

Universität Leipzig  
Fakultät für Mathematik und Informatik  
Wintersemester 2021/22  
Bachelorarbeit

---

# **Transformation von Daten des Amtlichen Liegenschaftskatasterinfor- mationssystem in semantische Datenformate**

---

Betreuer: Prof. Dr. Hans Gert Gräbe, Dr. Christian Zinke-Wehlmann, Norman Radke

Abgabedatum: .03.2022

Robin Seidel

# Inhaltsverzeichnis

1	Einleitung . . . . .	1
1.1	Themenbeschreibung/Zusammenfassung . . . . .	1
1.2	Motivation . . . . .	1
2	Grundlagen . . . . .	3
2.1	ALKIS . . . . .	3
2.2	ALKIS Daten . . . . .	4
2.3	RDF . . . . .	8
2.4	RML & Yarrml . . . . .	8
2.5	SPARQL . . . . .	9
2.6	Tripelstores . . . . .	9
2.7	RDF vs XML . . . . .	9
3	State of the Art . . . . .	11
3.1	Problembeschreibung . . . . .	11
3.2	Möglichkeiten der Transformation . . . . .	11
4	Anforderungsanalyse . . . . .	13
4.1	Allgemeine Anforderungen . . . . .	13
4.2	Spezifische Anforderungen der verschiedenen Anwendungsfälle .	13
5	Implementierung . . . . .	14
5.1	Beschreibung der Implementierung . . . . .	14
5.2	genutzte Technologien . . . . .	17
6	Ergebnisse und Evaluation . . . . .	17
7	Ausblick . . . . .	17
7.1	Notizen . . . . .	18
8	Quellenverzeichnis . . . . .	19

# 1 Einleitung

## 1.1 Themenbeschreibung/Zusammenfassung

Die Bachelorarbeit befasst sich mit der Transformation von Daten aus dem Amtlichen Liegenschaftskataster Informationssystem kurz ALKIS in ein semantisches Format. Im Vordergrund dieser Arbeit stehen dabei die Fragen inwiefern ALKIS Daten in semantischen Formaten nützlicher als die vorhandenen Daten im XML Format sind und ob eine vollständige Automatisierung der Transformation der Daten überhaupt möglich ist. Daten über die Liegenschaften in Deutschland werden auf eine fest definierte und strukturierte Art und Weise von den verschiedenen Vermessungsbehörden der Länder gesammelt und veröffentlicht. Die Varianz und Verfügbarkeit der Datensätze bildet dabei ein großes Problem. In dieser Arbeit wird die Umwandlung dieser Daten in das Resource Description Framework(RDF) betrachtet. Im Anschluss auf die Transformation werden die Vorteile der Daten im RDF Format betrachtet. These/Forschungsfrage erläutern..

## 1.2 Motivation

Mit der Einführung des World Wide Webs im Jahre 1990 durch Tim Burner Lee ist das Internet heute viel mehr als das, was es früher einmal war. Der Mensch nutzt das Internet immer häufiger als Hauptinformations- und Interaktionsmöglichkeit über diverse Unterhaltungskanäle wie Facebook, Youtube oder andere Websites. (QUELLE) Während am Anfang lediglich ein paar Dutzend Webseiten existierten sind es mittlerweile über zwei Milliarden mit 100 Milliarden Webdokumenten, die sich wiederum alle 6 Monate verdoppeln. (vgl. Meinel 2021) Aber es entstehen nicht immer nur mehr Webseiten auch die Anzahl der Internetnutzer hat sich in den letzten Jahren drastisch gesteigert. 1995 hat der Spiegel 250.000 Nutzer in Deutschland geschätzt, während 2018 durch das statistische Bundesamt ein Anstieg auf 90 wurde. (vgl. Köllinger 2003 zitiert nach: Der Tagesspiegel vom 19. August 1995, S. 23) Demnach nutzen mittlerweile 66,5 Millionen Menschen ab 10 Jahren das Internet als Informations- und Interaktionsquelle. (vgl. Statistisches Bundesamt 2018) Bei einer solchen Masse an Informationen und Daten wird es immer schwerer relevante Informationen herauszufiltern, obwohl das Hauptziel des Internets einst nicht die Unterhaltung, sondern die Informationsbeschaffung war. (QUELLE) Deshalb ist es wichtig die vorhandenen Daten mit Hilfe von RDF Graphen in einer geordneten Art und Weise zur Verfügung zu stellen. Das semantische Web war bereits eine Idee des Entwicklers/Gründers Tim Burner Lee. Er hatte die Vision, dass das Netz von Informationen nicht nur Dokumente und Daten miteinander verbindet, sondern das diesen Daten auch eine Bedeutung gegeben wird. Damit ermöglicht man einer Maschine die Anhäufung von Daten automatisiert nutzen zu können. - beschreibung semantik web und wichtigste Prinzipien - 4 prinzipien: tim berner lee - nutzen von URI's als namen

für Objekte - zur eindeutigen Identifikation für Dinge - nutze HTTP URIs - so können Menschen die Objekte nachschauen - nutze anerkannte Standards (RDF, Sparql) - linke URIs weiter. Das semantische Web ist eine Art rohe (also unearbeitete) Daten zu übertragen. (Interoperabilität) Um die Informationen, die das Web liefert, in einen strukturierten Weg wiederzugeben. Mit Hilfe der daraus entstehenden strukturierten Daten kann die Software diese Daten anschließend besser nutzen und verarbeiten. Das semantische Web ist damit sozusagen eine Alternative zu Webseiten, die ausschließlich für die menschliche Nutzung geeignet sind. Die Vorteile, die eine Semantifizierung der Daten mit sich bringt, sind unter anderem das Abfragen durch Sparql Querys, eine Verbesserung der Bedeutung der Daten, das Verbinden von Daten und die Nutzbarkeit für andere Bereiche. - Überleitung auf Alkis Thematik: Alkis Daten (siehe Begriffsklärungen) werden zwar in gewissen Standard und Einheitlichkeit erhoben, allerdings ist die Verarbeitung dieser Daten hauptsächlich für GeoProgramme (näher erklären) vorgesehen. Die Art der Daten ist nicht für die optimale Verarbeitung von ComputerProgrammen und zur Nutzung für den gewöhnlichen Nutzer vorgesehen. Eine Überführung in Rdf Graphen ermöglicht es, das Potenzial der Daten freizuschalten und die Nutzung für zahlreiche andere Bereiche zur Verfügung zu stellen. Quellen: (<https://www.spektrum.de/kolumne/semantic-web-wie-das-internet-inhalte-verstehen-koennte/> 1841389) Paper über Tim Berners-Lee's Arbeit [https://www.researchgate.net/publication/307845029\\_Tim\\_Berners-Lee's\\_Semantic\\_Web](https://www.researchgate.net/publication/307845029_Tim_Berners-Lee's_Semantic_Web) Paper von Burner über das Semantic Web 10.3.2022 <https://www.w3.org/2000/Talks/0906-xmlweb-tbl/text.htm> Statistik Bundesamt [https://www.destatis.de/DE/Presse/Pressemitteilungen/2018/09/PD18\\_330\\_634.html](https://www.destatis.de/DE/Presse/Pressemitteilungen/2018/09/PD18_330_634.html) Köllinger, Philipp <https://www.econstor.eu/handle/10419/151237> (Verlag und Jahr noch angeben)

## 2 Grundlagen

Hier werden die wichtigsten Konzepte erklärt, die in dieser Arbeit Anwendung gefunden haben.

### 2.1 ALKIS

Das Amtliche Liegenschaftskatasterinformationssystem, kurz ALKIS, ist ein Teil des AAA-Modells (AFIS-ALKIS-ATKIS Modell). AFIS (Amtliches Festpunktinformationssystem) ist dabei ein Raumbezugssystem, indem Informationen zu Festpunkten modelliert werden. In ATKIS (Amtliche Topographisch-Kartographische Informationssystem) geht es um digitale Karten zu Landschaften und Topografie. (seite 8 geoinfodoc). ALKIS umfasst die für uns relevanten Informationen aus dem Liegenschaftsbuch und -karte sowie Punktnachweisen. Die Verwaltung des AAA- Modells erfolgt durch die ADV (Arbeitsgemeinschaft der Vermessungsverwaltungen der Länder der Bundesrepublik Deutschland). Das Modell ist seit 2015 in allen Bundesländern aktiv. Ziel der Entwicklung eines solchen Modells war eine einheitliche und fachübergreifende Nutzung von Geodaten zu ermöglichen.(seite 2 geoinfodoc). Bei der Erschaffung des Datenmodells wurden Standards und Normen verwendet die vom W3C(World Wide Web Consortium), OGC(Open Geospatial Consortium) und von der ISO(International Organization for Standardization) anerkannt sind. WELCHE STANDARDS UND NORMEN Durch die Einführung von ALKIS wurden alle Daten des Liegenschaftskatasters redundanzfrei zusammengeführt. (quelle s 10) Kataster- und Liegenschaftsinformationen sind Geobasisdaten über sogenannte Liegenschaften, also zum Beispiel Flurstücke oder Gebäude. Diese Geobasisdaten werden durch die Vermessungs- und Katasterverwaltungen der Länder erhoben. Die Nachteile, die die Führung eines Liegenschaftsbuches mit sich bringen, sind deutlich: teilweise technisch veraltete Konzepte, verschiedene Datenformate, getrennte und teilweise redundante Datenhaltung und länderspezifische Unterschiede treffen hier aufeinander. (Quelle s8 13.3) Diese Nachteile konnten durch die Einführung von ALKIS gelöst werden.

<https://www.adv-online.de/AdV-Produkte/Liegenschaftskataster/ALKIS/broker.jsp?uMen=e5670f15-8e71-3c01-e1f3-351ec0023010>  
<https://www.adv-online.de/AdV-Produkte/Liegenschaftskataster/ALKIS/>

11.3.22 ADV Online - ALkis Beschreibung

## 2.2 ALKIS Daten

Die Veröffentlichung der ALKIS Daten erfolgt über Schnittstelle,n die von den jeweiligen Bundesländern angeboten werden. Einige Bundesländer haben jedoch gewisse Zugriffsbeschränkungen. 7 Bundesländer haben zumindestens einige Geodatenätze frei zugänglich gemacht. Sie unterliegen der Open Data Lizenz: "Datenlizenz Deutschland - Namensnennung- Version2.0"<https://www.govdata.de/dl-de/by-2-0> Andere WFS Schnittstellen sind nur durch Anmeldung und oder mit Kosten verbunden. (siehe Tabelle)(<https://learn.opengeoedu.de/opendata/vorlesung/offene-geodaten/amtliche-geodaten>

Die Bestandsdaten aus ALKIS können über eine sogenannte normbasierte Austausch-schnittstelle(NAS) abgerufen werden. Diese wurde speziell für den Austausch von Daten des AAA - Modells vom ADV entwickelt. Als Standards werden unter anderem XML, WFS (Web Feature Server) oder auch WMS (Web Map Server) genutzt. XML wird dabei als Datenaustauschformat verwendet, während WFS und WMS zum Abrufen der ALKIS-Objekte dienen. Diese vom OGC (Open Geospatial Consortium Inc) definierten Geodatendienste sind ein Teil der NAS Spezifikation. Geodatendienste sind standardisierte Services die über eine URL aufgerufen und zum Austausch von Daten genutzt werden. [https://geodatenonline.bayern.de/geodatenonline/seiten/wms\\_webdienste](https://geodatenonline.bayern.de/geodatenonline/seiten/wms_webdienste)

3.2022 Die Hauptaufgabe eines WMS ist die Visualisierung der Geodaten, er ist ein Darstellungsdienst. Das Ergebnis (Response) einer Anfrage (Request) auf einen WMS ist ein Kartenausschnitt. Der WFS ist ein Download Service, dem Rohdaten in Form der vom ADV definierten Objekte zugrunde liegen. Beim WFS bleibt die volle Struktur der Daten erhalten und sie werden im vollem Ausmaß bereit gestellt und sind somit optimal zur Weiterverarbeitung geeignet. Um an die Daten zukommen die für das Projekt benötigt werden, wird der WFS genutzt. Diese Schnittstelle besitzt drei verschiedene Abfrageoptionen. Die Basis URL ist für das Beispiel NRW: [https://www.wfs.nrw.de/geobasis/wfs\\_nw\\_alkis\\_vereinfacht?](https://www.wfs.nrw.de/geobasis/wfs_nw_alkis_vereinfacht?). An diese URL können nun verschiedene Parameter angehängen werden, um die richtige Antwort (Response) des Servers zu erhalten. Der WFS hat verschiedene Abfrageoptionen. Mit Request=GetCapabilities"werden alle Fähigkeiten und Metadaten des aufgerufenen Dienstes abgerufen. Durch Request=DescribeFeatureType die Struktur eines FeatureTypes abgefragt. Die wichtigste Abfrageoption ist Request=GetFeature". Eine komplette Anfrage (Request) an den WFS könnte so aussehen: [https://www.wfs.nrw.de/geobasis/wfs\\_nw\\_alkis\\_vereinfachtService=WFS&REQUEST=GetFeature&VERSION=2.0.0&Typenames=Flurstueck&Countasdfasdfasdf=10](https://www.wfs.nrw.de/geobasis/wfs_nw_alkis_vereinfachtService=WFS&REQUEST=GetFeature&VERSION=2.0.0&Typenames=Flurstueck&Countasdfasdfasdfasdf=10). Die Anfrage ist ein HTTP Get Call und beinhaltet neben der Basis URL verschiedene Parameter die erforderlich (Beispiel: Version, Typenames) oder optional (Beispiel:Count, Resulttype) sein können. Da der WFS ein Downloaddienst ist, erhält man als Response ein XML Dokument, das die abgefragten ALKIS Objekte enthält. Eine Übersicht über

alle möglichen Objekte findet man im Objektartenkatalog Seite 22. Das für uns wichtigste Objekt ist das AX\_Flurstueck. Laut Definition(Objectartenkatalog seite33) ist ein Flurstück ein Teil der Erdoberfläche, das mit einer Grenzlinie umschlossen ist. Das Objekt AX\_Flurstueck besitzt die Attribute zustaendigeStelle, gemarkung, flurstuecksnummer, flurstueckskennzeichen, amtlicheFlaeche, flurnummer, und die Relationen istGebucht, zeigtAuf, weistAuf, gehoertAnteiligZu, beziehtSichAufFlurstueck etc.. Die Veröffentlichung der ALKIS Daten erfolgt über Schnittstellen die von den jeweiligen Bundesländern angeboten werden. Einige Bundesländer haben jedoch gewisse Zugriffsbeschränkungen. 7 Bundesländer haben zu mindestens einige Geodatenätze frei zugänglich gemacht. Sie unterliegen der Open Data Lizenz: "Datenlizenz Deutschland - Namensnennung-Version2.0"<https://www.govdata.de/dl-de/by-2-0> Andere WFS Schnittstellen sind nur durch Anmeldung und oder mit Kosten verbunden. (siehe Tabelle)(<https://learn.opengeoedu.de/opendata/vorlesgeodaten/>

<https://www.adv-online.de/icc/extdeu/binarywriterservlet?imgUId=8f830072-8de8-9221-d5ad-8f138a438ad1&uBasVariant=11111111-11111111-11111111-11111111-11111111-11111111-11111111-11111111>  
geoInfoDoc

[https://www.lgln.niedersachsen.de/download/126790/Basiswissen\\_ALKIS\\_ETRS89\\_Schulungsmaterial\\_Stand\\_12.04.2010.pdf](https://www.lgln.niedersachsen.de/download/126790/Basiswissen_ALKIS_ETRS89_Schulungsmaterial_Stand_12.04.2010.pdf) bAsiswissen  
alkis 10.3

[https://www.bezreg-koeln.nrw.de/brk\\_internet/geobasis/webdienste/anleitung\\_wfs.pdf](https://www.bezreg-koeln.nrw.de/brk_internet/geobasis/webdienste/anleitung_wfs.pdf)

Bundesland	verfügbare Datensätze	Zugriffsbeschr.
NRW	AAA, NAS, vereinfacht	Quellennachweiß: Geobasis NRW, dl-de/by-2-0, (Daten geändert)
Berlin	AAA(funktioniert nicht, featurtype nicht gefunden)	Quellennachweiß: Geoportal Berlin, dl-de/by-2-0, (Daten geändert)
Thüringen	keine Angabe	Authentifizierung und Quellennachweiß: GDI-TH, dl-de/by-2-0, (Daten geändert)
Brandenburg	AAA, NAS, vereinfacht	Quellennachweiß: GeoBasis-DE/LGB, dl-de/by-2-0, (Daten geändert)
Mecklenburg-Vorpommern	vereinfacht	Kostenpflichtig, Quellennachweiß: GeoBasis-DE/M-V 2022 (Daten geändert)
Hamburg	vereinfacht	keine
Sachsen	vereinfacht	Quellennachweißes: GeoSN, dl-de/by-2-0, (Daten geändert)
Sachsen-Anhalt	NAS-konform, vereinfacht	kostenpflichtig
Bayern	vereinfacht	Authentifizierung /Freischaltung notwendig
Niedersachsen	NAS-konform, AAA-basiert, vereinfacht	kostenpflichtig
Hessen	vereinfacht, NAS-konform	keine
Rheinland-Pfalz	vereinfacht	Freischaltung notwendig, kostenpflichtig
Schleswig-Holstein	keine Angabe	Freischaltung notwendig, kostenpflichtig
Baden-Württemberg	keine Angabe	auth notwendig, Quellennachweiß: LGL, www.lgl-bw.de
Bremen	keine Angabe	Authentifizierung /Freischaltung notwendig
Saarland	keine Angabe	Freischaltung notwendig, kostenpflichtig



\*Die Endpoints für die einzelnen Bundesländer sind im Anhang zu finden.

## 2.3 RDF

Das Resource Description Framework kurz RDF, ist ein vom W3C konzipiertes Datenformat zum Beschreiben von Objekten im Internet. <https://www.w3.org/TR/rdf-concepts>

RDF ist ein universelles, maschinenlesbares Format, das für den Austausch und der Darstellung von Daten gedacht ist. Ursprünglich wurde es genutzt um Metadaten zu beschreiben. (Beleg) Später wurde es auch in vielen verschiedenen Gebieten (WELCHE?) und generellen Anwendungen benutzt. Heute gilt es als grundlegender Baustein des Semantischen Webs. Die Daten, die im RDF Format vorliegen, sind als Graph angeordnet, welcher Kanten und Knoten besitzt. Im Grunde ist ein RDF Graph eine Ansammlung von Triples, also Aussagen über Dinge. Ein Triple besteht aus einem Subjekt, was konkret eine URI oder eine blank note sein kann, einem Prädikat was in dem Triple die Eigenschaft widerspiegelt und aus einem Objekt. Ein Beispiel wäre:

```
<http://example.de/Author> <http://example.de/relationship/hatName> <http://example.de/name/Robin> .
```

In diesem Beispiel ist

<http://example.de/Author> das Subjekt,

<http://example.de/relationship/hatName> das Prädikat und

<http://example.de/name/Robin> das Objekt.

RDF ist ein Datenmodell was viele verschiedene Notationen bzw. Serialisierungen besitzt. Die ursprüngliche Notation ist RDF/XML, welche sich durch seine Bekanntheit und Austauschbarkeit auszeichnet.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:gndo="http://d-nb.info/standards/elementset/gnd#"
  <rdf:Description rdf:about="http://d-nb.info/gnd/118540238">
    <gndo:placeOfBirth rdf:resource="http://d-nb.info/gnd/4018118-2"/>
  </rdf:Description>
</rdf:RDF>
```

Im Vergleich zu Turtle ist sie jedoch eher schlecht durch den Menschen lesbar. Das ist der Vorteil von Turtle. Diese Schreibweise zeichnet sich dadurch aus, dass wenig Speicherplatz benötigt wird und sie für den Menschen intuitiv lesbar ist. URIs werden in eckigen Klammer geschrieben und Literale in Anführungszeichen. Die Triples sind durch einen Punkt getrennt.

```
@prefix gndo: <http://d-nb.info/standards/elementset/gnd#> .
```

```
<http://d-nb.info/gnd/118540238>
  gndo:placeOfBirth <http://d-nb.info/gnd/4018118-2> .
```

## 2.4 RML & Yarrml

Die RDF Mapping Language (RML) <https://rml.io/specs/rml/> ist eine Sprache mit der man Regeln aufstellt, um Datenstrukturen die in verschiedenen Formaten vor-

liegen in RDF umzuwandeln. RML beruht und erweitert die vom W3C definierte(?) Sprache R2RML <https://www.w3.org/TR/r2rml/> die dafür genutzt wird ein spezielles Mapping anzufertigen, um Daten aus einer relationalen Datenbank in ein RDF Datenset umzuwandeln. Sowohl R2RML Mappings als auch RML Mappings sind RDF Graphen und werden in Turtle geschrieben. Mit RML ist es möglich Daten aus heterogenen Datenquellen auf das RDF Format zu mappen. RML ist kein offiziell bestätigter Standard, es erweitert nur R2RML. Das bedeutet, dass RML nicht nur auf relationale Datenbanken, sondern auch auf weitere Datenstrukturen angewendet werden kann. Da in dieser Arbeit Daten in das XML Format umgewandelt werden, ist die Verwendung von RML Regeln zu bevorzugen. (REGELN?) Yarrml <https://rml.io/yarrml/spec/> ist wiederum eine Sprache, die auf Yaml <https://yaml.org/spec/1.2.2/> beruht. Sie ermöglicht es auf einfache Art und Weise deklarative Regeln für das Mapping von verschiedenen Datenquellen auf RDF anzuwenden. Yarrml hat den gleichen Zweck wie RML, mit dem Vorteil, dass es für den Nutzer einfacher ist die Mapping Regeln zu erstellen, wird somit die Effektivität der RDF Graphen Erstellung erhöht. (DEN ABSATZ NOCH MAL)

## 2.5 SPARQL

Erklärung von SPARQL eventuell noch hier

## 2.6 Tripelstores

Erklärung von Triplestores eventuell noch hier

## 2.7 RDF vs XML

Bei der Recherche dieser Arbeit ist dabei eine zentrale Frage entstanden. Warum sollte man überhaupt RDF nutzen, wenn dies ohnehin auf XML beruht und die ALKIS Daten bereits in diesem universellem Format vorliegen? Was ist der Vorteil von RDF gegenüber anderen Datenaustauschformaten? Für die Beantwortung der Frage wird auf das oben genannte Beispiel (SEITE) zurückgegriffen.

Eine Aussage kann in XML durch verschiedene Arten ausgedrückt werden. Für den Nutzer der diese Aussagen liest bedeuten Sie das Gleiche. Für eine Maschine bzw. einen Computer sind es verschiedene XML Bäume. In RDF wird die Aussage der Autor der Bachelorarbeit ist RobinSSeidel als Triple dargestellt. `triple(Autor, Bachelorarbeit, RobinSeidel)` in XML kann diese Aussage auf verschiedene Art und Weise dargestellt werden (SATZBAU?)

```
<autor>
```

```
<von>bachelorarbeit</von>
```

```
<name>Robin Seidel </name>
</autor>
oder auch:
<bachelorarbeit>
<autor>Robin Seidel </autor>
</bachelorarbeit>
```

Für den Nutzer der diese XML Dokumente liest, haben diese die gleiche Bedeutung. Für eine Maschine bzw. einen Computer sind es jedoch zwei verschiedene XML Bäume. Dies zeigt das die Nutzung von Technologien des Semantischen Webs sehr wertvoll sein können. <https://www.w3.org/DesignIssues/RDF-XML.html> <https://dbs.uni-leipzig.de/html/seminararbeiten/semSS99/arbeit5/Rdf.html>

## 3 State of the Art

### 3.1 Problembeschreibung

Das Ziel dieser Arbeit ist die Umwandlung von ALKIS Daten in ein semantisches Format. Um dieses Ziel zu erreichen findet der ETL-Prozess Anwendung. Der ETL-Prozess(Extract, Transform, Load) ist ein typisches Vorgehen in der Welt von Big-Data. Das Ziel des Prozesses sind Daten aus unterschiedlichen Datenquellen für die weitere Verarbeitung vorzubereiten und anschließend wieder bereitzustellen. Der Extract(auf deutsch extrahieren) Teil des Prozesses ist es die notwendigen ALKIS Daten zu sammeln. Dies gelingt durch HTTP-Requests an die Schnittstellen die von den Behörden der Bundesländer bereit gestellt werden. (siehe Tabelle 2.2) Durch die gezielte Angabe der richtigen Parameter erhält man so die notwendigen ALKIS Rohdaten. Der Teilprozess der Transformation enthält wiederum mehrere Einzelschritte. Ziel der Transformation ist es die Daten in ein semantisches Format so umzuwandeln, dass sie verknüpft sind und durch eine SPARQL Schnittstelle abgerufen werden können. Die ALKIS Daten müssen nun mithilfe von Mapping Regeln so konstruiert werden, dass sie am Ende ein möglichst nützlichen und strukturierten Graphen ergeben. Im Load Teil des Prozesses werden die Daten dann in einen Triple Store geladen. Dieser dient als Datenbank für Graphen. Die transformierten RDF Daten können nun über SPARQL Abfragen abgerufen werden.

<https://www.bigdata-insider.de/was-ist-etl-extract-transform-load-21.3>

### 3.2 Möglichkeiten der Transformation

Im ersten Schritt der Arbeit war es nötig, die möglichen Herangehensweisen an das Projekt und die damit einhergehenden Probleme zu identifizieren. ALKIS Daten werden in vielen verschiedenen Formaten angeboten. Es gibt zum Beispiel Datensätze über Gebäude, Bodenschätzungen und über Flurstücke. Die Art und Weise wie die Daten aufgebaut sind und abgerufen werden ist zwar durch den ADV geregelt, jedoch kommt es in den verschiedenen Bundesländern hinsichtlich der Erreichbarkeit dieser Datensätze noch zu großen Unterschieden. So gibt es beispielsweise Datensätze ohne Zugriffsbeschränkung und Datensätze auf die nur durch Bezahlung zugegriffen werden kann. (näheres siehe 2.2 ALKIS DATEN) Da es das Ziel dieser Arbeit ist, einen Wissensgraphen über ALKIS Daten zu erstellen, wurde der Fokus auf den Datensatz "Flurstuecke" gesetzt. Dieser ist der Datensatz mit den wichtigsten Informationen, er enthält alle wichtigen Daten über die Liegenschaften und Flurstücke der jeweiligen Bundesländer. Eine Möglichkeit um dieses Problem zu bewältigen, wäre die manuelle Erstellung der RDF Dateien. Aufgrund des Ausmaßes der Daten ist diese Möglichkeit allerdings als unrealistisch zu betrachten. Eine weitere Herangehensweise wäre die Entwicklung eines SELBSTENTWICKELTEN

Parser, [ es einen selbstentwickelten Parser zu entwickeln,] der es ermöglicht aus den vorhandenen ALKIS XML Dateien, RDF Dateien zu erzeugen. Vorteile dieser Methode sind, dass der entwickelte Parser sehr gut an die Gegebenheiten der ALKIS Daten angepasst werden könnte und man damit die Transformation nach eigenen Wünschen steuern könnte. Allerdings wäre die Entwicklung eines solchen Programmes zu komplex und kompliziert, dass damit ein erheblicher Mehraufwand für das Projekt einhergehen würde. Eine weitere Herangehensweise, die letztlich auch in dieser Arbeit Anwendung gefunden hat, ist die Verwendung einer bereits existierenden Software, den `rmlmapper`([link hier](#)). Auf der einen Seite bietet es den Vorteil, dass man davon ausgehen kann, dass es für die erwartenden Anwendungsfälle zuverlässig funktioniert. Auf der anderen Seite muss man aber mit den vorhandenen Gegebenheiten arbeiten und sich intensiv damit auseinandersetzen, wie die eventuell verwendete Software funktioniert. Wenn man nun davon ausgeht das zur Erstellung des RDF Graphens der `rmlmapper` verwendet wird, muss man sich die Frage stellen auf welche Art und Weise die RML Regeln entstehen, da der `rmlmapper` diese RML Regeln benötigt um aus verschiedenen Daten verknüpfte Daten zu erstellen. Dafür gibt es zwei Möglichkeiten. Entweder die Erstellung von RML Regeln mithilfe der vom w3c spezifizierten Sprache R2RML <https://www.w3.org/TR/r2rml/> oder mit Yarrml. Wie in Punkt 2.4 beschrieben, ist es mit Yarrml möglich auf einfachere Art und Weise deklarative Mapping Regeln zu erstellen. Diese müssen dann nur noch in R2RML Regeln umgewandelt werden. Vorteil daran ist das Yarrml Regeln leichter zu erstellen sind als R2RML Regeln. Nachteil ist das Yarrml nur in einer inoffiziellen Version vorliegt und die Dokumentation bezüglich der Anwendung eher mangelhaft ist. Bei der Abspeicherung der fertig umgewandelten RDF Graphen gibt es ebenfalls mehrere Möglichkeiten. Zur Speicherung von Graphen werden keine herkömmlichen relationalen Datenbanken genutzt, sondern speziell dafür konzipierte Graphen Datenbanken. Dadurch das Beziehungen/Relationen der Graphen in so einer Datenbank abgespeichert sind müssen diese bei einer Abfrage nicht erst berechnet werden sondern stehen bereits zur Verfügung. Dadurch ergibt sich eine bessere Performance gegenüber relationalen Datenbanken. Da das Angebot von Graphdatenbanken relativ groß ist, muss ein für das Projekt passendes System ausgewählt werden. In Frage kommen hierfür die Open Source Graphdatenbank Neo4j oder das Open-Source-Framework Apache Jena Fuseki. Bei ersterem werden Daten anstatt auf die herkömmliche Art und Weise nicht in Tabellen, sondern in Graphen strukturiert abgespeichert. Die Implementierung in das Projekt ist einfach und die Speicherung der ALKIS Daten in Neo4j funktioniert ohne Probleme schnell und zuverlässig. Jedoch ist es in Neo4j vorgesehen als Abfragesprache Cypher zu verwenden, anstatt der für Graphen in der Branche meist angewandten Sprache SPARQL. Apache Jena Fuseki ist speziell für den Austausch von semantischen Daten entwickelt. Hiermit ist auf simple Art und Weise möglich einen Fuseki SPARQL Server <https://jena.apache.org/documentation/fuseki2/> aufzusetzen

und diesen über eine Programmierschnittstelle abzufragen.

## **4 Anforderungsanalyse**

Zur Umsetzung des Projektes wird ein Programm zur automatisierten Transformation von ALKIS Daten entworfen und implementiert. Die Anforderungen die so eine Software haben sollte werden hier zur besseren Übersicht festgehalten. Das Programm soll dem Nutzer als Tool/Werkzeug dienen, Daten aus ALKIS, speziell den Datensätzen in dem Flurstücke beschrieben werden, in RDF umzuwandeln. Deshalb ist es hilfreich wenn der Nutzer des Programms mithilfe von User Input bestimmen kann, was er tun möchte. Das Programm ist für vier Anwendungsfälle konfiguriert.

### **4.1 Allgemeine Anforderungen**

Bei richtigem Input sollte das Programm den erwarteten Befehl ausführen und Rückmeldung geben, dass der Befehl erwartungsgemäß ausgeführt wurde. Bei fehlerhaftem Input sollte das Programm reagieren und den ausgelösten Fehler anzeigen. Das Programm sollte bei auftretenden Fehlern nicht einfach abstürzen, sondern einen erneuten Anlauf erlauben.

### **4.2 Spezifische Anforderungen der verschiedenen Anwendungsfälle**

Der erste Anwendungsfall, den der Nutzer des Programmes auswählen kann, ist der Download von ALKIS Daten. Dabei sollte der Nutzer die Möglichkeit haben aus einer vorgegebenen Liste, den gewünschten Datensatz auszuwählen. Diese Liste enthält die URL's, der durch die 16 Bundesländer zur Verfügung gestellten Datensätzen in den verschiedenen Schemavarianten (AAA-basiert, vereinfachtes Schema, oder NAS-konform). Weiterhin sollte der Nutzer mit Angabe einer Gemeindenummer oder Flurnummer den Abruf der Daten weiter spezifizieren können. Nach Eingabe der geforderten Daten wird dem Nutzer eine Rückmeldung gegeben ob der Download erfolgreich war. Bei Nichterfolg kann der Nutzer eine erneute Eingabe durchführen.

Die Transformation der Daten bilden den zweiten Anwendungsfall. Der Nutzer wählt aus diversen Vorgaben aus, welcher Datensatz umgewandelt werden soll. Auch hier soll der Nutzer bei einem fehlgeschlagenem Prozess eine Mitteilung erhalten. Eine erneute Eingabe sollte ermöglicht werden.

Der dritte Anwendungsfall ist die Überführung der transformierten Daten in eine Graphdatenbank. Der Nutzer sollte einen Pfad angeben können, um die Datei zu identifizieren, die abgespeichert werden soll.

Der vierte Anwendungsfall ist die Abfrage der Daten. Dem Anwender soll ermöglicht werden, die Daten in Form von Sparql Querys abzufragen. Ihm soll das Ergebnis in

Textform angezeigt werden.

## 5 Implementierung

### 5.1 Beschreibung der Implementierung

Zur Umsetzung der beschriebenen Software Anforderungen, wurde eine Konsolenanwendung prototypisch programmiert. Prototyping ist eine Methodik in der Softwareentwicklung, die zum Ziel hat schnell ein gewisses Ziel zu erreichen. Also die Lösung eines Kernproblems ohne vollständig alle gewünschten Funktionen umzusetzen. Das Kernproblem dieses Projektes ist es einen ALKIS Datensatz zu einem RDF Graphen umzuwandeln. Eventuelle Erweiterungen der Software werden im weiteren Verlauf der Arbeit beschrieben. Bei Programmstart wird der Nutzer aufgefordert auszuwählen welchen Anwendungsfall er nutzen möchte. Diese Auswahl wird im Programm durch ein if-Statement ermöglicht. Bei Eingabe der Zahl Eins, kommt der Nutzer in den Bereich DOWNLOAD. Dort wird ihm angezeigt welche ALKIS Datensätze er downloaden kann. In einem Dictionary sind die verfügbaren URLs der Schnittstellen gespeichert die dem Nutzer in vereinfachter Form angezeigt werden.

```
WFS_dictionary = {
    "NRW-vereinfacht": "https://www.wfs.nrw.de/geobasis/wfs_nw_alkis_vereinfacht?",
    "NRW-AAA-basiert": "https://www.wfs.nrw.de/geobasis/wfs_nw_alkis_aaa-modell-basiert?",
    "NRW-NAS-konform": "https://www.wfs.nrw.de/geobasis/wfs_nw_alkis_nas-konform?",
    "Berlin": "https://fbinter.stadt-berlin.de/fb/wfs/data/senstadt/s_wfs_alkis?",
    "Thüringen": "http://www.geoproxy.geoportal-th.de/geoproxy/services?", #auth notwendig
    "Brandenburg-vereinfacht": "https://isk.geobasis-bb.de/ows/alkis_vereinf_wfs?",
    "Brandenburg-AAA-basiert": "https://isk.geobasis-bb.de/ows/alkis_sf_wfs?",
    "Brandenburg-NAS-konform": "https://isk.geobasis-bb.de/ows/alkis_nas_wfs?SERVICE=WFS&",
    "Mecklenburg-vorpommern": "https://www.geodaten-mv.de/dienste/alkis_wfs_einfach?",
    "Hamburg": "https://geodienste.hamburg.de/WFS_HH_ALKIS_vereinfacht?",
    "Sachsen": "https://geodienste.sachsen.de/aaa/public_alkis/vereinf/wfs?",
    "Bayern": "https://geoservices.bayern.de/wfs/v1/ogc_alkis_ave.cgi?",
    "Sachsen-Anhalt": "", #kostenpflichtig
    "Reinland-pfalz": "", #kostenpflichtig https://www.geoportal.rlp.de/spatial-objects/353
    "Hessen-vereinfacht": "https://www.gds.hessen.de/wfs2/aaa-suite/cgi-bin/alkis/vereinf/wfs?",
    "Hessen-NAS-konform": "https://www.gds.hessen.de/wfs2/aaa-suite/cgi-bin/alkis/nas/wfs?",
    "Niedersachsen": "",
    "Schleswig-Holstein": "",
    "Baden-Württemberg": "",
    "Bremen": "https://geodienste.bremen.de/wfs_flurstuecke?SERVICE=WFS&REQUEST=GetCapabilities",
    "Saarland": " https://geoportal.saarland.de/spatial-objects/325?Service=WFS"
}
```

Ein Dictionary ist eine Ansammlung von Keys und passenden Values. Nach Eingabe des gewünschten Dictionary Keys, wird ein HTTP GET Call an die ausgewählte URL ausgeführt. Die Ausführung der verschiedenen GET Calls wird mithilfe der Python Library



Requests ermöglicht. Durch die Anwendung von einem Try Catch Block werden unerwartete Fehler abgefangen und das Programm nicht in seinem Ablauf gestört. Bei Download Erfolg befindet sich der abgerufene Datensatz dann im gewünschten Ordner. Bei Misserfolg kann der Nutzer eine erneute Eingabe durchführen. Durch Eingabe der Zahl Zwei kommt der Nutzer in den Bereich Transformation. Hier werden durch User Eingabe mehrere Funktionen angestoßen. Hauptbestandteil sind die drei Mapping Dateien, für die drei aufkommenden Schemavarianten. Je nach dem welche Datei transformiert werden soll, wird der entsprechende Dateiname in die Mapping Datei eingetragen. Die Mapping Datei liegt zunächst im Format .yaml vor. In ihr werden die Regeln angegeben, die notwendig sind um eine .xml Datei in das Format .ttl umzuwandeln. Nun folgt eine Beschreibung wie die Mapping Regeln implementiert wurden, anhand der Mapping Datei `alkis_nas_konform.yaml`. In den ersten Zeilen der Datei werden Key Value Paare aus Prefix und Namespace definiert. Hier werden zwei Prefixes definiert, `alkis` und `ex`. In der später vorliegenden Turtle Datei sind die Prefixes dann durch den vollständigen Namespace ersetzt.

```
prefixes:
  alkis: https://w3id.org/alkis/
  ex: http://example.com/
```

Mapping weiter beschreiben. Wie werden S,P,O generiert.

Je nachdem welche Datei transformiert werden soll, wird die richtige Mapping Datei ausgewählt. Es folgt ein Aufruf des Tools `yarrml-Parser`(siehe 5.2 genutzte Technologien). Dieser wandelt dann die `yarrml` Regeln in `R2RML` Regeln um. Hier ein Ausschnitt aus der Datei `R2RML-Mapping.ttl`

```
:rules_000 rdf:type void:Dataset ;
void:exampleResource :map_alkis_flurstueck_000, :map_alkis_flurstuecksnummer_000 .

:map_alkis_flurstueck_000 rml:logicalSource :source_000 ;
rdf:type rr:TriplesMap ;
rdfs:label "alkis_flurstueck" ;
rr:subjectMap :s_000 ;
rr:predicateObjectMap :pom_000, :pom_001 .

:source_000 rdf:type rml:LogicalSource ;
rml:source "/Testdata/NRW/AAA-basiert.xml" ;
rml:iterator "FeatureCollection/member/AX_Flurstueck" ;
rml:referenceFormulation ql:XPath .
```

Dieser Schritt, also die Umwandlung der Mapping Regeln von `yarrml` Regeln in `R2RML` Regeln, ist notwendig da der `rmlMapper` der im nächsten Schritt verwendet wird nur mit diesem Format umgehen kann. Nach erfolgreicher Umwandlung der Mapping Regeln in `R2RML`, wird mithilfe der Python Bibliothek `subprocess` das Programm `rml-mapper` aufgerufen. Die nötigen Parameter die dieses Programm benötigt sind vordefiniert

im Code vorhanden. Der rmlmapper erzeugt nun anhand der vorliegenden Mapping Regeln eine Datei im Turtle Format. Diese enthält alle erzeugten Triple. Hier ein Ausschnitt aus der Datei blabla.ttl

```
ex:DENW42Al1000NqDl_flurstuecksnummer a alkis:AX_Flurstuecksnummer;
  alkis:zaehler "76" .

ex:DENW42Al1000NqDl_gemarkung a alkis:AX_Gemarkung_Schluessel;
  alkis:gemarkung "Losheim";
  alkis:gemarkungsnummer "054394" .

ex:DENW42Al1000NqDl_gemeindezugehoerigkeit a alkis:AX_Gemeindekennzeichen;
  alkis:gemeinde "Hellenthal";
  alkis:land "Nordrhein-Westfalen";
  alkis:regierungsbezirk "Köln" .

ex:DENW42Al1000NqDl_lebenszeitintervall a alkis:AA_lebenszeitintervall;
  alkis:beginnt "2016-12-05Z" .

ex:DENW42Al1000NqfC_flurstuecksnummer a alkis:AX_Flurstuecksnummer;
  alkis:zaehler "54" .

ex:DENW42Al1000NqfC_gemarkung a alkis:AX_Gemarkung_Schluessel;
  alkis:gemarkung "Losheim";
  alkis:gemarkungsnummer "054394" .

ex:DENW42Al1000NqfC_gemeindezugehoerigkeit a alkis:AX_Gemeindekennzeichen;
  alkis:gemeinde "Hellenthal";
  alkis:land "Nordrhein-Westfalen";
  alkis:regierungsbezirk "Köln" .

ex:DENW42Al1000NqfC_lebenszeitintervall a alkis:AA_lebenszeitintervall;
  alkis:beginnt "2016-08-09Z" .

ex:DENW42Al1000Nscu_flurstuecksnummer a alkis:AX_Flurstuecksnummer;
  alkis:zaehler "28" .
```

Hier zu sehen sind die erzeugten Triples. Beschreibung folgt.

Im dritten Bereich des Programmes gibt der Nutzer den Pfad zu der Datei an, die letztlich in der Graph Datenbank Apache Fuseki gespeichert werden soll. Nach erfolgreicher Eingabe, wird eine Update Query an den Fuseki Endpoint gesendet und der Graph wird aktualisiert. Das Ansprechen des Fuseki Endpoint über den Python Programmcode wird mit der Bibliothek rdflib ermöglicht.

```
def saveGraph(path):
    query_endpoint = 'http://localhost:3030/ds/query'
    update_endpoint = 'http://localhost:3030/ds/update'

    store = sparqlstore.SPARQLUpdateStore()
    store.open((query_endpoint, update_endpoint))

    alkis_graph = URIRef('http://example.org/alkis_graph')
    store = Graph(store, identifier=alkis_graph)

    store.load(path, format="ttl")
```

Im letzten Bereich des Programmes kann der Nutzer die erzeugten Daten, die nun in der Graph Datenbank vorliegen, abfragen. Dies wird ermöglicht indem er selbst eine SPARQL Query erstellt oder eine vordefinierte Abfrage auswählt. Nach Verarbeitung der Anfrage an den Fuseki Server, antwortet dieser mit einem Ergebnis. Dieses beinhaltet RDF Triples im Turtle Format. Hier ein Beispiel zu der Abfrage aller Triples:

```
asdf
```

## 5.2 genutzte Technologien

Um die Anwendung zu entwickeln wurde die Programmiersprache Python gewählt. Als IDE(Integrated Development Environment) wurde PyCharm benutzt. Zur Versionsverwaltung wurde die Software git, mit dem Repository github genutzt. Der Zugang zum Programm Code und einer Anleitung zur Installation des Programmes wird bereit gestellt unter dem Link: <https://github.com/banzaiiiii/ALKIS-transform> Zur Umwandlung der Mapping Regeln im .yaml Format in R2RML wurde der Yarrml Parser <https://github.com/RMLio/yarrml-parser> benutzt. Zur schlussendlichen Transformation der Daten in das Turtle Format wurde der rmlmapper <https://github.com/RMLio/rmlmapper-java> verwendet. Weiterhin wurde zur Verarbeitung von RDF im Code, die Python Bibliothek rdflib 6.1.1 <https://github.com/RDFLib/rdflib/#getting-started> genutzt. Als RDF Triple Store oder auch Graph Datenbank wurde Apache Jena Fuseki verwendet. <https://jena.apache.org/documentation/fuseki2/>

<https://www.ontotext.com/knowledgehub/fundamentals/what-is-rdf-triplestore/>  
<https://www.youtube.com/watch?v=eDRfKAKZMQE>  
<https://db-engines.com/en/system/GraphDB>

## 6 Ergebnisse und Evaluation

Wurde die These/Forschungsfrage gelöst.

ALKIS Daten haben mangelhafte Auffindbarkeit und daraus resultierende Nutzbarkeit der Daten für Zwecke abseits der professionellen Nutzung durch GIS Programme. Wie hat sich das Nutzungspotenzial der Daten durch die Transformation der Daten geändert?

## 7 Ausblick

Die Steuerung des Programms durch die Zahleneingabe Eins bis Vier vereinfacht das Programm und lässt sich auf die Methodik des Prototypings zurückführen. Um die Nutzerfreundlichkeit in Zukunft weiter zu erhöhen sollte ein solches Programm perspektivisch eine Web Anwendung sein. Außerdem können zum jetzigen Zeitpunkt nicht alle

ALKIS Schnittstellen der Bundesländer angesprochen werden, da wie bereits erwähnt, einige Bundesländer eine vorherige Anmeldung benötigen oder Inhalte nur kostenpflichtig zur Verfügung stellen. (siehe Tabelle Abschnitt so und so) Um dieses Problem zu lösen, müsste man das Programm , um alle Schnittstellen abrufen zu können um ein Benutzermanagement erweitern. Dies hängt allerdings auch von den einzelnen Bundesländern selbst ab, ob sie sich in Zukunft dazu entscheiden ALKIS Daten unter einer Open Data Lizenz zu veröffentlichen.

Ein weiterer Punkt der nützlich wäre ist die Transformation von weiteren ALKIS Daten und das inbetracht ziehen von weiteren Quellen. Neben den Datensätzen über Flurstücke, die in dieser Arbeit transformiert werden, enthält das AAA Modell viele weitere Datensätze über zum Beispiel Gebäude, tatsächliche Nutzung etc..(siehe geoinfodoc)

Noch ein Punkt ist die Anreicherung der transformierten RDF Graphen durch weitere Daten.... Dies würde den erstellten RDF Wissensgraph noch nützlicher für etwaige Abfragen machen.

## **7.1 Notizen**

- Zugriffsbeschränkungen sowie Lizenzbestimmungen sind aus getcapabilities Aufruf entnommen - gml:id ist dokumentenweit eindeutig(durch entstehungsdatum-zeit)(quelle geoinfodoc s92)

- 4 prinzipien: tim berner lee - nutzen von URI's als namen für objekte - zur eindeutigen identifikation für dinge - nutze HTTP URIs - so können Menschen die Objekte nachschauen - nutze anerkannte standarts( RDF, Sparql) - linke URIS weiter

## 8 Quellenverzeichnis

- 4.2.22

zu einleitungserklärungen

- neo4j vs graph db:

Objektenartenübersicht ALKIS: <https://www.adv-online.de/icc/extdeu/nav/a63/binarywriterservlet?imgUid=b001016e-7efa-8461-e336-b6951fa2e0uBasVariant=11111111-1111-1111-1111-111111111111> Objektartenkata-  
log Seite: 22

GeoInfoDoc: <https://www.adv-online.de/GeoInfoDok/binarywriterservlet?imgUid=e9e304b7-96d3-de71-d261-dde30c21d06f&uBasVariant=11111111-1111isDownload=true>

WFS Spezifikation: [file:///C:/Users/robin/Downloads/05-008c1\\_OGC\\_Web\\_Services\\_Common\\_Specification\\_Corrigendum.pdf](file:///C:/Users/robin/Downloads/05-008c1_OGC_Web_Services_Common_Specification_Corrigendum.pdf)