatz zu 3005 Aufrufe zwischen dem 8.12.2014 und 21.12.2014. Diese Aufrufstatistik belegt das die Nutzung der Website etwa zur Hälfte durch Arbeitnehmer erfolgt und somit für Unternehmen interessant ist.

Schwer messbare Kriterien zur Softwareauswahl wie Marktposition, Support und Community fließen in diese Bewertung nicht ein, werden jedoch im Sinne der Agri Con ergänzend festgehalten.

Auswahl eines Frameworks

Aus der untersuchten Menge ist eines anhand der gemessenen Qualität auszuwählen. Der Ist-Zustand[2] der Agri Con ist über Jahre hinweg durch wachsende Anforderungen im technischen und organisatorischen Bereich entstanden. Aus diesem Grund ist die

[2] siehe 4.2

Auswahl eines Frameworks für den gesuchten Anwendungsfall aufwendig. Die Wahrscheinlichkeit der Eignung mehrerer Frameworks scheint daher gering. Dies ist eine subjektive Einschätzung des Autors, was es in dieser Teilaufgabe zu beweisen gilt. Eine detaillierte Bewertung der Frameworks anhand aller Qualitäten würde danach dazu führen, dass kein Framework geeignet erscheint. Deshalb ist die Auswahl eines Frameworks zunächst anhand deren Spezifikationen durchzuführen.

Entwurf eines Prototypen
Das ausgewählte Framework ist detailliert zu untersuchen. Aus dieser Untersuchung soll ein Entwurf zum Einsatz bei der Agri Con entstehen. Dabei ist besonders dessen Architektur zu beleuchten, eine Konfiguration zu erstellen und fehlende Funktionalitäten mit nachträglicher Implementierung in das Framework einzubinden. Auf Grund der wesentlich unterschiedlichen Frameworks, kann vor der Auswahl keine konkrete Architekturempfehlung verwendet werden. Dieser Entwurf ist nach den Anforderungen und den Eigenheiten des Frameworks zu erstellen. Eine wesentliche Grundlage sind dabei Referenzimplementierungen und Guidelines des entsprechendem Rahmenwerkes.

Prototypische Implementierung
Der Entwurf wird schlussendlich umgesetzt und anhand der Metriken mit Funktions- und Leistungstests bewertet. Auch diese Bewertung erfolgt im Rahmen einer Nutzwertanalyse, jedoch detaillierter. Ziel ist dabei die Eignung des Prototypen hinsichtlich des geforderten Einsatzzweckes darzustellen. Dafür werden die definierten Qualitäten herangezogen. Außerdem soll eine Einschätzung aus Entwicklersicht gegeben werden, welche Preis, Marktposition, Support und Community einschätzt. Diese Ergänzung ist notwendig, um weiterhin den Aufwand zur Einbindung des Frameworks in die Unternehmensstruktur und die Zukunftsfähigkeit einschätzen zu können.

Für die grobe Nutzwertanalyse zur Auswahl eines Framworks ist im nachfolgenden Kapitel die Darstellung des Ist-Standes sowie die Anforderungen an das Framework zu finden. Die Definition der Testfälle zur Datenerhebung für die detaillierte Nutzwertanalyse des Prototypen schließt sich daran an.

4 Ausgangsszenario

4.1 Anforderungen

Aktuelle Möglichkeiten der Datenerfassung über Sensoren und moderne Probenahmegeräte führen zu mehr und mehr Datensätzen, die für einen Landwirtschaftsbetrieb ausgewertet werden müssen. Darüber hinaus besteht die Notwendigkeit, Daten Jahresübergreifend und betriebsübergreifend auszuwerten, um pflanzenbauliche Zusammenhänge über statistische Methoden untersuchen zu können. In den letzten 3 Jahren wurde beispielsweise nur zum Thema N-Versorgung[1] für einen Betrieb etwa 800 Datensätze mit 1,9 Mio Einträgen erfasst. Alle diese Daten haben einen räumlichen Bezug, sie müssen weiterverarbeitet, kartographisch aufbereitet und dargestellt werden.
Daraus ergeben sich verschiedenen Anforderungen an die Technologie, die für die Verarbeitung, Analyse und Darstellung verwendet wird:

- PostgreSQL mit PostGIS zum Datenimport und -export nutzbar
- Gruppieren und Filtern mit geringer Laufzeit
- parallele Berechnung[2] über große Datenmengen mit geringer Laufzeit
- Räumliche Berechnungen wie Verschneiden, Berechnen von Overlays
- Unterschiedliche Prinzipien der Kartengenerierung, hier dynamisches rendern aus dem Datenbestand zur Laufzeit oder dynamisches rendern bei Dateneingang wodurch vorgerenderte Karten bereitstehen
- nutzbare Schnittstelle zur Darstellung mit dem UMN MapServer

[1] Stickstoffdüngung und -aufnahme
[2] hier Statistik bzw. Geostatistik sowie Interpolation

Konkret handelt es sich bei den Eingangsdaten um folgende:

Pflanzenbauliche Daten [3] Punktdaten

Basisdaten wie Feldgrenzen Vektordaten

Externe Satelliteninformationen und Multispektralanalysen Rasterdaten

4.1.1 Softwarequalität

Qualitätsmerkmale sind nach DIN 9126[4] in [boo98, S.258 f.] Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz, Änderbarkeit und Übertragbarkeit. Diese Merkmale werden durch Qualitätskriterien für jeden Anwendungsfall konkretisiert. Nachfolgend werden die Qualitätsmerkmale für diesen Anwendungsfall konkretisiert und darauf die zu untersuchenden aufgelistet. Da die zu analysierenden Systeme eine Datenbank beinhalten, welche mit räumlichen Datentypen arbeitet, wurde die im Anhang C von [Hoh96] enthaltene Checkliste zur Auswahl eines Objektorientiertes Datenbankmanagementsystem (ODBMS) berücksichtigt.

Funktionalität
Das System stellt alle geforderten Funktionen mit den definierten Eigenschaften zur Verfügung.

- **Richtigkeit:** Ergebnisse sind korrekt oder ausreichend genau. Die originalen räumlichen Daten werden von Sensoren erfasst, in wessen Toleranzbereich die Ergebnisse der Verarbeitung durch das Framework liegen müssen.
- **Interoperabilität:** Es sind Schnittstellen zur Ein- und Ausgabe vorhanden. Dabei soll es sich um PostgreSQL Import sowie PostgreSQL und UMN MapServer Export handeln.
- **Funktionsumfang:** Mindestens die benannte und essentielle Menge an Funktionalitäten wird bereitgestellt. Dazu zählt: parallele Verarbeitung, Gruppierungs-, Filter-, Verschneidungs- sowie Overlayfunktionen, Geostatistik und Umrechnung

[3] Sensoren, Bodenuntersuchung, Bonitur, Logger
[4] DIN 9126 wurde durch ISO/IEC 25000 ersetzt, jedoch sind beide nur proprietär verfügbar

zwischen Koordinatensystemen und -formaten. Außerdem sind vorhandene Datentypen und Schemaversionierung von Interesse.

- **Ordnungsmäßigkeit:** Die Implementation des Systems und dessen Funktionen erfüllt Normen, Vereinbarungen, gesetzliche Bestimmungen und andere Vorschriften. Hierzu ist zu nennen, dass besonders Berechnungsfunktionen nach mathematischen Gesetzen implementiert sein müssen. Konkret sind Berechnungen der räumlichen Verarbeitung nach anerkannten definierten Algorithmen durchzuführen.

Zuverlässigkeit

Nach [boo98, S.259] wird Zuverlässigkeit als die Fähigkeit einer Software ihr Leistungsniveau unter festgelegten Bedingungen über einen festgelegten Zeitraum bewahren, definiert. Nutzung von Tools zur Überwachung und Konfiguration immanent.

- **Fehlertoleranz:** Das System sollte auftretende Fehler des Tagesgeschäftes abfangen und weiterarbeiten. Besonders Fehler in den Quelldaten können zu Fehlern während der Ausführung von Berechnungen führen, was per sé abgefangen werden muss.
- **Wiederherstellbarkeit:** Auch die Möglichkeit bei einem schwerwiegendem Fehler Daten und Stände der abgebrochenen Operationen wiederherzustellen ist ein zu betrachtendes Qualitätskriterium.
- **Mean Time to Failure (MTTF):** Diese statische Kenngröße der erfahrungsgemäßen mittleren Lebensdauer ist für kritische Systeme relevant.

Benutzbarkeit

Qualität des Zugangs für Benutzer sowie Eignung für eine oder mehrere Benutzergruppen.

- **Verständlichkeit**
- **Bedienbarkeit**

- **Dokumentation:** Eine ausführliche, aktuelle und korrekte Dokumentation ist Voraussetzung zur produktiven Verwendung.
- **Eignung:** Die angestrebte Benutzergruppe muss mit der aktuellen Benutzergruppe übereinstimmen. Die aktuelle Benutzergruppe ist Programmierer bzw. Administrator.

Effizienz

Das Verhältnis zwischen Auslastung der Hardware und erfolgreich bearbeiteten Aufgaben. Nach [Han95, S.21] ist Leistung paralleler Programme das Verhältnis des Speedups zur Anzahl der verwendeten Prozessoren. Wobei Speedup als Verhältnis der Ausführungszeiten zwischen der auf N Prozessoren ausgeführten parallelen Version eines Programms und der sequentiellen Version des Programmes definiert ist. Diese Definitionen treffen für die zu untersuchenden Systeme zu, da es sich um parallelisierende GIS handelt.

- **Zeitverhalten:** Oder auch Laufzeitverhalten genannt, dient allgemein zur Darstellung des Durchsatzes. Die Skalierung des Systems zählt hier dazu. Dies wird speziell durch zusätzliche Leistungstests beurteilt.
- **Verbrauchsverhalten:** Das Verhältnis aus erbrachter Leistung und dem dafür notwendig gewesenen Aufwand in Form von Hardwarenutzung.
- **Skalierbarkeit:** Anzahl der zu verwendenden Computer um nach dem Speedup eine Effizienzsteigerung im Gegensatz zum Einsatz bei einem Computer zu erreichen.

Änderbarkeit

Aufwand zur Verbesserung oder Anpassung der Umgebung und der Spezifikationen, auch Wartungsaufwand genannt.

- **Analysierbarkeit:** „Aufwand, um Mängel oder Ursachen von Versagen zu diagnostizieren oder um änderungsbedüftige Teile zu bestimmen.“ [boo98, S. 260]

- **Modifizierbarkeit:** Notwendiger Aufwand für Änderungen zum Ziele der Verbesserung und Fehlerbehebung.
- **Stabilität:** Wahrscheinlichkeit vom ungewollten Auswirkungen von Änderungen.
- **Prüfbarkeit:** Oder Testbarkeit als Merkmal, welches die Möglichkeiten und den Aufwand zum testen der originalen und geänderten Systeme.

Übertragbarkeit
Die Fähigkeit das System auf andere Hard- und Software und andere Vorgehensweisen zu migrieren.

- **Anpassbarkeit:** Möglichkeiten des unveränderten Systems Änderungen vorzunehmen.
- **Installierbarkeit:** Systemvoraussetzung und Aufwand zur Installation des Systems.

Nichttechnische Kriterien
Erweiterte Qualitätskriterien, welche nicht nach der DIN 9126 zugeordnet werden können.

- **Herstellerfirma und Produkt:** Dazu zählt die Marktposition, der Preis, die Produktplanung und Service.

Die zu untersuchenden Qualitätskriterien für die Softwareauswahl sind Funktionsumfang, Fehlertoleranz, Dokumentation, Zeitverhalten, Analysier- und Modifizierbarkeit.

4.1.2 Qualitätsmetriken

Zu den wichtigen Qualitätsmerkmalen aus Kapitel 4.1.1 sind folgend Kriterien definiert, sowie je eine Bewertungsvorschrift und die geforderte Mindestbewertung für den Anwendungsfall angegeben.

Richtigkeit:
Berechnungen sind zu 99% korrekt. Ausnahme ist dabei die Berechnung von Koordinaten. Dabei haben die Ergebnisse bis acht Stellen nach dem Komma korrekt zu sein. Die statische Abbildung ist dabei $[korrekt, nicht\ korrekt]$ *nach* $[1,0]$ und die geforderte Mindestbewertung 1.

Interoperabilität:
Import und Export von räumlichen Daten aus PostgreSQL sowie eine Anbindungsmöglichkeit an den UMN MapServer. Statische Abbildung:
$[Datenschnittstelle\ und\ UMN\ Schnittstelle\ vorhanden, Datenschnittstelle\ vorhanden, UMN\ Schnittstelle\ vorhanden, keine\ Schnittstelle\ vorhanden]$ *nach* $[12,7,5,0]$
Der Bereich bis 12 soll die Wichtigkeit des Vorhandenseins der Schnittstellen verdeutlichen. Die geforderte Mindestbewertung ist 7.

Funktionsumfang:
Tabelle 4.1 gibt die Wertung bei Existenz der einzelnen Funktionen wieder. Existiert die Funktion nicht, ist die Wertung Null. Ein Zwischenwert bei eingeschränkter Funktionalität ist möglich. Maximale Wertung: 0
Es müssen mindestens geografische Datentypen und Filterfunktionen vorhanden sein.

Funktion	**Wertung**
parallele Verarbeitung	2
geografische Datentypen	14
Umrechnung zwischen Koordinatensystemen	10
Gruppierungsfunktionen	10
Verschneidungsfunktionen	4
Overlayfunktionen	4
Geostatistik	6
Filterfunktionen	10
Schemaversionierung	1

Tabelle 4.1: Wertungstabelle Funktionsumfang

Fehlertoleranz:
Es gilt zu messen, ob Fehler bei einer Berechnung andere verschränkt gleichzeitig lau-

fende Berechnungen beeinträchtigt. Aus diesem Grund wird Unabhängigkeit auf eins und Abhängigkeit auf null abgebildet. Die geforderte Mindestbewertung ist 1.

Dokumentation:
Vorhandene Dokumentation ist nach einzelnen Themen zu bewerten. Dabei kann ein maximaler Wert von 13 erreicht werden. Die geforderte Mindestbewertung ist 6.

Dokumentation zu	**Wertung je Eintrag**
Installation, Zeitverhalten	1
Funktionsumfang	2
Interoperabilität, Best practise, Anpassbarkeit	3

Tabelle 4.2: Wertungstabelle Dokumentation

Zeitverhalten:
Konkret wird eine Beschleunigung aufwendiger Vorgänge angestrebt. Die statische Abbildung auf Bezug auf die Laufzeiten des Ist-Standes:
$[\textit{Zeit wird berschritten}, \textit{Zeit ist gleich}, \textit{Zeit wird unterschritten}]$ *nach* $[0,1,3]$
Der Abruf von Punktdaten dauert neun Sekunden und die Interpolation ...

Modifizierbarkeit:
Anpassungen des Frameworks hinsichtlich der folgenden Punkte erhöhen den Wert um eins:
Verwendung eigener Datentypen, Erstellung eigener Schnittstellen, Erstellung eigener Funktionen, Verwendung der Programmiersprachen Scala oder R, Anlegen eigener Berechnungsvorgängen zur späteren Abarbeitung
Die geforderte Mindestbewertung ist abhängig vom Funktionsumfang und der Interoperabilität. Jedoch sollten fehlende Funktionen und Schnittstellen erstellt werden können.

4.1.3 Testfälle

Zur definierten und wiederholbaren Erfassung der Erfüllung von speziellen Kriterien wie Funktionalität und Zeitverhalten, folgt die Definition der Funktions- und Lasttests.

Funktionstests

Nach der Definition in 2.4.1 werden hiermit spezielle Funktionalitäten auf Vorhandensein und die Menge der Schnittstellen sowie der Austauschformate getestet. Diese Funktionstest sind nicht automatisiert für unterschiedliche Frameworks durchführbar. Deshalb werden sie manuell anhand der Spezifikation und des Quellcodes durchgeführt und die Ergebnisse tabellarisch festgehalten.

Folgende Schnittstellen sind zu testen:

- PostgreSQL
- PostGIS
- UMN MapServer

Als Austauschformate ist mindestens eines der folgenden Datentypen zum Datenaustausch notwendig:

- Simple Feature Access
- Objekte der JTS
- PostGIS geometry

Speziell in GIS gilt es die Koordinatenreferenzsysteme zu analysieren. Die EPSG Codes 3578 und 4326 werden im Anwendungsfall verwendet.

Für folgende Aufgaben sind die Funktionen zu analysieren:

- Umwandlung zwischen Koordinatensystemen
- Verschneidung von räumlichen Daten
- Geostatistik
- Interpolation
- Kriging
- topologische Filterung
- räumliche Filterung

Dabei ist stets zu berücksichtigen mit welchen Datentypen die Funktionen verwendet werden können.

Lasttests

Die Basis für die Erstellung von Lasttests ist Kapitel 4.2. Ein Lasttest misst die Laufzeit zur Aggregation von Punktdaten. Dies ist auf dem System der Datenbank bzw. des Masters über den gängigen Weg, vorrangig SQL, durchzuführen. Die Laufzeitmessung der Anfragen erfolgt über diese mit Parametern der Anfrage. Analog ist ein weiterer Lasttest zur Interpolation bzw. Kriging durchzuführen.

4.2 Ist-Stand

Die Durchführung einer Softwareauswahl zum teilweisen Ersatz eines bestehenden Systems setzt die Analyse des Systems voraus. In diesem Unterkapitel wird der Ist-Stand sowie der der angestrebte Zustand nach Einbau des Prototypen dargestellt. Firmeninterne Schnittstellen mit dem ist-Stand werden nicht konkretisiert, da sie den Anwendungsfall nicht schneiden.

In Abbildung 4.1 ist die Übersicht des Aufbaus ersichtlich. Das Herzstück bildet die objektrelationale Datenbank PostgreSQL mit der geografischen Erweiterung PostGIS. Diese dient zur Datenhaltung und wesentlich auch zur Datenverarbeitung. In den Programmiersprachen Delphi und R werden zusätzlich automatische Verarbeitungsvorgänge durchgeführt. Daten werden von extern und intern eingespeist. Dabei handelt es sich um Punkte, Vektoren und Raster mit dazugehörigen Metadaten. Im Rahmen der Datenverarbeitung werden Punktdaten interpoliert und mit Hilfe von Geostatistik Auswertungen aus originalen und verarbeiteten Daten erstellt. Als zentrales Element enthält die Datenbank neben den agrartechnischen Kennzahlen alle weiteren Daten des Unternehmens. In dieser Betrachtung werden einzig die agrartechnischen Kennzahlen berücksichtigt.

Es existiert eine Vielzahl von Vorgängen, welche zur Erhöhung der Useability in Hinsicht auf ihre Laufzeit verbessert werden können. Für die Bewertung der Frameworks und schließlich zum Entwurf des Prototypen, ist es notwendig eine Auswahl der zu untersuchenden Vorgängen zu treffen. Die Auswahl erfolgt durch Mitarbeiter der Agri Con und hat Vorgänge mit der längsten Laufzeit als Ergebnis. Dazu zählt das Laden von

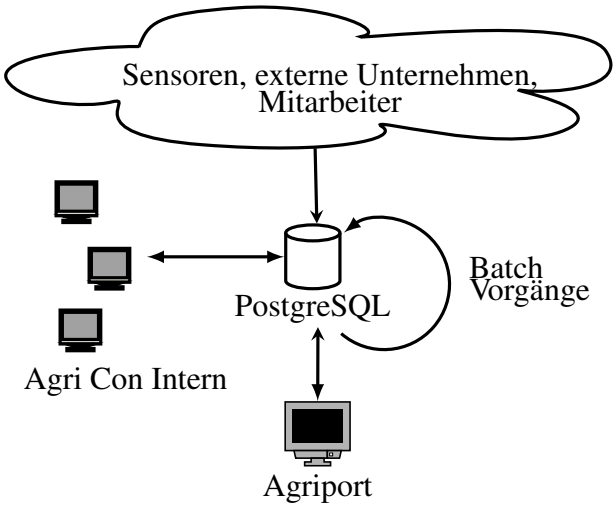

Abbildung 4.1: Aktuelle Infrastruktur bei Agri Con

Daten zum Zwecke der Verarbeitung und Anzeige. Die größten Datenmengen sind bei Punkten aus dem N-Sensor Bereich zu finden. Die Anzeige der Punktdaten für einen Betrieb kann bis zu neun Sekunden dauern. Punktdaten aus dem Docu Bereich werden ebenfalls nach mehrere Sekunden ausgegeben. Die Punktdaten aus dem N-Sensor Bereich sind jedoch repräsentativ und werden deshalb betrachtet. Weiterhin zählt Interpolation zu diesen Vorgängen. Punkte und Fahrspuren werden mit einem angepassten Kriging interpoliert und als Vektoren und Raster abgespeichert. Dies wird mit einer R Bibliothek realisiert und bei großen Datenmengen Nachts angestoßen. Das spezielle Kriging im jeweiligen Framework zu implmentieren ist aufwendig, weshalb ein unveränderter Kriging Algorithmus für den Vergleich verwendet werden muss. Diese zwei charakteristischen Vorgänge sind durch ein Framework zu realisieren.

Das Ziel ist es, ein Framework zusätzlich in den Ist-Stand zu integrieren. Es soll dabei die aufwendigen Vorgänge durchführen, als Permanentspeicher für historische Daten dienen und Daten zur späteren Anzeige aufbereiten und bereitstellen. Die aufwendigen Vorgänge werden in den Lasttests untersucht. In welcher konkreten Form das Framework als Speicher für historische und aufbereitete Daten dient, ist abhängig vom Framework. Auf Grund dessen ist dies nach Auswahl des Frameworks im Entwurf des Prototypen zu konkretisieren.

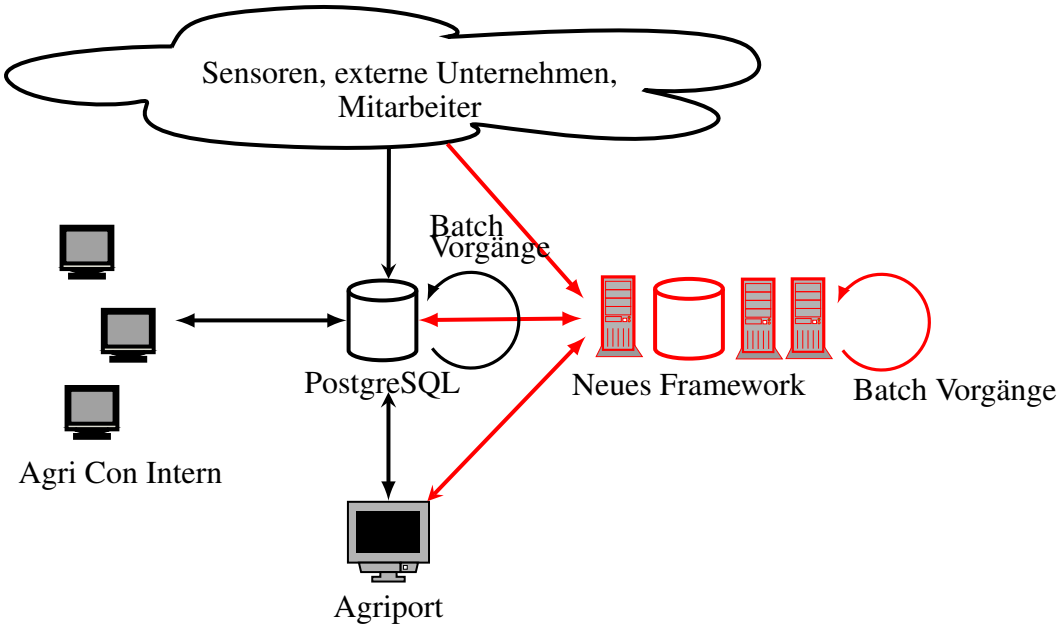

Abbildung 4.2: Übersicht des Aufbaus des Wunsch-Standes

5 Gegenüberstellung

Anhand der unter 4.1.2 erstellten Metriken sind die Frameworks GeoMesa, Postgres-XL und Rasdaman zu vergleichen. Der Vergleich findet im Rahmen einer Nutzwertanalyse statt. Hierbei werden keine Daten von durchgeführten Tests herangezogen, sondern es wird anhand der Spezifikation der einzelnen Frameworks untersucht.
Die drei Frameworks wurden aus der Tabelle der Abbildung 5.1 ausgewählt. Darin sind GIS zur räumlichen Datenverarbeitung mit wesentlichen Eigenschaften wie PostgreSQL Schnittstelle und räumliche Datentypen aufgelistet. Entsprechend den Anforderungen wurden daraus drei Frameworks für die Nutzwertanalyse ausgewählt.
Abbildung 5.1 stammt von der Wikipedia Seite `https://en.wikipedia.org/wiki/Spatial_database` und ist wie unter 3 beschrieben für Unternehmen interessant. Der Autor erschuf die abgebildete Tabelle durch Recherche und stellte sie am 1.2.2015 in den Artikel. In der Annahme, dass unternehmensbezogene Besucher der Seite fehlendes ergänzen oder falsches korrigieren würden, dient diese zur Auswahl geeigneter Frameworks.

Tabelle 5.1 zeigt die für die Nutzwertanalyse notwendige Wertung der einzelnen Metriken. Die Metriken Richtigkeit, Fehlertoleranz und Zeitverhalten werden nicht in die

Metrik	**Gewichtung in %**
Interoperabilität	30
Funktionsumfang	20
Dokumentation	35
Modifizierbarkeit	15

Tabelle 5.1: Wertungsmaßstab der einzelnen Metriken

Analyse aufgenommen, da sie über die Spezifikation nicht belegbar sind.

Table of free systems especially for spatial data processing

DBS	License	Distributed	Spatial objects	Spatial functions	PostgreSQL interface	UMN MapServer interface	Documentation	Modifiable	HDFS
AsterixDB	Apache License 2.0	yes	yes (custom)	center, radius, distance, area, intersect and cell	no	no	good in Google Code	own datatypes, functions and indexes	possible
ESRI GIS Tools for Hadoop	Apache License 2.0	yes	yes (own specific API)	yes (union, difference, intersect, clip, cut, buffer, equals, within, contains, crosses, and touches)	no	no	just briefly	forking	yes
GeoMesa (http://www.geomesa.org/)	Apache License 2.0	yes	yes (Simple Features)	yes (JTS)	no (manufacturable with GeoTools)	no	parts of the funcions, a few examples	with Simple Feature Access in Java Virtual Machine and Apache Spark are all kinds of tasks solvable	yes
H2GIS (http://www.h2gis.org/)	GPL 3	no	yes (custom, no raster)	Simple Feature Access and custom functions for H2Network	yes	no	yes (homepage)	SQL	no
Ingres	GPL or proprietary	yes (if extension is installed)	yes (custom, no raster)	Geometry Engine, Open Source (http://trac.osgeo.org/geos/)	no	with MapScript	just briefly	with C and OME	no
Neo4J-spatial (https://github.com/neo4j-contrib/spatial)	GNU affero general public license	no	yes (Simple Features)	yes (contain, cover, covered by, cross, disjoint, intersect, intersect window, overlap, touch, within and within distance)	no	no	just briefly	fork or JTS	no
Postgres-XL (http://www.postgres-xl.org/) with PostGIS	Mozilla public license and GNU general public license	yes	yes (Simple Features and raster)	yes (Simple Feature Access and raster functions)	yes	yes	PostGIS: yes, Postgres-XL: briefly	SQL, in connection with R or Tcl or Python	no
PostgreSQL with PostGIS	GNU General Public License	no	yes (Simple Features and raster)	yes (Simple Feature Access and raster functions)	yes	yes	detailed	SQL, in connection with R	no
Rasdaman	server GPL, client LGPL, enterprise proprietary	yes	just raster	raster manipulation with rasql	yes	with Web Coverage Service or Web Processing Service	detailed wiki	own defined function in enterprise edition	no

Abbildung 5.1: Übersicht relevanter GIS Frameworks nach [AT15] vom 2.2.2015

Für jedes Framework wird eine Nutzwertanalyse durchgeführt und die dazugehörigen Tabellen dazu präsentiert.

Zu jeder Metrik wird der erreichte Wert, die ungewichtete Erfüllung, die gewichtete Erfüllung ein ein Kommentar angegeben. Die ungewichtete Erfüllung bezieht sich auf den maximal zu erreichenden Wert der Metrik, die gewichtete Erfüllung dagegen auf die Erfüllung der Metrik in Bezug auf 5.1. Die Kommentarspalte dient der Darstellung des Erreichens der Mindestanforderungen. Der schlussendliche Nutzwert ergibt sich nach Zangemeister in [Ben13] aus der Summe der Produkte des Teilnutzens des jeweiligen Kriteriums mit der Gewichtung des Kriteriums. Der Teilnutzen ist hier der Prozentuale Anteil der erreichten Punktzahl an der maximalen Punktzahl des Kriteriums. Diese Prozentangabe wird als Wert mit der Gewichtung des Kriteriums multipliziert, woraus sich der Nutzwert für das Kriterium ergibt. Die Summe aller dieser Teilnutzwerte ergibt den Nutzwert des Frameworks für den Anwendungsfall.

5.1 GeoMesa

5.1.1 Interoperabilität

PostgreSQL - 7 Scala kann mit JDBC auf PostgreSQL zugreifen.

UMN MapServer - 0 UMN MapServer bietet Accumulo nicht als Quelle an und GeoMesa keine OGC konformen Dienste wie WMS.

Die Wertung für Interoperabilität ist somit 7 mit einer Erfüllung von 58%.

5.1.2 Funktionsumfang

Parallele Verarbeitung - 2 Verteilte Datenhaltung durch Accumulo auf HDFS und verteiltes sowie paralleles Rechnen mit beispielsweise Spark [Fox14]

Geografische Datentypen - 12 Vollständige Datentypen aus Simple Feature Access. [Fox14]

Umrechnungsfunktionen - 10 Datenverarbeitung direkt in Spark mit GeoTools möglich. [OSG15a]

Gruppierungsfunktionen - 7 Funktionale Verarbeitung mit Scala.

Verschneidungsfunktionen - 3 JTS stellt difference, union und symmetric difference zur Verfügung. [Wik15b]

Overlayfunktionen - 2 JTS stellt relate und overlay zur Verfügung.

Geostatistik - 0 Keine eingebaute Funktionalität.

Filterfunktionen - 10 Räumliche Filterung ist mit GeoTools möglich [Wik15a]

Schemaversionierung - 0 Accumulo erlaubt entsprechend des BigTable Ansatzes ein dynamisches Datenbankschema, jedoch ohne Versionierung. Einzig erzeugte Datentypen, bestehend aus Simple Features, können in GeoMesa als Konstrukt persistiert werden.

Daraus ergibt sich ein Wert von 48, was 79% des maximal zu erreichenden Wertes ist.

5.1.3 Dokumentation

Installation - 1 Knappe Hinweise für GeoMesa auf [Eic14], dagegen ausführliche Anleitungen für Accumulo auf [Apa14b].

Zeitverhalten - 0 Keine Dokumentation vorhanden.

Funktionsumfang - 1 Konkrete Funktionalität von GeoMesa nur grob auf [Loc14a] angedeutet. MapReduce mit Accumulo ist ausführlich beschrieben. [Apa14b]

Interoperabilität - 1 Nicht explizit bei GeoMesa angegeben, aber Anbindungsmöglichkeiten mit Scala bzw. Java sind im allgemeinen ausführlich dokumentiert.

Best Practise - 0 Keine Dokumentation vorhanden.

Anpassbarkeit - 1 In [Loc14a] sind einige Anregungen zu finden. Beispielsweise die Erzeugung eigener Schemabestandteile. [Loc14b]

Das Qualitätskriterium Dokumentation wird für GeoMesa mit dem Wert 4, bzw. der Erfüllung von 31%, belegt.

5.1.4 Modifizierbarkeit

Verwendung eigener Datentypen - 0 Es sind eigene Schemas aber keine Datentypen erstellbar. [Loc14b]

Erstellung eigener Schnittstellen - 1 Indirekt über JDBC und ODBC möglich.

Erstellung eigener Funktionen - 1 Durch verschiedenste Frameworks zur Datenverarbeitung wie Spark beliebige Funktionen erstellbar.

Verwendung der Programmiersprachen Scala oder R - 1 GeoMesa ist in Scala geschrieben und kann mit dieser verwendet werden. R kann über das Tool SparkR verwendet werden.

Anlegen eigener Berechnungsvorgängen zur späteren Abarbeitung - 1 Mit einer Vielzahl von Tools möglich, bspw. Spark, Storm, Pig und Cascading.

Metrik	erreichter Wert	Erfüllung in %	Kommentar	gewichteter Teilnutzen
Interoperabilität	7	58	Implementationen notwendig	17
Funktionsumfang	48	79	meisten Funktionen nur mit Scala verfügbar	16
Dokumentation	4	31	zumeist ist auf Community zurückzugreifen	11
Modifizierbarkeit	4	67	mit Simple Features und Spark mächtige Problemlösungen erstellbar	10

Tabelle 5.2: Nutzwertanalyse GeoMesa

Hier ist die Wertung 4 von 6 Punkten und damit 67%.

Der Nutzwert von GeoMesa ist nach Tabelle 5.2 54.

5.2 Postgres-XL

5.2.1 Interoperabilität

PostgreSQL - 7 PostgreSQL ist Bestandteil von Postgres-XL wobei die Datentypen vollständig verfügbar sind.

UMN MapServer - 5 Mit der Erweiterung Postgis direkt als Quelle für UMN MapServer angebbar. [Min14]

Die Wertung für Interoperabilität ist somit 12 mit einer Erfüllung von 100%.

5.2.2 Funktionsumfang

Parallele Verarbeitung - 2 Verteilte Datenhaltung mit partitioning der Daten und verschränkte parallele Datenverarbeitung mit MPP möglich. [Tra15a]

Geografische Datentypen - 14 Vollständige Datentypen aus Simple Feature Access sowie PostGIS raster. [OSG15b]

Umrechnungsfunktionen - 10 Direkter Funktionsaufruf zur Umrechnung von und in beliebige EPSG Codes. [OSGa]

Gruppierungsfunktionen - 10 SQL in PostgreSQL mit der Erweiterung PostGIS erlaubt beliebige Querys mit geografischen Daten. [OSG15b]

Verschneidungsfunktionen - 3 Funktionsübersicht zeigt intersection, difference und symmetric difference. [OSGb]

Overlayfunktionen - 2 Funktionsübersicht zeigt relation und intersects. [OSGb]

Geostatistik - 2 Interpolation nur von Linie zu Punkt mit PostGIS möglich. Jedoch kann mit R oder C++ beliebige Geostatistik mit vorhandenen und eigenen Funktionen durchgeführt werden.

Filterfunktionen - 10 In SQL mit mehreren Funktionen. [OSGb]

Schemaversionierung - 0 Nicht eingebaut. Mit eigenen Skripten möglich.

Daraus ergibt sich ein Wert von 53, was 87% des maximal zu erreichenden Wertes ist.

5.2.3 Dokumentation

Installation - 1 Knapp auf [Trab] beschrieben.

Zeitverhalten - 0 Keine Angaben.

Funktionsumfang - 2 Es existiert eine Übersicht zur Verwaltung eines Postgres-XL Clusters. Dazu ist die allgemeine Dokumentation zu PostgreSQL und PostGIS verfügbar. [Trac]

Interoperabilität - 2 Verweis auf Dokumentation von PostgreSQL und PostGIS sowie API auf [Traa].

Best Practise - 1 Einige Hinweise auf [Trac] vorhanden.

Anpassbarkeit - 3 [Trad] dokumentiert Erweiterung mit SQL, tcl, Perl und Python.

Das Qualitätskriterium Dokumentation wird für Postgres-XL mit dem Wert 9, bzw. der Erfüllung von 69%, belegt.

5.2.4 Modifizierbarkeit

Verwendung eigener Datentypen - 1 Mit PostgreSQL eigene Datentypen erstellbar.

Erstellung eigener Schnittstellen - 1 Für eigene Programme mit JDBC oder ODBC Daten verwendbar.

Erstellung eigener Funktionen - 1 Ebenso mit SQL möglich.

Verwendung der Programmiersprachen Scala oder R - 1 R kann direkt in SQL Funktionen eingebettet werden. Scala ist mit JDBC verwendbar.

Anlegen eigener Berechnungsvorgängen zur späteren Abarbeitung - 1 Hier sind Trigger und selbstständige Programme mit JDBC Nutzung zu nennen.

Hier ist die Wertung 5 von 6 Punkten und damit 83%.

Metrik	**erreichter Wert**	**Erfüllung in %**	**Kommentar**	**gewichteter Teilnutzen**
Interoperabilität	12	100	analog Ist-Stand	30
Funktionsumfang	53	87	Geostatistik und Versionierung nicht vorhanden	19
Dokumentation	9	69	Dokumentation zu PostGIS sehr gut, zu Postgres-XL grob	24
Modifizierbarkeit	5	83	meist eigenständige Programme notwendig	13

Tabelle 5.3: Nutzwertanalyse Postgres-XL

Aus Tabelle 5.3 ergibt sich ein Nutzwert von .

5.3 Rasdaman

Metrik	erreichter Wert	Erfüllung in %	Kommentar	gewichteter Teilnutzen
Interoperabilität	12	100	UMN MapServer Schnittstelle nur indirekt vorhanden	30
Funktionsumfang	12	20	umfangreiche Rasterverarbeitung möglich	4
Dokumentation	8	62	mehrere Dokumente vorhanden	22
Modifizierbarkeit	3	60	einfache Java und C++ API	9

Tabelle 5.4: Nutzwertanalyse Rasdaman

Aus Tabelle 5.4 ergibt sich ein Nutzwert von 65.

6 System 1

6.1 Aufbau

6.2 Installation

6.3 Datenimport

6.4 Verarbeitung

6.5 Schnittstelle

6.6 Leistungstests

7 Fazit

7.1 Zusammenfassung

7.2 Wertung

7.3 Ausblick

- auch Bezug auf Verarbeitung von ganzen Länderdaten mit dem System(en) - Darstellung als wichtige Komponente: Möglichkeiten und Performanz

Literaturverzeichnis

[AD14] AENDERUNG DNEBRO letzte: *Framework*. https://de.wikipedia.org/wiki/Framework. Version: 11 2014. – abgerufen am 21.11.2014

[Apaa] APACHE, Software F.: *Apache Thrift*. https://thrift.apache.org/. – abgerufen am 21.1.2015

[Apab] APACHE, Software F.: *ZooKeeper Overview*. https://cwiki.apache.org/confluence/display/ZOOKEEPER/ProjectDescription. – abgerufen am 21.1.2015

[Apa14a] APACHE, Software F.: *Apache Accumulo Notable Features*. https://accumulo.apache.org/notable_features.html. Version: 2014. – abgerufen am 21.1.2015

[Apa14b] APACHE, Software F.: *Apache Accumulo User Manual Version 1.6*. https://accumulo.apache.org/1.6/accumulo_user_manual.html. Version: 5 2014. – abgerufen am 13.1.2015

[AT15] AENDERUNG TBOONX letzte: *Spatial database*. https://en.wikipedia.org/wiki/Spatial_database. Version: 2 2015. – abgerufen am 1.2.2015

[Ben13] BENSBERG, Frank: *Nutzwertanalyse*. http://www.enzyklopaedie-der-wirtschaftsinformatik.de/wi-enzyklopaedie/lexikon/is-management/Management-von-Anwendungssystemen/Beschaffung-von-Anwendungssoftware/Nutzwertanalyse. Version: 8 2013. – abgerufen am 11.1.2015

[boo98] *Lehrbuch der Software-Technik. Software-Management, Software-Qualitätssicherung, Unternehmensmodellierung.*. Bd. Bd. 2. Heidelberg : Spektrum Akad. Verl., 1998. – XX, 769 S. http://www.bsz-bw.de/cgi-bin/ekz.cgi?SWB06262739. – ISBN 3827400651

[Bor07] BORTHAKUR, Dhruba: The Hadoop Distributed File System, Architecture and Design / The Apache Software Foundation. 2007. – Forschungsbericht

[Edl11] EDLICH, Stefan (Hrsg.): *NoSQL : Einstieg in die Welt nichtrelationaler Web 2.0 Datenbanken*. 2., aktualisierte und erw. Aufl. München : Hanser, 2011. – XVIII, 392 S. http://deposit.d-nb.de/cgi-bin/dokserv?id=3791928&prov=M&dok%5Fvar=1&dok%5Fext=htm. – ISBN 9783446427532

[Eic14] EICHELBERGER, Chris: *GeoMesa Quick Start*. http://www.geomesa.org/2014/05/28/geomesa-quickstart/. Version: 5 2014. – abgerufen am 2.12.2014

[ESR14a] ESRI: *geometry-api-java*. https://github.com/Esri/geometry-api-java. Version: 9 2014. – abgerufen am 21.1.2015

[ESR14b] ESRI: *GIS Tools for Hadoop*. 2014. – abgerufen am 21.1.2015

[Eva09] EVANS, Eric: *NOSQL 2009*. http://blog.sym-link.com/2009/05/12/nosql_2009.html. Version: 9 2009

[FC06] FAY CHANG, Sanjay Ghemawat Wilson C. Hsieh Deborah A. Wallach Mike Burrows Tushar Chandra Andrew Fikes Robert E. G. Jeffrey Dean D. Jeffrey Dean: Bigtable: A Distributed Storage System for Structured Data / Google, Inc. 2006. – Forschungsbericht

[Fox14] FOX, Anthony: *GeoMesa: Scaling up Geospatial Analysis*. http://www.eclipse.org/community/eclipse_newsletter/2014/march/article3.php. Version: 3 2014

[Geo15] GEOTOOLS: *GeoTools*. http://docs.geotools.org/latest/userguide/geotools.html. Version: 2015. – abgerufen am 21.1.2015

[Han95] HANSEN, Olav: *Leistungsanalyse paralleler Programme*. Heidelberg [u.a.] : Spektrum, 1995. – 200 S.. – ISBN 3860257072

[Hoh96] HOHENSTEIN, Uwe (Hrsg.): *Objektorientierte Datenbanksysteme : ODMG-Standard, Produkte, Systembewertung, Benchmarks, Tuning*. Braunschweig : Vieweg, 1996. – VII, 269 S.. – ISBN 3528055014

[Hä13] HÄBERLEIN, Dan: *Migration und Extrahierung von Datensätzen mittels spaltenorientierten Datenbanken am Beispiel von Apache HBase*, Universität Leipzig, Bachelor Thesis, September 2013. – Wirtschaftswissenschaftliche Fakultät

[JD04] JEFFREY DEAN, Sanjay G.: MapReduce, Simplified Data Processing on Large Clusters / Google, Inc. 2004. – Forschungsbericht

[Jun13] JUNGHANNS, Kurt: *NOSQL*. 5 2013. – Präsentationsfolien zum Oberseminar Datenbanken - aktuelle Trends

[Kud07] KUDRASS, Thomas (Hrsg.): *Taschenbuch Datenbanken : Mit ... 28 Tabellen*. München : Fachbuchverl. Leipzig im Carl-Hanser-Verl., 2007. – 582 S. http://swbplus.bsz-bw.de/bsz260569755inh.htm. – ISBN 3446409440

[Lan13] LANGE, Norbert d.: *Geoinformatik in Theorie und Praxis*. 3. Springer, 2013

[Liv14] LIVE, OSGeo: *Rasdaman*. http://live.osgeo.org/de/overview/rasdaman_overview.html. Version: 2014

[Loc14a] LOCATIONTECH: *Tutorials*. http://www.geomesa.org/tutorials/. Version: 12 2014. – abgerufen am 7.12.2014

[Loc14b] LOCATIONTECH: *Tutorials*. http://www.geomesa.org/2014/10/09/geomesa-tools-features/. Version: 10 2014. – abgerufen am 7.12.2014

[Lud07] LUDEWIG, Jochen ; LICHTER, Horst (Hrsg.): *Software-Engineering : Grundlagen, Menschen, Prozesse, Techniken*. 1. Aufl. Heidelberg : dpunkt-Verl., 2007. – XXI, 618 S. http://bvbr.bib-bvb.de:8991/F?func=service&doc%5Flibrary=BVB01&doc%5Fnumber=

012989845&line%5Fnumber=0002&func%5Fcode=DB%5FRECORDS&service%5Ftype=MEDIA. – ISBN 3898642682

[Min14] MINNESOTA, University of: *LAYER*. http://mapserver.org/mapfile/layer.html. Version: 2014. – abgerufen am 17.2.2015

[OSGa] OSGEO: *UpdateGeometrySRID*. http://postgis.net/docs/manual-2.1/UpdateGeometrySRID.html. – abgerufen am 12.2.2015

[OSGb] OSGEO: *Using OpenGIS Standards*. http://postgis.net/docs/manual-2.1/reference.html. – abgerufen am 8.2.2015

[OSG15a] OSGEO: *Geometry CRS Tutorial*. http://docs.geotools.org/latest/tutorials/geometry/geometrycrs.html#coordinate-reference-system. Version: 2 2015. – abgerufen am 12.2.2015

[OSG15b] OSGEO: *Using OpenGIS Standards*. http://postgis.net/docs/manual-2.1/using_postgis_dbmanagement.html. Version: 2 2015. – abgerufen am 8.2.2015

[Ras14] RASDAMAN: *Welcome to rasdaman – the World's Leading Array Database*. http://www.rasdaman.org/. Version: 2014

[Traa] TRANSLATTICE: *Client Interfaces*. http://files.postgres-xl.org/documentation/client-interfaces.html. – abgerufen am 8.2.2015

[Trab] TRANSLATTICE: *Installation from Source Code*. http://files.postgres-xl.org/documentation/installation.html. – abgerufen am 8.2.2015

[Trac] TRANSLATTICE: *Postgres-XL 9.2 Documentation*. http://files.postgres-xl.org/documentation/. – abgerufen am 8.2.2015

[Trad] TRANSLATTICE: *Server Programming*. http://files.postgres-xl.org/documentation/server-programming.html,note=

[Tra15a] TRANSLATTICE: *Overview*. http://www.postgres-xl.org/overview/. Version: 2 2015. – abgerufen am 8.2.2015

[Tra15b] TRANSLATTICE: *What is Postgres-XL?* http://files.postgres-xl.org/documentation/intro-whatis.html. Version: 2 2015. – abgerufen am 8.2.2015

[Uni14] UNIVERSE, Parallel: *Overview*. http://docs.paralleluniverse.co/spacebase/. Version: 2014

[Wik15a] WIKIPEDIA: *GeoTools*. https://en.wikipedia.org/wiki/GeoTools. Version: 2 2015. – abgerufen am 2.2.2015

[Wik15b] WIKIPEDIA: *Java Topology Suite*. https://en.wikipedia.org/wiki/JTS_Topology_Suite. Version: 2 2015. – abgerufen am 6.2.2015

[Wik15c] WIKIPEDIA: *Wikipedia article traffic statistics*. http://stats.grok.se/en/latest90/spatial%20database. Version: 2 2015. – abgerufen am 1.2.2015

Eidesstatliche Erklärung

Ich versichere, dass die Masterarbeit mit dem Titel „Untersuchung und Optimierung verteilter Geografischer Informationssysteme zur Verarbeitung Agrartechnischer Kennzahlen“ nicht anderweitig als Prüfungsleistung verwendet wurde und diese Masterarbeit noch nicht veröffentlicht worden ist. Die hier vorgelegte Masterarbeit habe ich selbstständig und ohne fremde Hilfe abgefasst. Ich habe keine anderen Quellen und Hilfsmittel als die angegebenen benutzt. Diesen Werken wörtlich oder sinngemäß entnommene Stellen habe ich als solche gekennzeichnet.

Leipzig, 17. Februar 2015 Unterschrift