

Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D-70569 Stuttgart

Bachelorarbeit Nr. 214

Webbasierte, visuelle Exploration von Regesten deutsch-römischer Könige

Christian Richter

Studiengang:	Softwaretechnik
Prüfer/in:	Prof. Dr. Thomas Ertl
Betreuer/in:	Dr. Steffen Koch, M.Sc. Markus John
Beginn am:	21. April 2015
Beendet am:	21. Oktober 2015
CR-Nummer:	H.5.1 H.5.2

Kurzfassung

Auf dem Online-Portal der Regesta Imperii stellt die Akademie der Wissenschaften und Literatur Mainz einen umfangreichen Datensatz von urkundlichen und historiographischen Quellen der deutsch-römischen Könige von den Karolingern bis zu Maximilian I. zur Verfügung. Teil dieser Regestendatenbank ist eine Regestensuche mit einer zusätzlichen Expertensuche.

Analysen haben gezeigt, dass die zur Verfügung gestellten Möglichkeiten zur Suche jedoch nicht optimal ausgeschöpft werden.

Aus diesen Grund wird in dieser Arbeit ein Visualisierungsansatz vorgestellt, welcher den Zugang zu den Regesten weiter verbessert, indem Nutzern der Regesta Imperii die Möglichkeit gegeben wird, mit Hilfe einer interaktiven Visualisierung die Ergebnismenge leichter zu explorieren und einzuschränken.

Inhaltsverzeichnis

1	Einleitung	7
1.1	Motivation	7
1.2	Ausgangslage	7
1.3	Vorgehen	11
1.4	Gliederung	12
2	Grundlagen	13
2.1	Regesta Imperii	13
2.2	Webanwendung	19
3	Verwandte Arbeiten und Technologien	21
3.1	Daten	21
3.2	Visualisierungsansätze	21
3.3	Brushing und Linking	30
3.4	Verwandte Probleme	31
3.5	Multi Coordinated View	33
3.6	Verwendete Technologien	34
4	Lösungsansatz	37
4.1	Visualisierung	37
5	Implementierung	43
5.1	Datenaufbereitung	43
5.2	Visualisierung	44
6	Ergebnisse	47
6.1	Verbesserungen	47
6.2	Use Case	47
6.3	Expertenfeedback	48
7	Zusammenfassung und Ausblick	51
7.1	Herausforderungen	51
7.2	Zusammenfassung	52
7.3	Ausblick	52
7.4	Fazit	54
	Literaturverzeichnis	55

Abbildungsverzeichnis

1.1	Schnellsuche der Regesta Imperii	7
1.2	Expertensuche der Regesta Imperii	8
1.3	Gantt Diagramm zur Planung dieser Arbeit	11
2.1	Digitalisat eines Regests [RIo]	14
2.2	Digitalisat eines Registers	18
2.3	Schematischer Aufbau einer Web-Anwendung	20
3.1	Simile Timeline Screenshot	22
3.2	Ableton Live Screenshot	23
3.3	Heatmap über Open Street Map	24
3.4	Bubble Glyphs über Open Street Map	25
3.5	Alphaslider [AT95]	26
3.6	Lensbar [len]	27
3.7	LensList [fis]	28
3.8	Indented Tree	29
3.9	Screenshot der Christchurch 2010 Timeline	30
3.10	DARIAH-DE Geo-Browser	32
3.11	PatViz	33
4.1	Screenshot des Visualisierungsansatzes	37
4.2	Timeline	38
4.3	Karte	39
4.4	Register	40
4.5	Übersicht über selektierte Ausstellungsorte und Entitäten	41
4.6	Tabelle mit Ergebnissen einer Suchanfrage	41
5.1	Schematischer Aufbau der Entwickelten Software	44
6.1	Use Case Diagram	48
7.1	Relative Verteilung von Regesten über die Zeit	53
7.2	Colorcoding für Personen und Orte im Register	53

Verzeichnis der Listings

2.1	Gekürzte Version eines Regests im XML-Format	15
2.2	Die Speicherung von Informationen ist nicht einheitlich	16
2.3	Beispieleintrag eines Registers	19
5.1	Modell zur Speicherung von Regesten	45
5.2	Client-Format der ausgewählten Einträge	46

1 Einleitung

1.1 Motivation

Durch das Internet stehen immer größere Datenmengen zur Verfügung. Viele dieser Daten werden frei, etwa unter CC-Lizenz ¹, bereitgestellt. Gerade die Digitalisierung existierender Daten ermöglicht das Untersuchen von Datensätzen, die über lange Zeit gesammelt wurden. Dieses gesammelte Wissen kann nun mit Hilfe neuer Technologien erforscht werden. Ein Ansatz für die Untersuchung dieser neuen Daten ist das Entwickeln einer interaktiven Visualisierung. Die Regesta Imperii ermöglichen es Daten, die seit 1829 gesammelt werden, bei Nennen der Quelle, für neue Projekte zu verwenden. Im Rahmen dieser Arbeit soll deswegen eine Visualisierung entworfen werden, um diese Daten weiter erkunden und erforschen zu können.

1.2 Ausgangslage

Die Regesta Imperii sind ein Grundlagenwerk, bestehend aus urkundlichen und historiographischen Quellen der römisch-deutschen Könige von den Karolingern bis zu Maximilian I. Um den vorhandenen Datensatz von ungefähr 125000 Regesten möglichst vielen Nutzern zugänglich zu machen, wurde ein Online-Angebot entwickelt, welches mit einer Suchfunktion das Finden von Regesten vereinfachen soll. Die Suchfunktion kann in zwei unterschiedlichen Suchmodi verwendet werden.

Zum einen steht eine Schnellsuche zur Verfügung, welche wie in Abbildung 1.1 zu sehen ist, nur aus einem einzigen Suchfeld besteht und ähnlich dem Suchangebot von Google versucht eine möglichst sinnvolle Antwort zu generieren. Die Schnellsuche ist demnach eine einfache Volltextsuche.

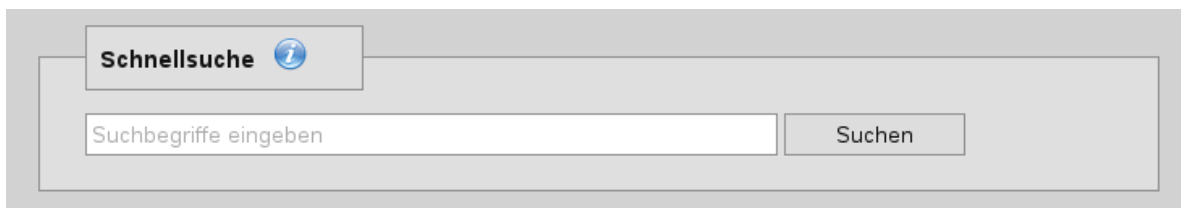
The image shows a web interface for a search function. At the top, there is a tab labeled 'Schnellsuche' with a blue information icon. Below this, there is a large search input field with the placeholder text 'Suchbegriffe eingeben'. To the right of the input field is a button labeled 'Suchen'. The entire search area is enclosed in a light gray border.

Abbildung 1.1: Schnellsuche der Regesta Imperii

¹<https://creativecommons.org/licenses/>

1 Einleitung

Um es Nutzern mit entsprechendem Fachwissen zu ermöglichen, Suchparameter auf ihre Ansprüche anzupassen, wurde eine Expertensuche entwickelt. Diese besteht aus dem in Abbildung 1.2 dargestellten Suchformular.

Expertensuche ⓘ

Suche: Stichworte ▾ Suchbegriffe eingeben

+ −

Textauswahl: ☒ Regestentext ☒ Überlieferung ☒ Kommentar ☒ Nachträge

Schreibweisen: ☒ toleranter Modus ☐ exakter Modus

Von Datum JJJJ-MM-TT **Bis Datum** JJJJ-MM-TT

Abteilung Alle Abteilungen ▾

Band Alle Bände ▾

Nummern Regestennummern

Relevanzgrenze 10.000 Treffer ▾

Absenden

Abbildung 1.2: Expertensuche der Regesta Imperii

Mit der Expertensuche kann nach Stichwörtern, Phrasen und Ausstellungsorten gesucht werden. Diese können mit den logischen Operatoren „und“, „oder“ und „nicht“ kombiniert werden, um wie in der Online-Hilfe[Rlh] beschrieben, nach „Wein“, jedoch nicht nach „Weinsberg“ oder „Weinheim“ zu suchen. Die Suche der Regesta Imperii unterstützt demnach die aussagenlogischen Operationen, Konjunktion, Disjunktion und Negation.

Weiterhin kann der Nutzer auswählen, welche Textarten er durchsuchen möchte, in welchem Zeitraum die Veröffentlichung des Regests liegen soll und welche Bände der Regesta Imperii durchsucht werden sollen.

Abschließend kann bestimmt werden, wie viele Suchergebnisse angezeigt werden sollen. Dieser Wert lässt sich von 1000 bis unbegrenzt vielen Einträgen wählen und ist standardmäßig auf 10000 Treffer festgelegt. Dies stellt gleichzeitig die größte Herausforderung bei der Nutzung der Suche dar. Wählt man die Ergebnismenge zu klein, werden vielleicht wichtige Ergebnisse nicht angezeigt, wählt man die Ergebnismenge zu groß, kann man die Fülle an Informationen kaum bis gar nicht bewältigen.

1.2.1 Problemstellung

Ziel der Arbeit „Webbasierte, visuelle Exploration von Regesten deutsch-römischer Könige“ ist es, den Zugang zu den Regesten der Regesta Imperii weiter zu verbessern. Dafür wird eine interaktive Visualisierung entworfen, die den Nutzer bei der Exploration der Regesten unterstützt. Diese Visualisierung folgt einem MCV-Ansatz und ermöglicht es, mittels Brushing und Linking den Datensatz von Friedrich III. zu explorieren.

Dadurch soll es sowohl für Historiker mit entsprechendem Fachwissen, als auch fachfremden Nutzern mit Interesse an den Dokumenten von Friedrich III. einfacher gemacht werden, Daten zu erkunden und in ihre Arbeit einfließen zu lassen.

1.2.2 Daten

Die Regesta Imperii stellen für diese Arbeit einen Datensatz von Urkunden von Friedrich III. zur Verfügung. Zusätzlich können die Registerbände zu Friedrich III. verwendet werden. Diese beiden Datensätze stehen im XML-Format zur Verfügung und müssen für eine Visualisierung aufbereitet werden. In Abschnitt 2.1 werden diese Daten vorgestellt und ihre (technischen) Eigenschaften erläutert.

1.2.3 Anforderungen

Im Rahmen dieser Arbeit soll eine Visualisierung entwickelt werden, die es ermöglicht den von den Regesta Imperii gestellten Datensatz mit neuen Mitteln zu erforschen. Dabei werden gewisse Anforderungen an die Software gestellt. Diese Anforderungen wurden in Absprache mit den Regesta Imperii bestimmt und festgehalten. Eingeteilt werden diese in funktionale und nicht-funktionale Anforderungen. Genauer wird diese Einteilung in [RR12] beschrieben.

Funktionale Anforderungen

Funktionale Anforderungen sind Antworten auf die Frage „Was soll das zu entwickelnde System leisten?“. Diese, für die Visualisierung festgestellten Anforderungen, werden im Folgenden festgehalten.

Die zu entwickelnde Software soll es dem Nutzer ermöglichen, ähnlich der Expertensuche der Regesta Imperii, nach Regesten zu suchen. Dazu kann der Nutzer Filterkriterien in der Software einstellen und erhält als Ergebnis genau die Regesten, die den Filterkriterien entsprechen. Filterbar sind dabei das Ausstellungsdatum und der Ausstellungsort eines Regests, sowie die im Regest erwähnten Personen, Orte und Institutionen.

Nicht-funktionale Anforderungen

Nicht-funktionale Anforderungen beantworten die Frage „Wie soll das zu entwickelnde System die Leistung erbringen?“. Diese Anforderungen werden im Folgenden beschrieben.

Die Visualisierung soll in einem „akzeptablen Rahmen“ auf Anfragen des Nutzers reagieren. Das bedeutet, dass keine langen Wartezeiten auftreten sollen. Genauer wurde diese Anforderung nicht bestimmt. Die Oberfläche des Systems muss so gestaltet sein, dass sie auch von Nutzern bedient werden kann, die keinen informationstechnischen Hintergrund haben. Also insbesondere Historiker und Studierende historischer Studiengänge sollen die Software einfach nutzen können.

Weiterhin muss die Bereitstellung des Systems einfach gestaltet sein, um die Visualisierung in das existierende Online-Angebot der Regesta Imperii einbinden zu können.

1.3 Vorgehen

Damit eine methodische Durchführung dieser Arbeit gewährleistet werden kann, wurde ein Projektplan erstellt und mit Hilfe von Meilensteinen der zeitliche Fortschritt überwacht.

Diese Bachelorarbeit wurde, als Projekt nach dem in [Ben83] vorgestellten Wasserfallmodell für die Softwareentwicklung, geplant. Dabei wird das Projekt in aufeinander aufbauende Phasen eingeteilt. Diese Phasen sind:

1. Anforderungsanalyse (Bestimmung der Anforderungen an das System)
2. Systemdesign (Planung der Softwarearchitektur)
3. Implementierung (Programmierung der Software)
4. Systemtest (Überprüfen der Software)

Im Rahmen dieser Arbeit wurde dieses Modell auf das Erstellen derselbigen angepasst. Es wurden die folgenden neuen Phasen definiert:

1. Analyse (Anforderungsanalyse/ Literaturrecherche)
2. Entwurf (Entwurf der Visualisierung/ Initiale Implementierung)
3. Test (Test der Software/ Erste Teile der Ausarbeitung)
4. Dokumentation (Fertigstellung der Ausarbeitung)

Der Ablauf sowie Meilensteine können dem Gantt-Diagramm in Abbildung 1.3 entnommen werden.

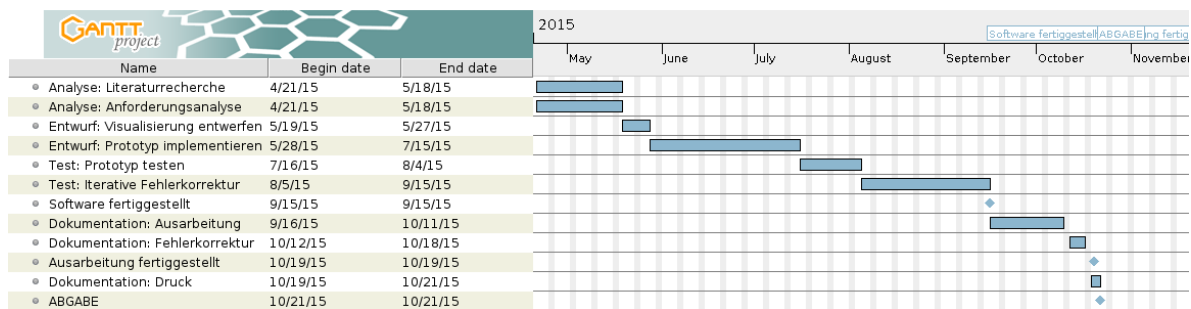


Abbildung 1.3: Gantt Diagramm zur Planung dieser Arbeit

Wie in Abbildung 1.3 zu erkennen, wurde innerhalb der Phasen wieder der Ansatz des Wasserfalls verwendet. Die Entwicklung der Visualisierung war demnach ein eigenes „kleines Wasserfallprojekt“, ebenso die Dokumentation beziehungsweise Erstellung der Ausarbeitung.

1.4 Gliederung

Diese Arbeit befasst sich mit dem Entwurf und der Entwicklung einer Visualisierung für historische Daten. Die Arbeit ist dabei wie folgt gegliedert:

Kapitel 1 – Einleitung: präsentiert eine Einführung in die Themen dieser Arbeit.

Kapitel 2 – Grundlagen: geht auf die Grundlagen dieser Arbeit ein. Insbesondere die Regesta Imperii und deren Anforderungen.

Kapitel 3 – Verwandte Arbeiten und Technologien: betrachtet verwandte Arbeiten und die von dieser Arbeit eingesetzten Technologien.

Kapitel 4 – Lösungsansatz: wird die Ausgangslage, den Lösungsansatz, und alternative Lösungen dieser Arbeit präsentieren.

Kapitel 5 – Implementierung: stellt die Implementierung des Systems vor und beschreibt wichtige Designentscheidungen.

Kapitel 6 – Ergebnisse: behandelt die Ergebnisse, die mit einer interaktiven Visualisierung erzielt wurden.

Kapitel 7 – Zusammenfassung und Ausblick fasst die Ergebnisse und Hindernisse des Projektes zusammen und liefert einen Ausblick auf Möglichkeiten zum Ausbau des Systems.

2 Grundlagen

In diesem Kapitel werden Grundlagen der Arbeit präsentiert. Anfangs wird die Arbeit der Regesta Imperii vorgestellt und Begrifflichkeiten in deren Umfeld definiert. Im Anschluss werden einige Begriffe aus den Themengebieten Softwareentwicklung und Visualisierung erklärt und ihr Zusammenhang mit dieser Arbeit erläutert.

2.1 Regesta Imperii

Die Regesta Imperii sind eine Sammlung von ca. 125000 urkundlichen und historographischen Quellen der deutsch-römischen Könige von den Karolingern bis zu Maximilian I. Die Urkunden werden sowohl in Buchform als auch digital auf der Website der Regesta Imperii ¹ veröffentlicht. Das Angebot ist dabei aufgeteilt in Regesten- und Registerbände.

2.1.1 Regest

Ein Regest ist eine kurze inhaltliche Zusammenfassung eines Textes. Es enthält die wichtigsten Meta-Daten eines Dokuments. Die Regesta Imperii stellen folgende Daten zur Verfügung:

- Titel des Dokuments
- IDs zur eindeutigen Zuordnung
- Veröffentlichungsdatum
- Veröffentlichungsort als Name und gegebenenfalls Koordinaten
- Kurzzusammenfassung des Dokuments

Ursprünglich wurden die Regesten der Regesta Imperii 1829 in Buchform erstellt und veröffentlicht. Abbildung 2.1 zeigt das Digitalisat eines Regests, das von der Regesta Imperii veröffentlicht wurde. Die Überschrift enthält dabei Ausstellungsdatum und -ort der Urkunde. Anschließend folgt eine Zusammenfassung des Inhalts der Urkunde. Diese Regesten wurden mittlerweile digitalisiert und liegen nun im XML-Format vor. Listing 2.1 zeigt ein gekürztes Regest im XML-Format wie es von den Regesta Imperii zur Verfügung gestellt wird.

¹<http://regesta-imperii.de>

1440 Februar 6, Wien**1**

H.z.F. (V.) von Österreich¹ und H.z. Albrecht (VI.) von Österreich geloben, zur Abstellung der zwischen ihnen bestehenden Streitigkeiten den Schiedsspruch zu befolgen, den ihre (unten aufgeführten) Spruchleute einmütig oder mehrheitlich bis *nachsten freytag vor dem suntag, so man singet Letare in der vasten* (1440 März 4), fällen und brieflich bestätigen sollen. Die Spruchleute seien bereits schriftlich und mündlich über ihre (der Hzz.) Vorbringen unterrichtet worden und würden, wenn notwendig, weitere Vorbringen anhören. Sollten Spruchleute wegen Krankheit oder aus anderen Gründen ausfallen, würden H.z.F. bzw. Albrecht an deren Stelle andere Personen bevollmächtigen. Um die Sache *unverdecktlich* durchführen zu können, sollen Spruchleute, die ihnen besonders *gelobt und verpunden* sind, für die Zeit der Verhandlungen von *glübd und verpuntnuss* ledig sein. Als Spruchleute wurden von H.z.F. seine Räte Bf. Nikodemus von Freising, Gf. Johann von Schaunberg, Oberstmarschall in Steyr, Hans von Neitperg, Hans (III.) von Stubenberg, Hauptmann in Steyr, Konrad Zeidler, sein Kanzler und Pfarrer zu Pürgg (zu der Purg)², Pankraz Rindscheit, Landschreiber in Steyr, Walther Zebinger, Pfleger zu Pfannberg, und Leopold Aspach, und von H.z. Albrecht Bf. Leonhard von Passau, Stephan von Hohenberg, Rudolf von Tiernstein, Jörg Scheck von Wald, Hans Schweinbarther, Andreas Süssenheimer, Jörg Schweinpeck und Konrad Peßnitzer ernannt. *An sand Dorothen tag*.

KVr: *D(omini) duces in consilio* (rechts unter der Plica). – Eigenhändiger Rekognitionsvermerk von H.z.F.: *Prescripta recognoscimus* (links unter der Plica)³.

Org. im HHStA Wien (Sign. AUR, sub dat. 1439 XI 26⁴), Perg., rotes S 3 in wachsfarbener Schlüssel an Ps. und rotes S H.z. Albrechts VI. (S.A.V. S. 149, Abb. 90) in wachsfarbener Schlüssel an Ps.⁵. – Kop.: Abschrift (18. Jh.) ebd. (Sign. Urkundenabschriften, Collationierte Copien, Karton 4).

Reg.: LICHNOWSKY(-BIRK) 6 n. 29; QGStW I/7 n. 14834.

Lit.: BAUM, Albrecht VI. 1 S. 25.

Zum weiteren Verlauf der Streitbeilegung zwischen Kg.F. und H.z. Albrecht vgl. n. 4 u. 17.

¹ Die Wahl zum römischen Kg. (1440 Februar 2) nahm H.z.F. erst am 6. April 1440 zu Wiener Neustadt an, vgl. RTA 15 n. 100, Lhotsky, Zur Königswahl S. 165, u. Regg.F.III. H. 4 n. 1.

² Vgl. LICHNOWSKY(-BIRK) 5 n. 3626 u. GÖHLER, Das Wiener Kollegiat-, nachmals Domkapitel S. 73ff. n. 6.

³ Hingewiesen sei auch auf einen rücks. Archivierungsvermerk (Mitte 15. Jh.): *Ain taydingbrief zwischen baiden meinen herren h(ern) Frid(rich) und h(ern) Albr(echt)* (Es folgt Fortsetzung von einer späteren Hand des 15. Jh.). Vermerke wohl von der ersten Hand finden sich auf n. 17, 20 (jeweils Ausfertigung A) u. 19, 132 (jeweils Ausfertigung B).

⁴ Sign. gemäß Datum beiliegender Urk. H.z. Albrechts VI. von Österreich, mit der sich dieser zur Bestellung von Spruchleuten verpflichtete; Reg: LICHNOWSKY(-BIRK) 6 n. 9; QGStW I/7 n. 14826.

⁵ Auf dem Ps. steht rücks. ein Vermerk: 40.

Abbildung 2.1: Digitalisat eines Regests [R10]

Listing 2.1 Gekürzte Version eines Regests im XML-Format

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <cei xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:noNamespaceSchemaLocation="http://www.cei.lmu.de/schema/cei060122.xsd">
4   <teiHeader>
5     <fileDesc>
6       <titleStmt>
7         <title>Ludwig - [RI VII] H. 2 n. 2</title>
8       </titleStmt>
9       <editionStmt>
10        <p n="volume">[RI VII] H. 2 - Ludwig der Bayer, Baden</p>
11        <p n="repository">Regesta Imperii Online:
12          <ref type="external" target=
13            "http://www.regesta-imperii.de/cei/007-002-000/sources/1314-10-29_1_0_7_2_0_2_2">
14          </ref>
15        </p>
16      </editionStmt>
17      <sourceDesc>
18        <bibl>
19          <idno n="uri">1314-10-29_1_0_7_2_0_2_2</idno>
20          <idno
21            n="exchange">ri_1314-10-29_000001_000001_007_002_000_000002_0000000002</idno>
22          <idno n="department">007</idno>
23          <idno n="volume">002</idno>
24          <idno n="issue">000</idno>
25        </bibl>
26      </sourceDesc>
27    </fileDesc>
28    <charter>
29      <chDesc>
30        <head>
31          <idno>[RI VII] H. 2 n. 2</idno>
32          <issued>
33            <issueDate>
34              <date value="1314-10-29">[1314] Oktober 29</date>
35            </issueDate>
36            <issuePlace>
37              <placeName key="Mainz">Mainz</placeName>
38              <location><geo decls="#WGS">47.8333, 11.9667</geo></location>
39            </issuePlace>
40            <dateOrig><hi rend="italic">Datum Moguntie, quarto kalen.
41              Novembris</hi>.</dateOrig>
42          </issued>
43        </head>
44        <relevantPersonal>
45          <issuer><persName>Ludwig</persName></issuer>
46        </relevantPersonal>
47        <abstract>Abstract</abstract>
48        <diplomaticAnalysis>Analyse</diplomaticAnalysis>
49        <div type="resources">
50          <list><item><ref
51            target="http://regesta-imperii.digitale-sammlungen.de/seite/ri07_wet1994_0022"
52            type="external">Digitalisat der Buchseite</ref></item></list>
53        </div>
54      </chDesc>
55    </charter>
56  </cei>

```

2 Grundlagen

Die XML-Dateien sind in zwei Teile gegliedert. Der TEI-Header enthält Metadaten des Regests. Dazu zählen der Titel des Regests (Listing 2.1 Z. 7) und IDs zur eindeutigen Bezeichnung des Regests (Listing 2.1 Z. 19-23). Unter Charter wird der Inhalt des Regests beschrieben. Relevant sind dabei der Kurzbezeichnung des Regests (Listing 2.1 Z. 31), das Ausstellungsdatum (Listing 2.1 Z. 34) und der Ausstellungsort (Listing 2.1 Z. 37-38).

Das Ausstellungsdatum steht dabei auf zwei Arten zur Verfügung. Einmal in deutscher Sprache als Inhalt des „date“-Tags und einmal im in ISO 8601 ([ISO04]) beschriebenen Format „YYYY-MM-DD“. Dabei wird das Jahr mit vier Ziffern beschrieben, gefolgt von zwei Ziffern für den Monat, auf die wieder zwei Ziffern für den Tag stehen.

Der Ausstellungsort steht als Name zur Verfügung. Dieser wird als Inhalt des „placeName“-Tags gespeichert. Sollten Koordinaten zu einem Ort bekannt sein, werden diese als Inhalt des „geo“-Tags in Form von WGS-Koordinaten [Nat00] gespeichert.

Da die Regesten händisch digitalisiert wurden, enthalten die XML-Dateien teilweise Syntax- oder andere Übertragungsfehler. Weiterhin sind Informationen nicht immer in einem einheitlichen Format gespeichert. In Listing 2.2 werden diese unterschiedliche Darstellung der Daten noch einmal verdeutlicht.

Listing 2.2 Die Speicherung von Informationen ist nicht einheitlich

```
<placeName key="Neustadt">Neustadt</placeName>

<placeName>Lanstein</placeName>

<placeName>(s. l.)</placeName>

<placeName>[-]<note n="1">Datierung nach <ref target=
"http://opac.regesta-imperii.de/lang_de/kurztitelsuche_r.php?kurztitel=regg.f.iii._h._12">
Regg.F.III. H. 12</ref> n. 325. </note></placeName>

<placeName>[-]</placeName>
```

Ist der Ausstellungsort bekannt, wird dieser in ein Tag „placeName“ gespeichert. Teilweise wird diesem Tag auch ein Attribut „key“ zugewiesen. Die Tags enthalten jedoch, wie in der dritten Zeile zu erkennen, nicht immer den Ausstellungsort, sondern natürlichsprachliche Annotationen. Dies ist darauf zurückzuführen, dass die XML-Dateien auch verwendet werden, um das Web-Angebot der Regesta Imperii zu generieren. Wenn der Ausstellungsort eines Regests nicht bekannt ist, werden unterschiedliche Ansätze angewandt. Teilweise existieren die „placeName“-Tags nicht. Sollten trotzdem Informationen zu einem Ort existieren, werden diese wie in Zeile vier nach einem speziellen String, in einem „note“-Tag annotiert. Dieser String wurde jedoch auch in unterschiedlichen Formen verwendet, um zu zeigen, dass der Ausstellungsort nicht bekannt ist.

Probleme dieser Art sind in allen Regesten und innerhalb verschiedenster Tags zu finden. Im Rahmen dieser Arbeit mussten die Daten auf diesem Grund erst aufbereitet werden, um sie für eine Visualisierung nutzen zu können.

2.1.2 Register

Zusätzlich zu den Informationen eines Regests, stellen die Regesta Imperii auch Register zur Verfügung. Dies sind alphabetisch sortierte, in Bänder eingeteilte Listen von Orten, Personen und Institutionen, welche in den Regesten erwähnt werden. Ein Registereintrag enthält die folgenden Informationen:

- Name des Ortes, der Person oder der Institution
- Nummern der Regesten in denen die Entität erwähnt wird
- gegebenenfalls eine Referenz auf einen anderen Registereintrag
- eine eindeutige ID

Referenzen werden verwendet, falls ein Ort, eine Person oder eine Institution unter mehreren Namen bekannt ist. Für eine Entität werden dann alle Registernummern angegeben, in denen sie erwähnt wird. Alle weiteren Vorkommen der Entität mit anderen Namen, referenzieren deshalb auf die erste Entität, welche die benötigten Informationen enthält.

Registereinträge sind in Stufen eingeteilt. Diese Stufen bilden eine Hierarchie, die eine „beinhaltet“-Beziehung darstellt. Beispielsweise steht auf „Stufe0“ eine Stadt, auf „Stufe1“ das Gericht der Stadt und auf „Stufe2“ ein Richter, welcher für dieses Gericht arbeitet.

Abbildung 2.2 zeigt das Digitalisat einer Seite aus einem Registerband, wie er von den Regesta Imperii zur Verfügung gestellt wird. In dieser Fassung werden die Stufen nicht ausgeschrieben, sondern durch Einrückungen und vorgestellte Geviertstriche dargestellt. Auf den Namen einer Person oder eines Ortes folgen die Nummern der Regesten, in denen die Person oder der Ort erwähnt wird.

- Aginno* siehe Agen
 Agius, Pfalzpriester, Bischof von Orléans (843-867) (†)132, 144, 447
 Agnellus von Ravenna, Schriftsteller, Teilnehmer der Schlacht von Fontenoy 216-219, 225f
 Agout, Fluß in Südwestfrankreich, linker Nebenfluß des Tarn (†)436
 Aigues-Mortes, dép. Gard, arr. Nîmes 435
 Aimerico, Abt von Dèvre (ca. 844) 420
 Aimerius, Vasall Karls 554
 Aimoin, Mönch in Saint-Germain-des-Prés († nach 896), Geschichtsschreiber 462, 466, 469-471, 473, 475f.
 Ainardus, Abt von Saint-Martial zu Limoges (848) 594
 Ainardus, Bischof von Périgueux (843/4) †603
 Airan, dép. Calvados, arr. Caen, c^m Bourguébus 394
 Aisne, Fluß in Nordfrankreich, linker Nebenfluß der Oise 533, 539, 555
 Alagon, katalan. Geschlecht †459
 Alaón (Alaó, Alaon), Spanien, com. Baja Ribagorza, term. Sopeira †459
 – Kloster Santa Maria †459
 – – Äbte siehe Obbonius
 Albanyà, Spanien, com. Alt Empordà 410
 Albario (*Leocarcari in monte Albario*) siehe Cantalops
 Albère siehe L'Albère und Mont d'Albère
 Albi, dép. Tarn 407
 – Gau (†)436
 Albinian, dép. Hérault, arr. Béziers, c^m Agde, c^{nc} Vias 417
Albinianus, Ort in der Grafschaft Béziers (Albinian oder Abeilhan?) 417
Albinianus, Ort im Gau von Besalú siehe Albanyà
 Aldrich (800-857), Bischof von Le Mans (832-857) 66, 74, 122, 146, 148-150, 236, 496
 Aldrich, Erzbischof von Sens (828/9-836) 65, 518
 Aledramnus (820-855), Graf von Troyes, Statthalter Barcelonas 610
 Alemannen, Alemannien 6, 9-11, 19f, 23-25, 28, 35, 189, 293
 Alençon, dép. Orne 268, (†)393
 Alexandria, Ägypten 10
 Alkuin (ca. 730-804), Gelehrter 187
 Allier, frz. Fluß, linker Nebenfluß der Loire 451
 Alpen, europäisches Gebirgsmassiv 319
 Altheus, Bischof von Autun (843-849) 273, 366
 Altmar, Abt von Montier-en-Der (845), Missus Karls (853) 372, 374, 376, 378, 380, 484, 495
 Altmar, Seneschall der Ks.in Judith (838) 380
 Amalbert, Geistlicher, Inhaber von Reimser Kirchengut 379, 495
 Amalrich / Amalricus, *levita* des Klosters Saint-Martin zu Tours (845), Erzbischof von Tours (nach 850-855) 455, †603
Amancianica / Martinianica, colonica im Gau von Maguelone 435
Amaus (L'Amous, Region um Dole?; vgl. Ann. Bertiniani, ed. GRAT, S. 254 s.v. Amous)
 – Grafschaft 82
 Amer, Fluß in Nordspanien, linker Nebenfluß des Ter 413
 Amiens, dép. Somme AO (†)355, 356, 565, 572
 – Bischöfe siehe Ragenar
 – Gau 342, 572
 – Grafen siehe Eckard, Engilwin, Ingelwin
 – Notre-Dame et Saint-Firmin 572
 Amolo, Erzbischof von Lyon (841-852) 273f., 277, 358, 448, 527, 561
 Ampoigné, dép. Mayenne, arr. und c^m Château-Gontier 606
Andiliaco (Andillé?, dép. Maine-et-Loire, arr. Angers, c^m Saint-Georges-sur-Loire, c^{nc} Savennières) 604
 Andorra (Les Vallées d'Andorre), Kleinstaat in den östl. Pyrenäen 352
 Andrea Dandolo siehe Dandolo, Andrea
 Angers, dép. Maine-et-Loire 357, 392f., 404, 498, 502, 534, 589
 – Bischöfe siehe Dodo, Licinius
 – Gau, Grafschaft siehe Anjou
 – Grafen siehe Odo
 – Saint-Aubin †575, †576, 540
 – – Äbte siehe Lambert
 – Sainte-Croix 404
 – Sainte-Geneviève 404
 – Saint-Étienne 404
 – Saint-Jean-et-Saint-Lézin 130, 542, †575, 588
 – – Äbte siehe Teutbold
 – Saint-Maurice 404, 487, 604, 606
 – Saint-Maurille 404
 – Saint-Serge 565, 589
 – – Äbte siehe Gairardus
 Angilbert, Gelehrter, Abt von Saint-Riquier (790-814) 108, 338, 363
 Angilbert, Teilnehmer der Schlacht von Fontenoy 216
 Angilguinus siehe Ingelwin
 Angoulême, dép. Charente 110, 237
 – Grafen siehe Turpion
 – Schlacht bei Angoulême (844) 358, 375, 403, 422, 429, 439-441, 444, 452, 504, 572
 Angoumois, Region um die Stadt Angoulême 486

Abbildung 2.2: Digitalisat eines Registers

Listing 2.3 Beispieleintrag eines Registers

```
<?xml version="1.0" encoding="UTF-8"?>
<document xmlns:telota="http://www.telota.de" telota:doctype="ri_13_register" source="A.rtf"
  time="2013-03-07 12:02:21">
  <body>
    <section page="Same">
      <Register>Friedrich Register - A</Register>
      <Stufe0 id="A000000001">
        <Inhalt>Aa, Johann von ~<Regestennummer>#7-678</Regestennummer></Inhalt>
        <Stufe1 id="A000000002">
          <Inhalt>Sophie von ~, Tochter Johanns, Bürgerin zu Köln
            <Regestennummer>#7-678,#7-680</Regestennummer></Inhalt>
        </Stufe1>
      </Stufe0>
    </section>
  </body>
</document>
```

Listing 2.3 zeigt einen Ausschnitt aus einem Register. Für die Visualisierung wurde das „Inhalt“-Tag ausgelesen, welches Namen der Entität enthält. In diesem Tag sind die Tags „Regestennummer“ und „RegestennummerFett“ enthalten. Diese werden ebenfalls extrahiert und ermöglichen das Verknüpfen von Registereinträgen, mit den entsprechenden Regesten.

Der in Listing 2.3 abgebildete Registereintrag verweist dabei auf das Register mit der Nummer 687, in Band 7 der Abteilung RI XIII - Friedrich III.

2.2 Webanwendung

Ziel dieser Arbeit ist es, die Online-Suche der Regesta Imperii zu ergänzen. Aus diesem Grund ist es nötig die Visualisierung als Webanwendung zu implementieren. Diese Webanwendung ist dabei aus zwei Teilen aufgebaut. Sie besteht dabei aus einem Server, der die aufbereiteten Daten zur Verfügung stellt und einem Client. Dieser stellt die Daten graphisch dar und ermöglicht dem Nutzer die Interaktion mit diesen Daten. Vorteilhaft ist in diesem Zusammenhang auch, dass auf das Online-Angebot der Regesta Imperii einfach verlinkt werden kann. Damit ist ein Arbeiten mit der Visualisierung und dem bestehenden Angebot der Regesta Imperii möglich.

In Abbildung 5.1 ist der Schematische Aufbau einer Webanwendung dargestellt. Dabei liefert der Server mit Hilfe von HTTP-Nachrichten Daten an den Client, die er aus der Datenbank lädt. Dies geschieht über Routen, also verschiedene Adressen an die Anfragen gestellt werden. Die Services des Clients nehmen diese Daten entgegen und bereiten diese für die Visualisierung auf. Bei Änderungen in der Visualisierung werden die Optionen an einen Service weitergegeben, der diese via HTTP an den Server weiterleitet. Der Server sucht dann in der Datenbank nach den Ergebnissen und schickt diese wieder mit Hilfe von HTTP an den Client.

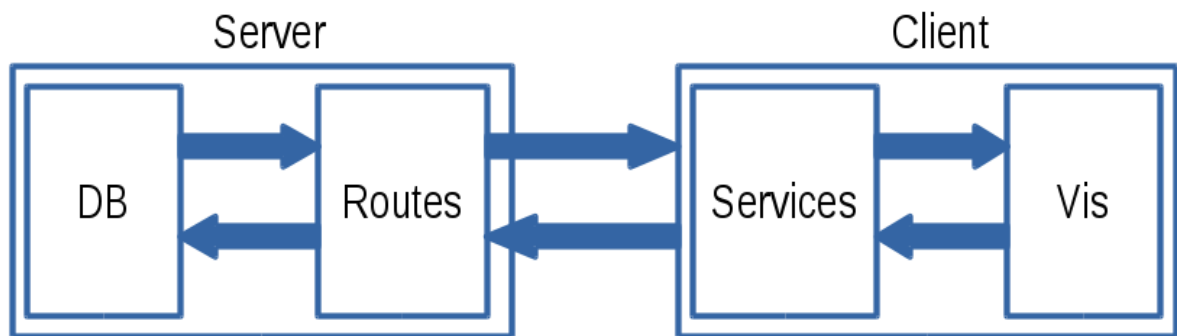


Abbildung 2.3: Schematischer Aufbau einer Web-Anwendung

Im Rahmen der Entwicklung sind weitere Besonderheiten von Webanwendungen zu beachten. Da die entwickelte Software in der Sandbox des Browsers ausgeführt wird ist beispielsweise der Zugriff auf das Dateisystem nur eingeschränkt möglich. Zusätzlich ist die Auswahl an Programmiersprachen und entsprechenden Bibliotheken und Frameworks eingeschränkt. JavaScript ist die Standardsprache für Entwicklung von Skripten für den Browser. Der Internet Explorer unterstützte in früheren Versionen auch VBScript, jedoch spielt diese Skriptsprache mittlerweile keine bedeutende Rolle in der Webentwicklung mehr. Andere Sprachen, die nach JavaScript transpiliert werden (TypeScript, PureScript, CoffeeScript, etc.) fügen eine weitere Abstraktionsschicht in der Entwicklung ein und gestalten den Umgang mit einigen Frameworks schwieriger, weshalb diese Sprachen für diese Arbeit nicht in Frage kamen. Dadurch, dass JavaScript für die Entwicklung von Webentwicklung der Quasi-Standard ist, können in der Entwicklung für diese Arbeit nur JavaScript-Bibliotheken oder -Frameworks verwendet werden. Beziehungsweise solche Bibliotheken, die eine Schnittstelle für JavaScript anbieten.

3 Verwandte Arbeiten und Technologien

Nachdem die Grundlagen dieser Arbeit erläutert wurden, werden in diesem Abschnitt verwandte Arbeiten vorgestellt, die ähnliche Probleme gelöst haben. Diese Ideen werden ergänzt durch eine kurze Vorstellung der Technologien, die für die Entwicklung der Software verwendet wurden.

3.1 Daten

Aus den von den Regesta Imperii zur Verfügung gestellten XML-Daten konnten die Ausstellungsorte und Ausstellungsdaten der Urkunden extrahiert werden. Zusätzlich wurden aus dem Register zu Friedrich III. Entitäten extrahiert die in den Urkunden erwähnt werden. Diese sind wie bereits beschrieben hierarchisch aufgebaut.

Shneiderman beschreibt in [Shn96] Datentypen in einer Data Type Taxonomy. Die Daten der Regesta Imperii enthalten dabei die folgenden Typen:

- Temporal
- 2-dimensional
- Hierarchical

Weiterhin beschreibt Shneiderman in [Shn96, S. 2] das Visual Information Seeking Mantra, dass auch im Rahmen dieser Arbeit eine bedeutende Rolle spielt: „Overview first, zoom and filter, then details-on-demand“.

3.2 Visualisierungsansätze

Die drei gegebenen Datentypen sollen im Rahmen dieser Arbeit visualisiert und miteinander verknüpft werden. In diesem Abschnitt werden verwandte Arbeiten vorgestellt die Lösungen für die Visualisierung dieser Datentypen bieten.

3.2.1 Temporale Daten

Zur Visualisierung temporaler Daten sind bereits einige Ansätze etabliert, jedoch sind viele nicht für den Einsatz als Filter geeignet, da sie entweder keine Selektion ermöglichen oder durch Visual Clutter den Nutzer verwirren. Barcharts, Linecharts und Scatterplots eignen sich um Zeitpunkte oder Zeiträume abzubilden. Diese Visualisierungen sind jedoch statisch und müssten so erweitert werden, dass eine Selektion von Datenpunkten möglich wäre. Im Folgenden Abschnitt werden Visualisierungen vorgestellt, die das Selektieren von Zeiträumen ermöglichen und deswegen für den Einsatz in dieser Arbeit geeignet sind.

Simile Timeline

Die im Rahmen von Simile ([Sim]) entwickelte Timeline bietet dem Nutzer die Möglichkeit einen Zeitraum auszuwählen und genauere Informationen zu diesem Zeitraum zu erhalten.

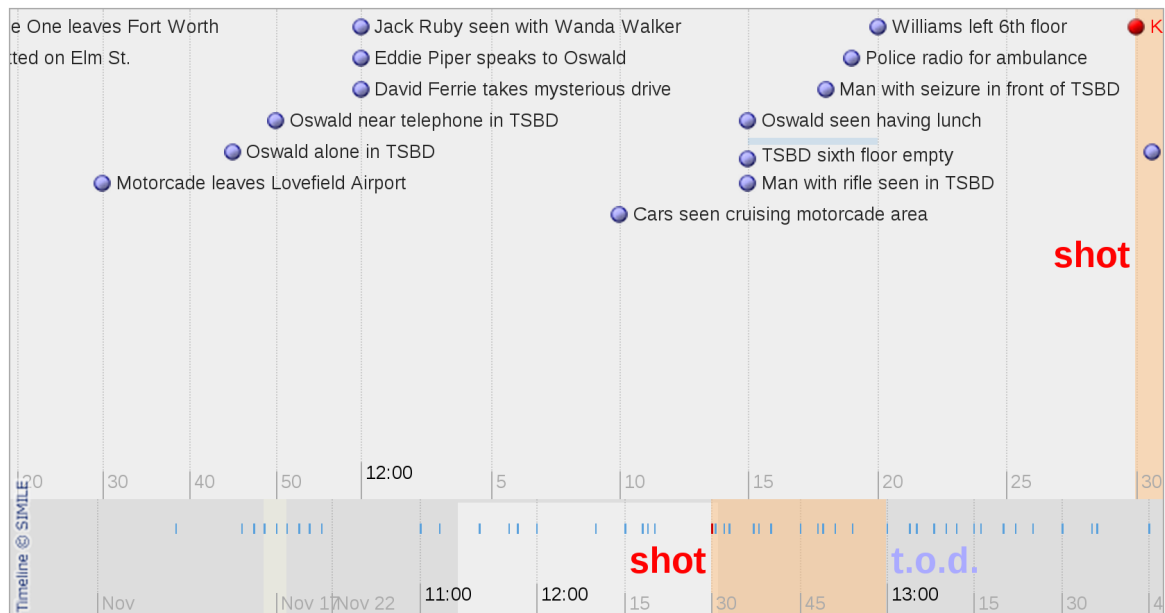


Abbildung 3.1: Simile Timeline Screenshot

Abbildung 3.1 zeigt einen Screenshot der Simile Timeline. Diese ist in zwei Bereiche eingeteilt. Im oberen Abschnitt werden die Datenpunkte auf einem Zeitstrahl abgebildet. Die Daten werden horizontal so abgetragen, dass sie an der Stelle platziert werden, an der das Ereignis stattgefunden hat. Vertikal werden die Punkte so verteilt, dass sich Texte nicht überlappen und so alle Beschreibungstexte jederzeit gut lesbar sind.

Der untere Abschnitt ermöglicht die Selektion eines kleinen Zeitabschnitts, dessen Ereignisse dann im oberen Abschnitt der Visualisierung genauer präsentiert werden. Dazu kann der Nutzer durch eine Klick- und Ziehbewegung den selektierten Bereich auf der unteren Zeitachse verschieben. Die

Zeitachse im oberen Bereich wird dann dynamisch aktualisiert. Der selektierte Bereich wird auf dem unteren Zeitstrahl hell hervorgehoben, während der nicht selektierte Bereich leicht abgedunkelt wird.

Eine Einschränkung, die das Verwenden der Simile Timeline für diese Arbeit ausschließt ist, dass die Größe des selektierten Bereichs nicht von Nutzer angepasst werden kann. Dadurch wäre der Nutzer nicht mehr in der Lage, wie bei der Expertensuche, beliebig große Zeiträume nach den gewünschten Regesten zu durchsuchen.

Ableton Live

Die von Ableton entwickelte Digital Audio Workstation Live bietet die Möglichkeit Teile eines Produzierten Stücks auf einer Art Timeline auszuwählen.

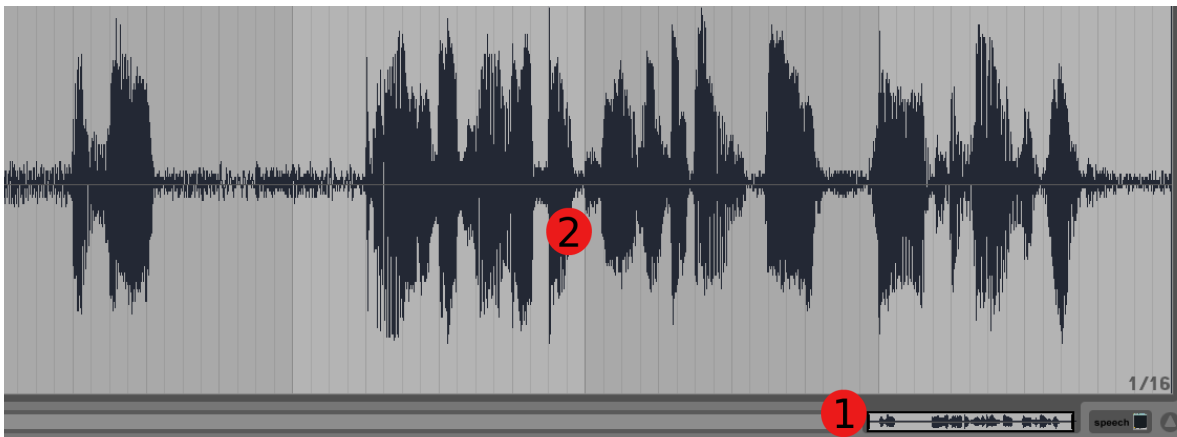


Abbildung 3.2: Ableton Live Screenshot

Der Screenshot in Abbildung 3.2 zeigt die Timeline, wie sie in Ableton Live 9 verwendet wird. Diese Timeline ermöglicht es dem Nutzer schnell in den Spuren eines Stückes zu navigieren. Im markierten Feld (Abbildung 3.2 (1)) kann der Nutzer einen Teil der Spur selektieren. Dieser selektierte Teil der Spur wird dann im oberen Teil der Oberfläche (Abbildung 3.2 (2)) detailliert dargestellt. In diesem View kann der Nutzer die Noten bearbeiten. Der selektierte Bereich kann bei Ableton Live nicht nur verschoben werden, auch die Größe des selektierten Bereichs lässt sich anpassen. Auf diese Weise ist es möglich, gezielt Anpassungen an einzelnen Noten durchzuführen oder beliebig große Bereiche einer Spur gleichzeitig zu bearbeiten.

Die Timeline von Ableton Live bietet dabei den Vorteil, dass der ausgewählte Zeitraum beliebig groß sein kann und nicht auf eine feste Länge beschränkt ist. Möchte der Nutzer jedoch einen sehr kleinen Bereich auswählen muss er sehr genau agieren, um keinen zu großen Bereich auszuwählen.

3.2.2 Spatial Data

Als einfachste Art der Visualisierung von geographischen Daten bietet es sich an Marker auf einer Karte zu setzen. Dieser Ansatz bringt jedoch einige Probleme mit sich. Viele Marker führen zu Visual Clutter und es wird schwer einen Überblick über die Daten zu erhalten. Gleichzeitig werden, wenn viele Datenpunkte an einer Stelle liegen, mehrere Marker auf eine Stelle gesetzt und die Auswahl des Datensatzes wird erschwert.

Heatmap

Eine Heatmap löst dabei das Problem mit Visual Clutter. Die in Abbildung 3.3 dargestellte Heatmap, welche über eine Karte gelegt wurde kombiniert mehrere Datensätze auf einem Punkt und färbt diese Stelle rot ein. Implementiert wurde diese Heatmap mit HeatmapJS ¹.

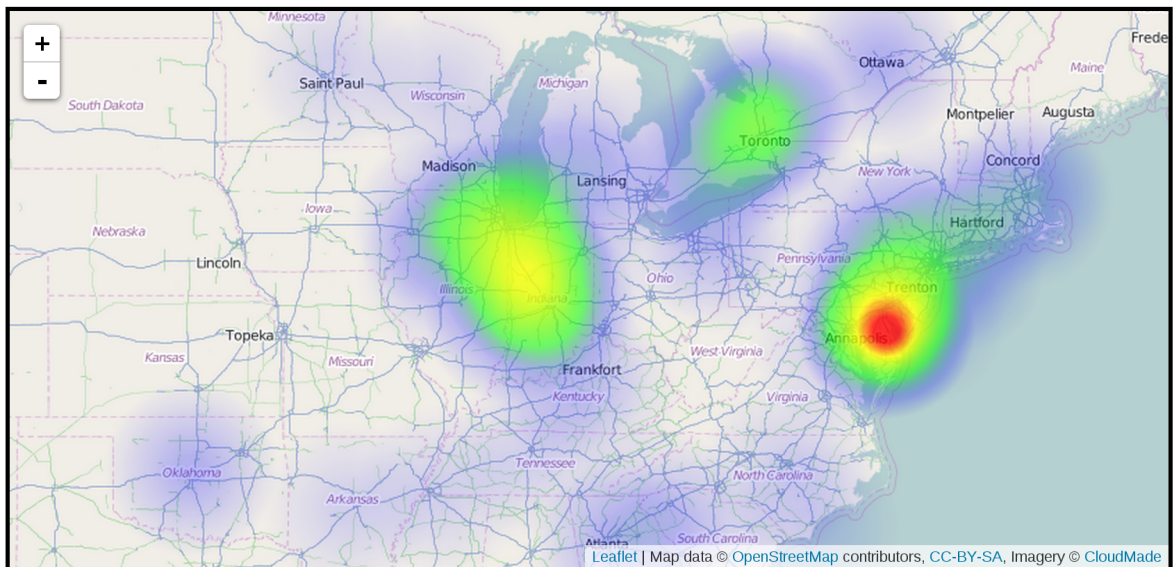


Abbildung 3.3: Heatmap über Open Street Map

Der Vorteil Visual Clutter zu vermeiden bringt jedoch auch einen erheblichen Nachteil mit sich. Eine Auswahl kann bei einer Heatmap nur durch eine Fläche getroffen werden, da keine einzelnen Orte auf einer Karte markiert werden. Gerade bei den Ausstellungsorten handelt es sich aber um diskrete Punkte, wofür eine Heatmap nicht geeignet ist.

¹<http://www.patrick-wied.at/static/heatmapjs/>

Bubble Glyphs

Eine Alternative bieten Bubble Glyphs, wenn sie über eine Karte gelegt werden. Abbildung 3.4 zeigt eine Umsetzung mit D3.

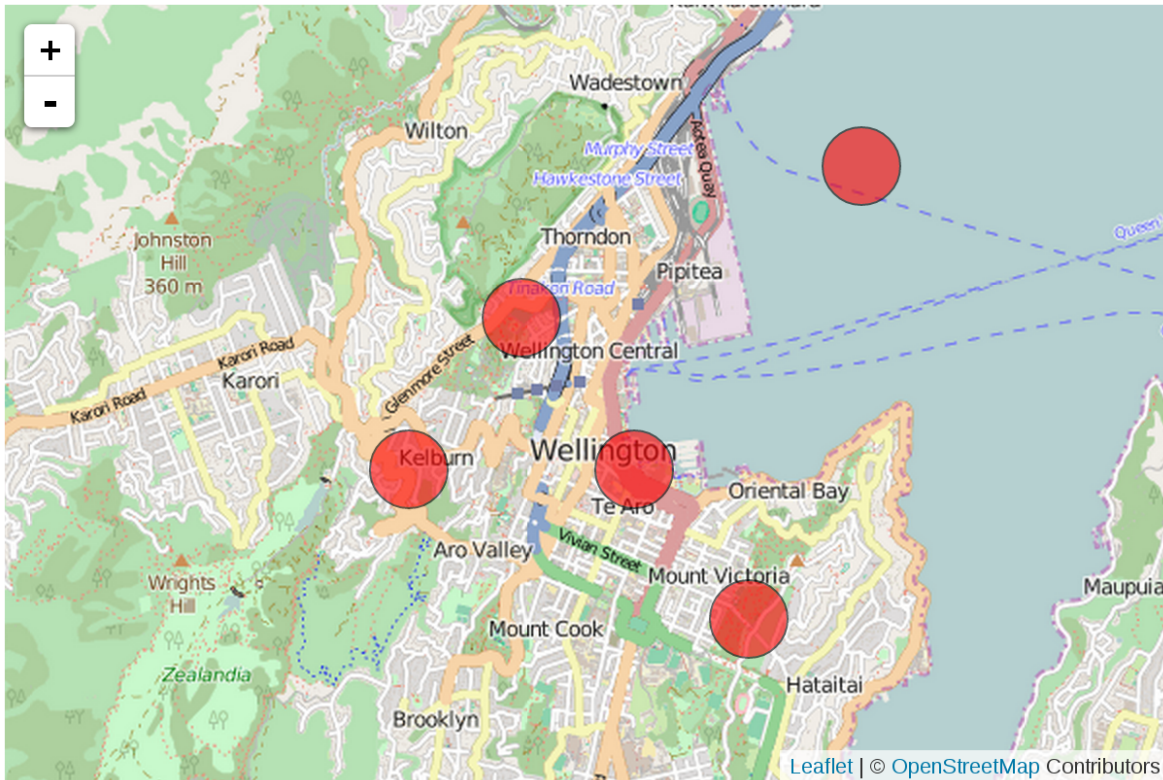


Abbildung 3.4: Bubble Glyphs über Open Street Map

Auf einzelne, diskrete Orte werden dabei Kreise gelegt. Diese sind in der Hinsicht genau wie Marker, haben aber einen erheblichen Vorteil. Der Radius oder die Fläche des Kreises können eine weitere Eigenschaft des Datenpunkts abbilden. In [JHSS12] wird diese Idee um Circle Packing erweitert. Circle Packing ist jedoch ungeeignet für die Anwendung in dieser Arbeit, da es die Auswahl einzelner Orte verhindert.

3.2.3 Hierarchical Data

Die Registereinträge sind alphabetisch sortiert. Für die Visualisierung kann dieser Vorteil genutzt werden, um die Benutzung einfacher zu gestalten. Ursprünglich war es nicht klar, ob die hierarchische Struktur beim Extrahieren der Daten sinnvoll erhalten werden kann, weshalb auch Visualisierungen betrachtet wurden, die keine hierarchische Struktur darstellten.

Alphaslider

Da das Register eine große Menge an Einträgen enthält (49856) ist darauf zu achten, dass Visual Clutter vermieden wird. Gleichzeitig kann es dadurch, dass die Software im Browser ausgeführt wird zu Performance-Problemen kommen, die größtenteils ausgeschlossen werden müssen.

Eine Möglichkeit möglichst wenige Einträge gleichzeitig anzuzeigen bietet der in [AS94] vorgestellte Alphaslider.



Abbildung 3.5: Alphaslider [AT95]

Abbildung 3.5 zeigt einen Screenshot des Alphaslider. Dieser bietet wie andere Slider eine Schaltfläche die vom Nutzer innerhalb eines gewissen Rahmens frei bewegt werden kann. Der Alphaslider in Abbildung 3.5 ermöglicht die Auswahl von Filmtiteln. Dabei sind die Filmtitel alphabetisch sortiert auf der horizontalen Achse abgetragen. Der Bereich ab dem die Filme mit einem neuen Buchstaben beginnen ist durch den entsprechenden Buchstaben markiert. Dadurch ist es möglich schnell einen Überblick über die Verteilung der Filme zu erhalten. Durch das Bewegen der Schaltfläche kann ein Film ausgewählt werden, der Titel des ausgewählten Films wird dabei über dem Alphaslider dargestellt.

Zu den Vorteilen des Alphasliders gehören vor allem der geringe Platzverbrauch und das auch unerfahrene Nutzer, wie in [OLS93] festgestellt, schnell zu den gewünschten Ergebnissen gelangen. Der Alphaslider bringt jedoch auch Nachteile mit sich. Die maximale Anzahl an Einträgen, die der Alphaslider darstellen kann ist durch die Größe des Sliders begrenzt, da dieser mindestens in Ein-Pixel-Schritten verschoben werden muss. Weiterhin bietet der Alphaslider nicht die Möglichkeit mehrere Einträge auszuwählen. Auch Hierarchien können nicht mit dem Alphaslider dargestellt werden, weshalb er für die Datenstruktur des Registers nicht geeignet ist.

LensBar

Die LensBar löst das Problem, dass nur ein Eintrag ausgewählt werden kann, da mehrere Einträge gleichzeitig dargestellt werden. Abbildung 3.6 zeigt einen Screenshot der LensBar, die in [Mas98] vorgestellt wurde.

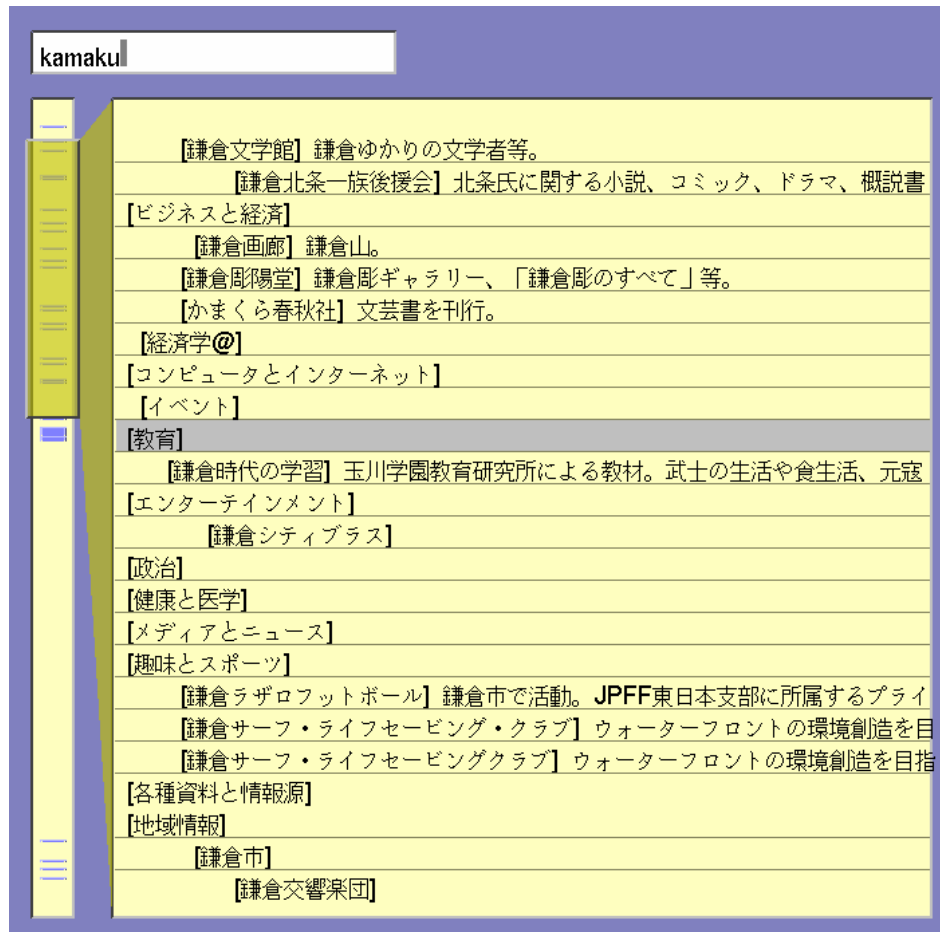


Abbildung 3.6: Lensbar [len]

Bei der LensBar wird, wie in Abbildung 3.6 dargestellt, ein Slider mit einer Liste kombiniert. Mit dem Slider ist es möglich, wie beim Alphaslider eine Liste zu durchsuchen. Mit Schaltfläche, die im Slider verschoben wird, selektiert der Nutzer jetzt jedoch keine einzelnes Element mehr, sondern mehrere Einträge die direkt hintereinander liegen. Diese Einträge werden dann als Liste neben dem Slider dargestellt. Um schnell nach einem Eintrag suchen zu können steht dem Nutzer ein Suchfeld zur Verfügung. Suchergebnisse die zur ein Eingabe in dieses Feld passen werden im Slider markiert. Der genaue Suchalgorithmus wurde in [Mas98] genauer erläutert.

Die Lensbar ermöglicht das einfache Selektieren mehrerer Einträge, ist jedoch auch nicht in der Lage eine Hierarchie darzustellen. Das Suchfeld unterstützt jedoch den Nutzer dabei, schnell die gewünschten Einträge zu finden.

LensList

Die LensList ist eine Möglichkeit große Listen auf kleinem Raum darzustellen. Vorgestellt wurde sie ursprünglich in [SQT07].



Abbildung 3.7: LensList [fis]

Abbildung 3.7 zeigt einen Screenshot der LensList. In diesem Fall werden alle Einträge der Liste alphabetisch sortiert untereinander dargestellt. Diese werden jedoch so skaliert, dass alle Einträge sichtbar sind. Dadurch ist es nicht mehr möglich die Einträge zu lesen. Die Ansicht ermöglicht es aber dem Nutzer einen Überblick über die Daten zu erhalten. Wie in Abbildung 3.7 zu erkennen, werden die mit der Maus selektierten Einträge und die in ihrer näheren Umgebung größer dargestellt. Durch diese Verwendung des Fisheye-Zooms ist es möglich die Einträge zu lesen. Gleichzeitig erhält der Nutzer einen Überblick darüber, in welchem Bereich der Liste sich der selektierte Eintrag befindet.

Die LensList bietet die Möglichkeit viele Einträge auf kleinem Raum darzustellen. Außerdem ist es mit ihr möglich mehrere Einträge zu selektieren. Mit der LensList ist es jedoch immer noch nicht

möglich Hierarchien darzustellen. Aus diesem Grund ist sie für die Verwendung in dieser Arbeit nicht geeignet.

Indented Tree

Der als Block ² vorgestellte Collapsible Indented Tree stellt hierarchische Daten durch das Einrücken von Rechtecken dar, die Stufen einer Hierarchie darstellen.

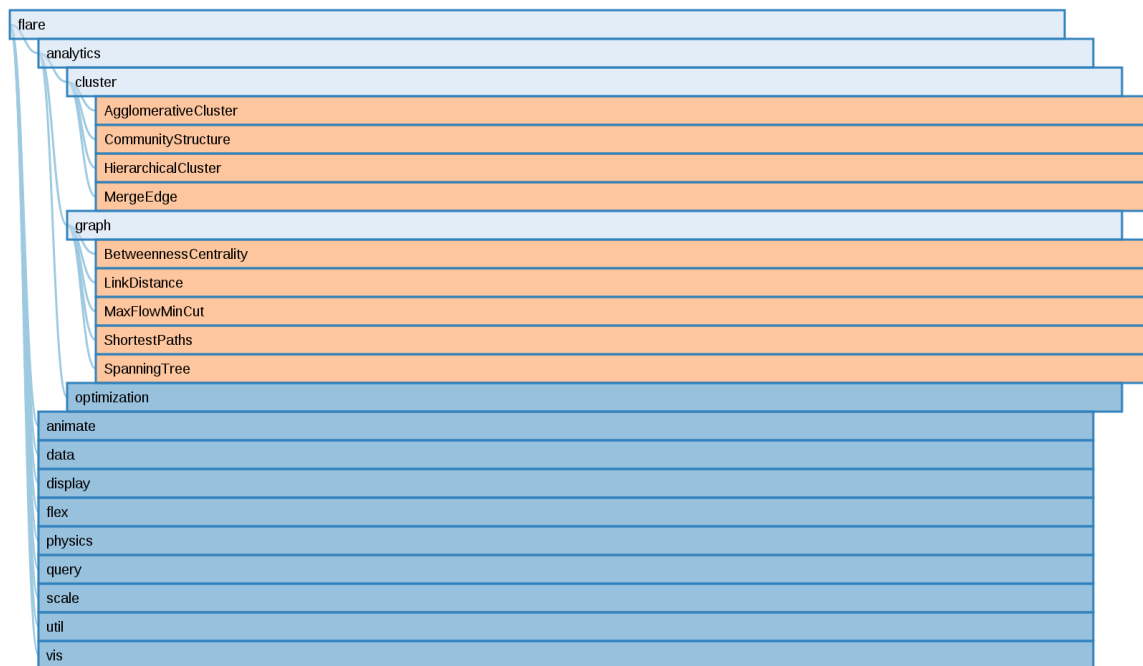


Abbildung 3.8: Indented Tree

Abbildung 3.8 zeigt einen Screenshot des Collapsible Indented Trees. Er löst dabei das Problem durch Visual Clutter, da Einträge eingeklappt werden können. Gleichzeitig erlaubt er die Hierarchie des Registers darzustellen und kann zusätzlich durch Farbcodierung weitere Informationen visualisieren. Ein Nachteil des Collapsible Indented Trees ist es jedoch, dass der Baum bei vielen Einträgen der Lang wird und die Suche nach einem Element dadurch umständlich wird. Für diese Arbeit ist der Indented Tree jedoch geeignet, da er die Hierarchie des Registers darstellen kann und das Selektieren von Einträgen ermöglicht.

²<http://bl.ocks.org/mbostock/1093025>

3.3 Brushing und Linking

Verknüpfte Visualisierungen können den Informationsgehalt einer Anwendung weiter erhöhen. Eine Möglichkeit diese Verknüpfung zu realisieren bieten die in [BMMS91] vorgestellten Techniken Brushing und Linking. Brushing beschreibt dabei das Selektieren und Hervorheben von Datenpunkten in einer Visualisierung. Führt Brushing in einem View auch zur Hervorhebung oder sonstigen Veränderungen in anderen Views wird dies als Linking bezeichnet.

Christchurch 2010 Timeline

Die Christchurch 2010 Timeline ist eine mit D3 entwickelte Visualisierung, welche inklusive Quellcode frei im Internet ³ verfügbar ist.

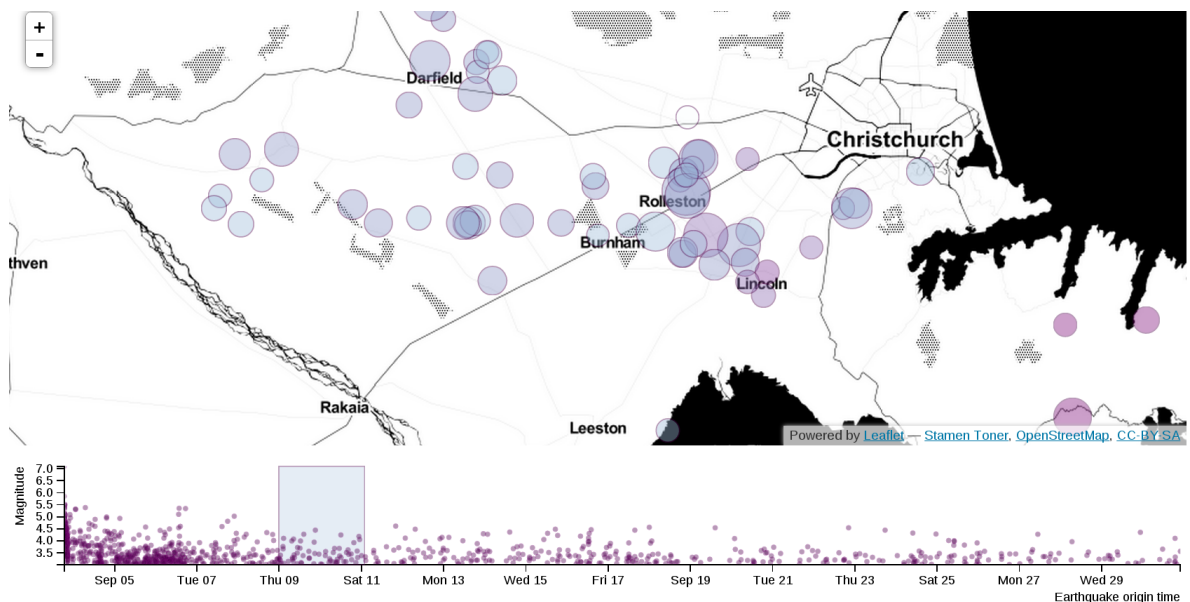


Abbildung 3.9: Screenshot der Christchurch 2010 Timeline

Abbildung 3.9 zeigt einen Screenshot der Timeline. Die Christchurch Timeline visualisiert die Erdbeben, die 2010 in Christchurch gemessen wurden. Der Ort des Bebens wird dafür auf der Karte mit einem Kreis markiert. Die Größe des Kreises spiegelt dabei die Stärke des Erdbebens wieder. In der Timeline unterhalb der Karte werden die Zeitpunkte markiert, an denen Beben stattfanden.

Sobald eine Zeitspanne in der Timeline ausgewählt wurde werden auf der Karte die Erdbeben dargestellt, die im selektierten Zeitraum liegen. Auf diese Weise wurde durch Brushing in der Timeline die Kartenansicht aktualisiert.

³<http://bl.ocks.org/tnightingale/4718717>

Einen ähnlichen Ansatz verfolgt die in [JHSS12] vorgestellte Visualisierungsm bei der Karte und Timeline mittels Brushing und Linking miteinander verknüpft sind. Die Christchurch Timeline nutzt jedoch im Gegensatz zu diesem Tool kein Circle Packing, sondern legt die Kreise übereinander.

3.4 Verwandte Probleme

Im Rahmen der Digital Humanities werden immer neue Forschungsfragen gestellt und diese mit Hilfe von Software gelöst. Auch Visualisierungen werden verwendet, um neue Erkenntnisse in den Geisteswissenschaften zu erlangen. Im folgenden Abschnitt werden deshalb Arbeiten vorgestellt, die Visualisierungen nutzen, um die Geisteswissenschaften zu unterstützen.

3.4.1 Jigsaw

Bei Jigsaw handelt sich um eine Software zur Analyse und Exploration von Textsammlungen, die in [GLP⁺07] vorgestellt wurde. Dabei werden dem Nutzer Visualisierungen zur Verfügung gestellt, die ihm helfen sollen den Inhalt des Textes besser zu erfassen. Diese Visualisierungen sind jedoch unabhängig und nicht miteinander verknüpft. Jigsaw bietet dabei die folgenden Views an:

- Document View
- List View
- Cluster View
- Graph View
- Grid View
- Calendar View
- WordTree View
- Scatterplot View
- Circular Graph View
- Timeline View

Diese Views ermöglichen es unterschiedliche Eigenschaften der geladenen Texte zu explorieren. Auf der Projektwebseite ⁴ werden dabei auch Tutorialvideos und Dokumentation angeboten.

Jigsaw löst in der Hinsicht ein verwandtes Problem, dass Visualisierungen genutzt werden, um Forschungsfragen der Geisteswissenschaften zu beantworten. In diesem Fall die Fragen der Literaturwissenschaftler. Dazu nutzt Jigsaw Visualisierungen, um die Informationen für den Nutzer aufzubereiten. Alle Visualisierungen wurden dabei so gestaltet, dass kein Fachwissen benötigt wird

⁴<http://www.cc.gatech.edu/gvu/ii/jigsaw/tutorial/>

3 Verwandte Arbeiten und Technologien

um diese zu bedienen. Außerdem sind alle Features dokumentiert. Dadurch ist es auch dem Laien möglich alle Funktionen des Tools zu nutzen.

3.4.2 DARIAH-DE Geo-Browser

Der DARIAH-DE Geo-Browser ist ein Tool zur Analyse von Raum-Zeit-Relationen und steht Online auf der Projektwebseite ⁵ zur Verfügung. Ziel des Geo-Browsers ist die Unterstützung von Forschern der Geistes- und Kulturwissenschaften. Er besteht aus drei Views von denen zwei in Abbildung 3.10 dargestellt werden.

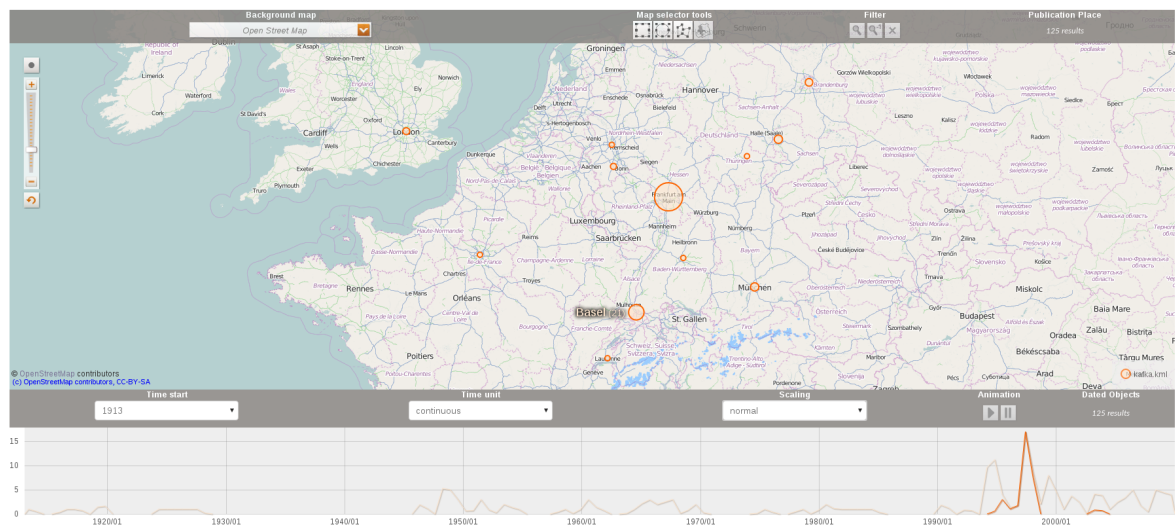


Abbildung 3.10: DARIAH-DE Geo-Browser

Der DARIAH-DE Geo-Browser zeigt auf einer Karte Vorkommen von Texten an und verlinkt diese mit temporalen Informationen, die auf einer Timeline darunter dargestellt werden. Unter der Timeline werden die Titel der Texte, sowie die Veröffentlichungsorte der Texte in einer Tabelle abgetragen. Der Geo-Browser unterstützt Brushing und Linking, in dem Karte und Timeline miteinander verknüpft sind und jeweils den selektierten Bereich in der anderen Ansicht farblich hervor heben. Um zeitliche Verläufe genauer zu Untersuchen ist es möglich, einen Zeitabschnitt in der Timeline zu selektieren und durch eine Animation den zeitlichen Verlauf auf der Karte zu betrachten. Auf diese Weise ist es möglich eine Relation zwischen der zeitlichen Entwicklung und dem Veröffentlichungsort der Texte zu erkennen.

Die Verknüpfung zwischen temporalen und zweidimensionalen Daten ist für diese Arbeit interessant und kann für die Nutzung mit den Daten der Regesta Imperii angepasst werden. Ergänzt werden muss diese jedoch, um die Möglichkeit Einträge des Registers zu selektieren, um auch nach diesen Filtern zu können. Der DARIAH-DE Geo-Browser ist nicht als Alternative zu einer Suchfunktion entwickelt

⁵<https://de.dariah.eu/geobrowser>

worden und verfolgt demnach ein anderes Ziel als diese Arbeit. Der Ansatz des Geo-Browsers kann jedoch auf diese Arbeit übertragen werden.

3.5 Multi Coordinated View

Im Rahmen dieser Arbeit wird ein Software entwickelt die Suche der Regesta Imperii durch einen Multi-Coordinated-View-Ansatz ergänzt. In [WBWK00] werden Multiple-View-Systeme dadurch definiert, dass sie zwei oder mehr verschiedene Ansichten verwenden um die Erkundung einer einzelnen Entität zu unterstützen.

Scherr [Sch08] teilt diese Systeme in Kategorien ein und beschreibt Möglichkeiten der Verknüpfung von Ansichten. Dabei geht er auch auf die in dieser Arbeit verwendeten Techniken Brushing und Linking ein. Beim Brushing werden bei Auswahl von Elementen in einer Ansicht auch relatierte Elemente in anderen Ansichten hervorgehoben. Auf diese Art können Elemente aus unterschiedlichen Blickwinkeln betrachtet werden und so neue Einblicke in einen Datensatz gewonnen werden. Linking bezieht sich Änderungen der Parameter einer Ansicht, die dann die Parameter eine anderen Ansicht beeinflussen. Beispielsweise führt das Scrollen in einer Liste dazu, dass die daneben angezeigte, verknüpfte Liste auch gescrollt wird.

In einem „State of the Art“ geht Roberts [Rob07] auf Fortschritte in der Forschung an MCV ein. Dabei vergleicht er verschiedene Aspekte von der Verarbeitung und Vorbereitung der Daten bis zu Usability und Wahrnehmung durch den Nutzer. Dabei kommt er zu der Erkenntnis, dass die Forschung schon weit vorangeschritten ist und eine der großen Schwierigkeiten darin besteht, alle existierenden Möglichkeiten gezielt umzusetzen und miteinander zu verknüpfen.

Das in [KBGE09] vorgestellte Tool PatViz soll als Beispiel für die Umsetzung von MCV dienen. PatViz ist ein Tool zur Suche in und Exploration von Patenten.



Abbildung 3.11: PatViz

Abbildung 3.11 zeigt einen Screenshot von PatViz. Wie dort zu erkennen ist das Tool aus zehn verschiedenen Views aufgebaut, die miteinander verknüpft sind.

In Abbildung 3.11 werden dabei von links oben nach rechts unten die folgenden Views angezeigt:

- Weltkarte - Verteilung von Patenten über verschiedene Länder
- Treemap - Verteilung der Urkunden über ein Klassifikationsschema
- Tabelle - Zusammenfassung der wichtigsten Informationen der selektierten Patente
- Patent View - Graphansicht, die verschiedene Relationen zwischen Patenten darstellt
- Selection Management - Graphbasiertes Tool zum Erstellen von Selektionen
- 3D-Treemap - Wie Treemap nur in 3D
- Geo-Timeline - Scatterplot, der Einreichungsdatum und Ort der Einreichung von Patenten darstellt
- Term Cloud - Word Cloud mit den in Patenten am häufigsten erwähnten Begriffen
- Barcharts - Barcharts über ein auswählbares Metadatum
- Text View - Zeigt den Text des Patents an und markiert Ergebnisse der linguistischen Analyse

Diese Views sind miteinander mittels Brushing und Linking verbunden. Selektiert der Nutzer beispielsweise auf der Weltkarte ein Land werden die Patente aus dem selektierten Land in allen anderen Ansichten farblich hervorgehoben. Diese Verknüpfung ist bei Selektion in allen Views möglich und ermöglicht dadurch das Betrachten des selben Datensatzes aus unterschiedlichen Perspektiven.

Mit PatViz ist es möglich iterativ nach Patenten zu suchen, dazu können in den verschiedenen Views Eigenschaften der Patente untersucht und miteinander in Verbindung gebracht werden. Durch den Selection-Management-View ist es möglich die Filterung der Patente iterativ anzupassen und so uninteressante Patente auszublenden. Dieser Ansatz ist gerade auch im Bezug auf die Regesta Imperii interessant, da derzeit eine komplett neue Suche durchgeführt werden muss, wenn das Suchergebnis nicht zufriedenstellend ist. Eine einfache Anpassung der Suchparameter wie in PatViz würde eine große Zeitersparnis und eine intuitive Arbeitsweise erlauben.

3.6 Verwendete Technologien

Im Rahmen der Entwicklung wurden diverse Technologien verwendet. Für die gesamte Software wurde JavaScript als Programmiersprache gewählt, vor allem um einen weitverbreiteten Workflow nutzen zu können. Dieser wird im Folgenden vorgestellt.

Der Server wurde mit NodeJS ⁶ entwickelt und verwendet ExpressJS ⁷, um einen Webserver mit einer RESTful JSON API zu implementieren. NodeJS ist eine Laufzeitumgebung für JavaScript, die

⁶<https://nodejs.org/>

⁷<http://expressjs.com/>

auf der V8 JavaScript Engine von Google basiert. Diese ermöglicht es eigenständige Applikationen mit JavaScript zu entwickeln, für die kein Webbrowser benötigt wird. Zusätzlich existiert der Node Package Manager (npm). Dieser ermöglicht es benötigte Bibliotheken und Frameworks auf einfache Weise zu installieren und in einer speziellen Datei (package.json) zu dokumentieren. ExpressJS ist ein Webframework, das Hilfsfunktionen für HTTP bereitstellt, um APIs zu entwickeln. Mit Hilfe von ExpressJS ist es demnach möglich, RESTful Webservices zu entwickeln. In [Fie00] werden die Bedingungen für RESTful Webservices beschrieben:

- Es existiert eine Trennung zwischen Client und Server
- Der Service ist zustandslos. Der Kontext des Clients wird demnach zwischen Anfragen nicht zwischengespeichert.
- Caching ist möglich. Daten die sich nicht ändern, werden markiert und können vom Client zwischengespeichert werden.
- Es existiert eine Schnittstelle in der Ressourcen und Operationen einheitlich beschrieben werden. Beispielsweise mit URIs und HTTP-Methoden.
- Der Service ist mehrschichtig aufgebaut. Ein Client kann folglich nicht unterscheiden, ob er direkt mit dem Server verbunden ist oder etwa zuerst mit einen Load Balancer kommuniziert.

Beider Entwicklung der Visualisierung wurde eine RESTful Web-API implementiert, welche über HTTP kommuniziert und Daten im Body der HTTP-Nachrichten in Form von JSON (JavaScript Object Notation) ⁸ überträgt. Zusätzlich wird über die gleiche Serverinstanz der Client ausgeliefert.

Als Datenbanksystem wurde MongoDB ⁹ gewählt. Bei MongoDB handelt es sich um eine dokumentenbasierte Datenbank. Objekte werden dabei nicht mehr in relationale Strukturen aufgeteilt und gespeichert, sondern als „Dokument“ direkt in der Datenbank abgelegt und mit einer eindeutigen ID versehen. Das BSON-Format, das MongoDB intern nutzt um Objekte zu speichern, basiert dabei auf JSON. Das ermöglicht es JSON-Objekte direkt in MongoDB zu speichern und wieder aus ihr zu laden. Dies ist ein Vorteil, da JavaScript auch für die Entwicklung des Servers gewählt wurde.

Im Rahmen dieser Arbeit wurde demnach ein angepasster MEAN-Stack ¹⁰ (MongoDB, ExpressJS, AngularJS, NodeJS) verwendet. Dieser Stack hat sich für die Entwicklung von Webanwendungen als geeignet erwiesen und wurde auch in dieser Arbeit erfolgreich angewandt. Einzig AngularJS ¹¹, ein Framework zur Entwicklung von Single Page Apps, also Apps die nur aus einer HTML-Datei bestehen in die „Views“ geladen werden, wurde nicht verwendet, da derzeit keine Anforderungen existieren, die die Entwicklung einer SPA verlangen.

Der Client ist wie für Webanwendungen typisch mit HTML, CSS und JavaScript entwickelt worden. Als CSS-Framework wurde Twitter Bootstrap ¹² gewählt, da es durch das „Grid-System“ eine einfache Anordnung der Komponenten ermöglicht. Dabei kann das Layout der Anwendung durch CSS-Klassen

⁸<http://www.json.org/>

⁹<https://www.mongodb.org/>

¹⁰<http://mean.io/>

¹¹<https://angularjs.org/>

¹²<http://getbootstrap.com/>

3 Verwandte Arbeiten und Technologien

festgelegt werden. Ein Layout besteht aus einem Container, der in Zeilen und diese wieder in Spalten aufgeteilt sind. Durch unterschiedliche Klassen ist es möglich, das Layout auf die Größe des Bildschirms des Clients anzupassen.

Die Visualisierungen bestehen aus Scalable Vector Graphics, die in JavaScript, mit dem in [BOH11] vorgestellten Framework D3 entwickelt wurden. D3 ermöglicht es, wie der Name Data Driven Documents es schon beschreibt, DOM-Elemente an Daten zu koppeln und daraus automatisiert Dokumente zu erstellen. In dieser Arbeit wurden alle graphischen Views mit D3 umgesetzt, indem die vom Server gelieferten Daten an Elemente in Vektorgraphiken gebunden wurden. Auf diese Weise wurde je nach gelieferten Daten, eine entsprechende Visualisierung automatisch erstellt.

Um eine Karte darzustellen wurde Leaflet ¹³ verwendet. Dabei handelt es sich um eine JavaScript Bibliothek, die es ermöglicht Karten mit Hilfe von Open Street Map ¹⁴ in HTML-Dateien einzubinden.

¹³<http://leafletjs.com/>

¹⁴<https://www.openstreetmap.org>

4 Lösungsansatz

Ziel der Arbeit ist es eine Visualisierung zu entwickeln, die die Arbeit mit den Regesta Imperii vereinfacht. Die Suche soll dabei graphisch unterstützt werden. Die Expertensuche der Regesta Imperii wird deswegen graphisch interpretiert. In der entwickelten Suche ist es möglich, nach Urkunden zu suchen die gewissen Kriterien erfüllen. Folgende Kriterien lassen sich dabei einschränken:

- die Zeitspanne, in der eine Urkunde ausgestellt wurde
- der Ort, an dem eine Urkunde ausgestellt wurde
- Personen oder Orte, die in einer Urkunde erwähnt werden

4.1 Visualisierung

Um jederzeit einen Überblick über die Suchanfrage zu haben, wurde für jedes Kriterium ein View angelegt. Abbildung 4.1 zeigt einen Screenshot des Ansatzes, welcher aus fünf verschiedenen Ansichten besteht. Diese Ansichten werden in den folgenden Abschnitten genauer vorgestellt.

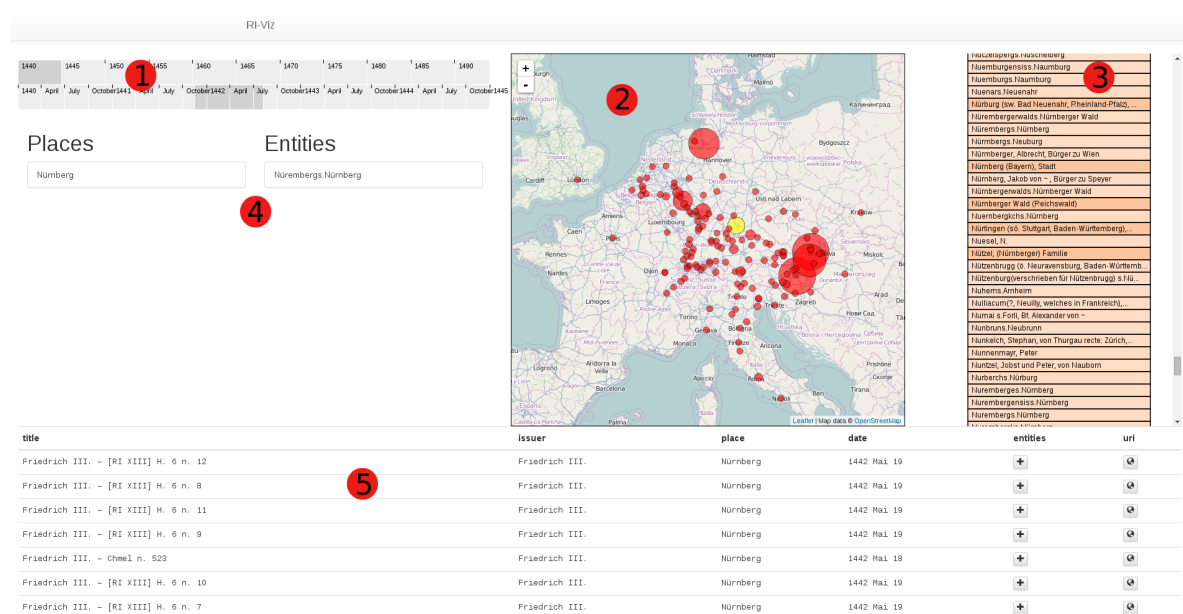


Abbildung 4.1: Screenshot des Visualisierungsansatzes

4 Lösungsansatz

Ansicht (1) ermöglicht dem Nutzer, das Auswählen einer Zeitspanne, in der das Ausstellungsdatum einer Urkunde liegen soll.

Ansicht (2) erlaubt dem Nutzer auf der Karte Orte auszuwählen, an denen eine Urkunde ausgestellt wurde.

Mit Hilfe von **Ansicht (3)** kann der Nutzer Urkunden filtern, in denen eine Person oder ein Ort erwähnt wird.

Die beiden Spalten in **Ansicht (4)** geben eine Übersicht über die ausgewählten Suchkriterien. In dieser View können sie auch schnell wieder abgewählt werden.

In **Ansicht (5)** wird anschließend das Ergebnis der Suche dargestellt. Also genau die Regesten die allen in den anderen Views definierten Suchkriterien entsprechen.

4.1.1 Timeline

Ansicht (1) ermöglicht das Filtern von Urkunden anhand des Ausstellungsdatums. Durch den Nutzer werden das früheste und späteste Ausstellungsdatum ausgewählt, in dem die gesuchten Urkunden liegen sollen. Ermöglicht wird dies über einen zweiteiligen Brush. Der obere Brush ermöglicht dabei eine grobe Filterung. Er bildet die komplette Zeit ab, zu der Urkunden von Friedrich III. existieren. Sobald in diesem Brush eine Auswahl getroffen wurde, wird der untere Brush aktualisiert. Dieser bildet nun die Auswahl des oberen Brushes ab und ermöglicht eine feinere Filterung des Ausstellungszeitraums.

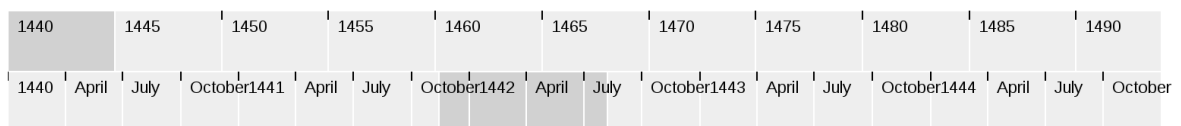


Abbildung 4.2: Timeline

In Abbildung 4.2 ist eine Auswahl mit dem Brush exemplarisch dargestellt. Im oberen Brush ist der Zeitraum von 1440 bis 1445 ausgewählt. Start- und Enddatum dieser Auswahl sind nun die Grenzen des unteren Brushes. Im unteren Brush wurde nun ein genauerer Zeitrahmen ausgewählt.

4.1.2 Karte

Ansicht (2) besteht aus einer Karte, auf der alle bekannten Ausstellungsorte von Urkunden Friedrich III. abgetragen sind. Die Größe der Markierung ist dabei abhängig von der Anzahl, der an diesem Ort veröffentlichten Regesten. Das heißt, je mehr Regesten an einem Ort veröffentlicht wurden, desto mehr wird dieser Ort hervorgehoben. Dies soll dem Nutzer ermöglichen interessante Orte schnell zu finden und diese Orte genauer betrachten zu können.

In Abbildung 4.3 ist dies noch einmal dargestellt.

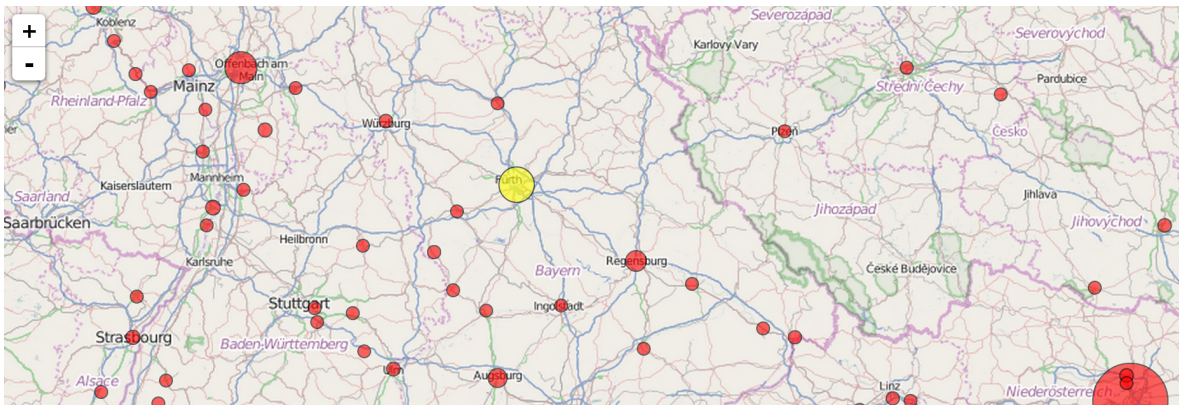


Abbildung 4.3: Karte

Durch klicken auf einer der Orte werden diese farblich hervorgehoben und für den Filter selektiert. Ausstellungsorte werden dabei „verodert“. Damit ist gemeint, dass das Ergebnis der Suchanfrage genau aus den Regesten besteht, die einen der ausgewählten Orte als Ausstellungsort haben.

Da in der Übersicht einige Markierung sehr groß im Verhältnis zu anderen sind, bietet die Karte einen geometrischen Zoom, der die Größe der Markierung inklusive der Karte anpasst. Auf diese Weise ist es möglich zwischen Markierungen zu differenzieren. Außerdem werden kleinere Markierungen über größere gelegt, um diese jederzeit markieren zu können.

Die Namen der Städte werden in einem Tooltip angezeigt, sobald der Nutzer mit der Maus über einen der Kreise fährt. Dabei wird der am häufigsten verwendete Name angezeigt, um eine möglichst sinnvolle Zuordnung zu erhalten. Dies wird gemacht, da teilweise viele alternative Namen zu einem Ort verwendet werden und deswegen das Anzeigen aller Namen zu Visual Clutter führen würde.

4.1.3 Register

Das bereits existierende Register der Regesta Imperii wird in **Ansicht (3)** mit dem Collapsible Indented Tree visualisiert. Die hierarchische Struktur des Registers und die alphabetische Sortierung bleiben dabei erhalten.

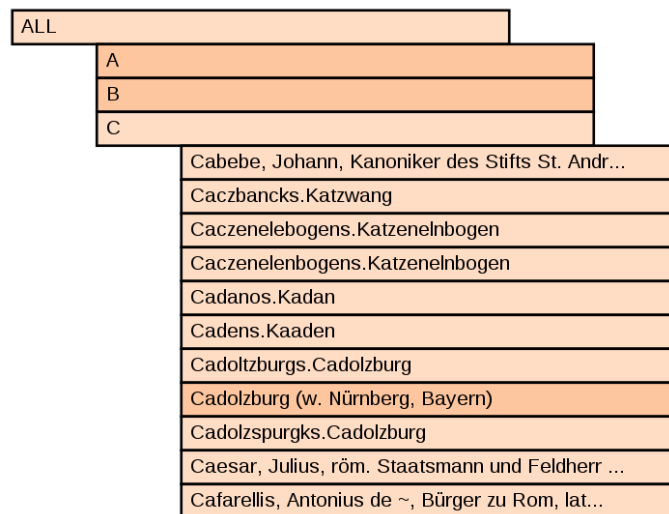


Abbildung 4.4: Register

Abbildung 4.4 zeigt einen Screenshot des Registers. Der Nutzer kann durch Klicken die Hierarchie des Registers weiter aufklappen, um so tiefere Stufen zu erkunden. Knoten, die weitere Unterpunkte enthalten sind dabei dunkler dargestellt, um den Nutzer graphisch darauf hinzuweisen, dass es dort weitere Informationen gibt.

Um eine Entität aus dem Register zu markieren und damit den Filter anzupassen, kann der Nutzer auf den angezeigten Namen der Entität klicken. Dadurch wird die Entität selektiert und zur einfacheren Übersicht in einer separaten Liste unter der Timeline aufgelistet. Möchte der Nutzer eine Entität wieder aus der Liste der Filterkriterien entfernen, kann er in dieser Liste einen Eintrag selektieren und diese Entität wird aus dem Filter entfernt.

Da die Namen der Registereinträge teilweise sehr lang sind (sie enthalten teilweise zusätzliche Beschreibungen), werden diese ab einer gewissen Länge (relativ zur Fenstergröße) gekürzt. Damit trotzdem der komplette Name des Eintrags dargestellt wird, existieren Tooltips, die beim Hovern über einen Registereintrag angezeigt werden.

Auch die Registereinträge werden innerhalb der Ansicht „verodert“. Das heißt, dass in der Ergebnismenge Regesten enthalten sind, in der mindestens eine der Entitäten erwähnt wird.

4.1.4 Übersicht über Filter

Mit **Ansicht (4)** erhält der Nutzer, Feedback über den derzeitigen Filter und kann diesen anpassen.

Places

Linz
Nürnberg
Waldshut

Entities

Aachen (Aache; Nordrhein-Westfalen), Stadt
Einwohner und Bürger
Fischmarkt (Parwisch)
Kopist s.K. F. Meyer

Abbildung 4.5: Übersicht über selektierte Ausstellungsorte und Entitäten

Wie in Abbildung 4.5 zu sehen, enthält der View eine Liste der selektieren Ausstellungsorte und Registereinträge. Diese werden hier zur Übersicht abgetragen, um auf einen Blick zu sehen welche Regesten in der Ergebnismenge enthalten sind. Der Nutzer kann durch Klicken auf Einträge der Liste die Selektion und damit die Einschränkung des Filters wieder entfernen.

4.1.5 Suchergebnisse

Nach dem der Nutzer die gewünschten Filtereinstellungen getätigt hat, werden die Ergebnisse in einer Tabelle, wie in **Ansicht (5)** dargestellt, abgetragen. Abbildung 4.6 zeigt einen Screenshot der Tabelle mit dem Ergebnis eine Anfrage. Von Nutzer müssen für eine Suchanfrage nicht alle Filter bestimmt werden. Es ist möglich nur nach dem Ausstellungsdatum, dem Ausstellungsort oder erwähnten Personen, Orten und Institutionen zu suchen. Einige beliebige Kombination dieser drei Kriterien ist jedoch jederzeit möglich.

title	issuer	place	date	entities	uri
Friedrich III. - Chmel n. 1758	Friedrich III.	Nürnberg	1444 September 25		
Friedrich III. - Chmel n. 1776	Friedrich III.	Nürnberg	1444 Oktober 4		
Friedrich III. - [RI XIII] H. 14 n. 281	Friedrich III.	Nürnberg	1444 Oktober 4		

Abbildung 4.6: Tabelle mit Ergebnissen einer Suchanfrage

Um das Ergebnis zu erhalten werden die Kriterien der anderen Ansichten (Timeline, Karte und Register) „verundet“. Das heißt, dass sobald ein Ausstellungszeitraum, Ausstellungsorte und Registereinträge ausgewählt wurden die Ergebnismenge aus Regesten besteht, die in diesem Zeitraum liegen, an einem der Orte ausgestellt wurden und mindestens eine der ausgewählten Entitäten im Regest erwähnt wird.

Die logischen Verknüpfungen innerhalb und zwischen den Ansichten, wurden dabei in Absprache mit dem Verantwortlichen der Regesta Imperii bestimmt, um möglichst gut auf die Bedürfnisse der Nutzer einzugehen.

Wie in Abbildung 4.6 auch zu erkennen ist, enthält die Tabelle die wichtigsten Meta-Daten eines Regests. Dazu zählen der Titel des Regests, der Author, der Ausstellungsort und das Ausstellungsdatum. Die letzten zwei Spalten enthalten zusätzlich interessante Informationen.

Durch die Selektion der Schaltfläche in der Spalte Entities, öffnet sich ein Menü, dass die Schnittmenge zwischen den ausgewählten Registereinträgen des Filters und den im Regest vorkommenden Entitäten enthält. Damit erhält der Nutzer Feedback darüber, welche der gesuchten Entitäten explizit in diesem Regest erwähnt werden.

Die letzte Spalte ist dabei für das weitere Arbeiten mit dem Suchergebnis relevant. Da die Visualisierung das existierende Online-Angebot der Regesta Imperii ergänzen soll, wechselt der Nutzer bei Klick auf die Schaltfläche in der letzten Spalten zum Eintrag des Regests auf der Webseite der Regesta Imperii. Dort kann der Nutzer ein Regest genauer untersuchen und seine Forschung weiterführen.

5 Implementierung

Im folgenden Kapitel werden wichtige Designentscheidungen und Herausforderungen bei der Umsetzung der Software vorgestellt und erläutert.

5.1 Datenaufbereitung

Wie bereits erwähnt wurden von den Regesta Imperii für diese Arbeit Regesten und das Register von Friedrich III. in Form von XML-Dateien für diese Arbeit zur Verfügung gestellt. Da XML-Dateien ungeeignet sind, um Queries auf ihnen auszuführen und wiederholte laden und parsen dieser Dateien nicht performant ist, wurden alle Regesten eingelesen, geparsed und in eine MongoDB gespeichert.

Umgesetzt wurde dies mit libxmljs ¹. Dieses NodeJS-Modul ermöglicht das Verwenden der libxml Bibliothek zum parsen von XML-Dateien in NodeJS. Dabei unterstützt libxmljs das parsen von XML mithilfe eines SAX-Parsers, der Knoten rekursiv einliest und immer nur einen Knoten des Baums im Speicher behält. Weiterhin steht ein DOM-Parsers zur Verfügung, der den kompletten Baum in den Speicher lädt und das verwenden der xpath-Syntax erlaubt, um direkt auf Knoten zuzugreifen. Dazu wird der Pfad von der Wurzel des XML-Baums bis zum gewünschten Knoten in xpath-Syntax angegeben.

Dabei war eine der größten Herausforderungen, dass die Daten uneinheitlich beschrieben werden und deswegen viele Sonderfälle abgefangen werden mussten. Wo möglich wurden falsche Daten verbessert (31. Februar) oder, wenn nötig, komplett ignoriert, da eine automatische Korrektur nicht möglich war und der Datensatz zu groß ist, um alle Einträge händisch zu korrigieren.

Nach dem Speichern von Regesten und Registereinträgen in der MongoDB, wurden diese Einträge miteinander verknüpft. Dafür wurden in den Regesten die IDs der erwähnten Registereinträge hinterlegt, um die Zeit die Suchanfragen an die Daten benötigen zu verkürzen.

Die drei Skripte die zur Datenaufbereitung verwendet wurden, können später auch für andere Autoren verwendet werden, da diese generisch gehalten wurden. Solange die Datensätze anderer Autoren das selbe Format haben, können die existierenden Skripte verwendet werden, um auch die Datensätze anderer Autoren für diese Visualisierung vorzubereiten.

¹<https://github.com/polotek/libxmljs>

5.2 Visualisierung

Im folgenden Abschnitt werden die wichtigsten Designentscheidungen für die Entwicklung der Visualisierung vorgestellt. Dabei wird zwischen den Entscheidungen für das Back End (Server) und denen für das Front End (Client) unterschieden. Anschließend wird die Umsetzung der Filterfunktionalität im Detail erläutert.

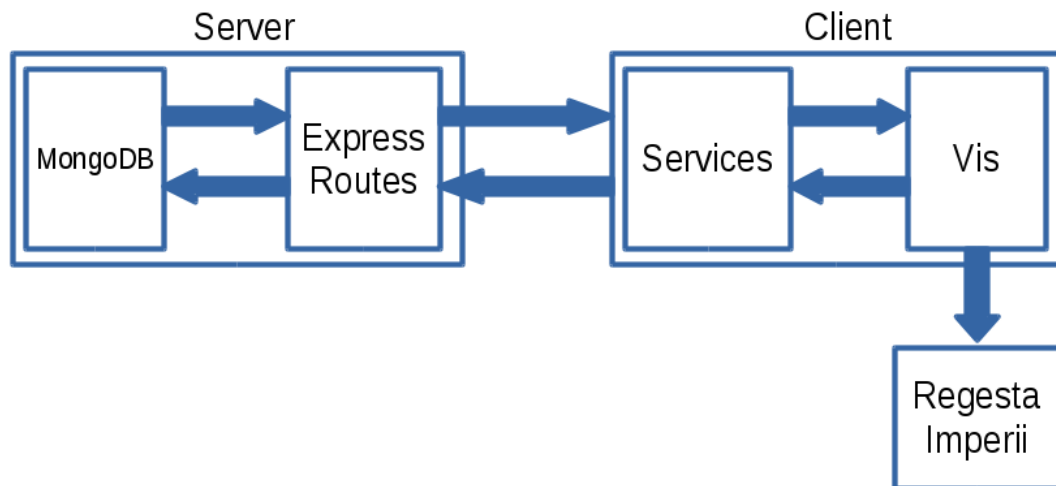


Abbildung 5.1: Schematischer Aufbau der entwickelten Software

5.2.1 Back End

Das Back End der Visualisierung besteht aus einem NodeJS Server, der mit Hilfe von ExpressJS einen Webserver implementiert. In Kombination mit einer MongoDB ergibt das beinahe den MEAN-Stack ², der häufig für die Entwicklung von Webanwendungen verwendet wird. Das Verwenden von JavaScript auf Server- und Client Seite bietet dabei den Vorteil, dass die selben Build-Tools verwendet werden können und es nicht zu Inkompatibilitätsproblemen kommen kann. Der Server besteht wie in Kapitel 2 vorgestellt aus einem Webserver, der mittels Routen Daten an den Client ausliefert. Die Daten erhält der Server aus einer MongoDB-Instanz. Der schematische Aufbau des Servers wird in Abbildung 5.1 noch einmal dargestellt.

Das Speichern beziehungsweise das Laden von Einträgen aus der MongoDB wird dabei durch mongoose ³ ergänzt. Mongoose wird dabei verwendet, um Regesten und Register in einem einheitlichen Format zu speichern und eine Validierung der Daten vorzunehmen. Die MongoDB bietet von Haus aus keine Möglichkeit die Daten auf Korrektheit zu überprüfen, beziehungsweise Modelle für diese Daten zu definieren. Listing 5.1 zeigt das Modell zur Speicherung von Regesten in der MongoDB.

²<http://mean.io/>

³<http://mongoosejs.com/>

Listing 5.1 Modell zur Speicherung von Regesten

```
var Regesta = mongoose.model('Regesta', {
  title: String,
  issuer: String,

  issueInformation: {
    number: String,
    department: String,
    volume: String,
    issue: String
  },
  issueDate: {
    text: String,
    date: Date
  },
  issuePlace: {
    name: String,
    coordinates: {
      lat: Number,
      lng: Number
    }
  },
  abstract: String,
  entities: [String]
});
```

Das Modell enthält dabei auch den Abstract eines Regests, der derzeit nicht verwendet wird, jedoch für eine möglicherweise zukünftige Nutzung bereits gespeichert wird.

5.2.2 Front End

Das Front End wurde wie für Webapplikationen typisch mit HTML, CSS und JavaScript entwickelt. Dabei wurden alle Visualisierungen mit D3 [BOH11] erstellt. Insbesondere wurde zum einfachen Erstellen der Filterkriterien Underscore ⁴ verwendet, da mit diesem Framework Mengenoperationen einfach zu implementieren sind. Bei Underscore handelt es sich um eine JavaScript Bibliothek, die über 100 Hilfsfunktionen zur funktionalen Programmierung in JavaScript zur Verfügung stellt. Zu diesen Hilfsfunktionen gehören auch solche, die Mengenoperationen auf JavaScript Arrays ermöglichen.

Die einzelnen Ansichten der Visualisierung sind jeweils in eigene Module gekapselt, die auch unabhängig von den anderen existieren und funktionieren können. Dadurch ist es möglich weitere Ansichten hinzuzufügen oder existierende Ansichten durch neue Ansichten zu ersetzen. Derzeit ist dies mit Hilfe von separaten Dateien und Closures gelöst. In Zukunft soll dafür Browserify ⁵ verwendet werden, um den Code von Server und Client noch einheitlicher zu gestalten.

⁴<http://underscorejs.org/>

⁵<http://browserify.org/>

Abbildung 5.1 zeigt den schematischen Aufbau des Clients. Dieser enthält Services, welche die Daten von Server abrufen und aufbereiten. Die Views erhalten dann ihre Daten von den Services und zeigen die verschiedenen Visualisierungen an. Die Tabelle mit der Ergebnismenge verlinkt weiter auf das Online-Angebot der Regesta Imperii. Auch diese Verlinkung ist in Abbildung 5.1 schematisch dargestellt.

5.2.3 Filter

Der Filter ist in zwei Teile aufgeteilt. Einerseits werden im Client alle Teile der Filter-Query zusammengesetzt und an den Server übertragen, andererseits nutzt der Server diese Informationen um die Query für Mongoose aufzubereiten.

Listing 5.2 Client-Format der ausgewählten Einträge

```
var selected = {  
  coordinates: places,  
  dates: selectedDates,  
  entities: selectedEntities  
};
```

Listing 5.2 zeigt wie der Client die für den Filter benötigten Informationen in einem einfachen JavaScript-Objekt speichert. Dieses enthält eine Liste von IDs von Orten, eine Listen von JavaScript Date-Objekten und eine Liste von IDs von Registereinträgen.

Immer dann wenn die Auswahl in einer der Ansichten aktualisiert wird, wird ein Objekt dieser Art an den Server übertragen, um dort eine Query für Mongoose zu generieren. Dieses Objekt wird auf Serverseite validiert. Wenn es alle benötigten Informationen enthält, wird eine Query für Mongoose erstellt. Dabei können auch nur Teile der Informationen gegeben sein. Es müssen also nur der Veröffentlichungszeitraum oder die Ausstellungsorte oder die Registereinträge definiert sein. Beliebige Kombination sind dabei jedoch möglich.

6 Ergebnisse

In Kapitel 4 und 5 wurde die entwickelte Visualisierung und wichtige Designentscheidungen vorgestellt. In diesem Kapitel sollen die Ergebnisse und Verbesserungen gegenüber der existierenden Suche der Regesta Imperii vorgestellt werden.

6.1 Verbesserungen

Die Expertensuche im Online-Angebot der Regesta Imperii ist ein mächtiges Werkzeug, um Regesten aus einem großen Satz an Daten zu extrahieren. Ein großer Nachteil der Expertensuche ist jedoch, dass wenn eine Anfrage einmal gestellt wurde, diese nicht mehr angepasst werden kann und deswegen eine Suchanfrage gestellt wird. Dieses neue Erstellen eine Anfrage ist nicht nur Aufwendig sondern auch demotivierend, wenn jeweils nur kleine Änderungen durchgeführt werden müssen, um zum gewünschten Ergebnis zu kommen. Außerdem ist es auf diese Weise kaum möglich, den existierenden Datensatz zu explorieren.

Die in dieser Arbeit vorgestellte Visualisierung bietet aus diesem Grund die Möglichkeit, die Filterparameter anzupassen. Durch ein schnelleres Feedback kann der Nutzer die komplette Menge an Regesten explorieren und ist in der Lage genauere Suchanfragen aus existierenden Fragestellungen zu erstellen.

Eine der größten Herausforderungen bei der Nutzung der Expertensuche ist die Größe der Ergebnismenge. Je nach Suchanfrage fällt das Ergebnis entweder sehr klein oder unüberschaubar groß aus. Für diese Arbeit wurde nur der Datensatz von Friedrich III. verwendet, was bereits ein gewaltige Vorteil ist, da jetzt nur circa 22.000 Regesten verarbeitet werden. Zusätzlich können die Filterparameter iterativ angepasst werden. Dies gibt dem Nutzer die Möglichkeit bessere Ergebnisse zu erzielen, da die Kriterien nach belieben hinzugefügt und entfernt werden können, bis eine kontrollierbare Ergebnismenge erreicht ist.

6.2 Use Case

Ziel dieser Arbeit ist es die Expertensuche der Regesta Imperii zu ergänzen. Die Expertensuche eignet sich für die Suche nach Regesten, für die die Regestennummer bekannt ist, beziehungsweise genug Informationen über das Regest bekannt sind, um die Größe der Ergebnismenge gering zu halten. Dann ist es möglich die verbleibenden Regesten händisch nach dem gewünschten Regest zu durchsuchen.

Wenn jedoch nicht genug über das gesuchte Regest oder die gesuchten Regesten bekannt ist muss die Expertensuche mehrmals hintereinander durchgeführt werden. Dabei gehen jedoch die Ergebnisse

der letzten Suche verloren oder es muss ein neuer Tab im Browser geöffnet und dort die neue Suche durchgeführt werden.

Die entworfene Visualisierung soll diese neue Form der Exploration ermöglichen. Dafür werden die Filterkriterien graphisch aufbereitet und der Nutzer kann diese jederzeit anpassen. Das iterative Anpassen der Filterkriterien führt dabei zur gewünschten Ergebnismenge, bestehend aus genau den Regesten, die alle Kriterien erfüllen.

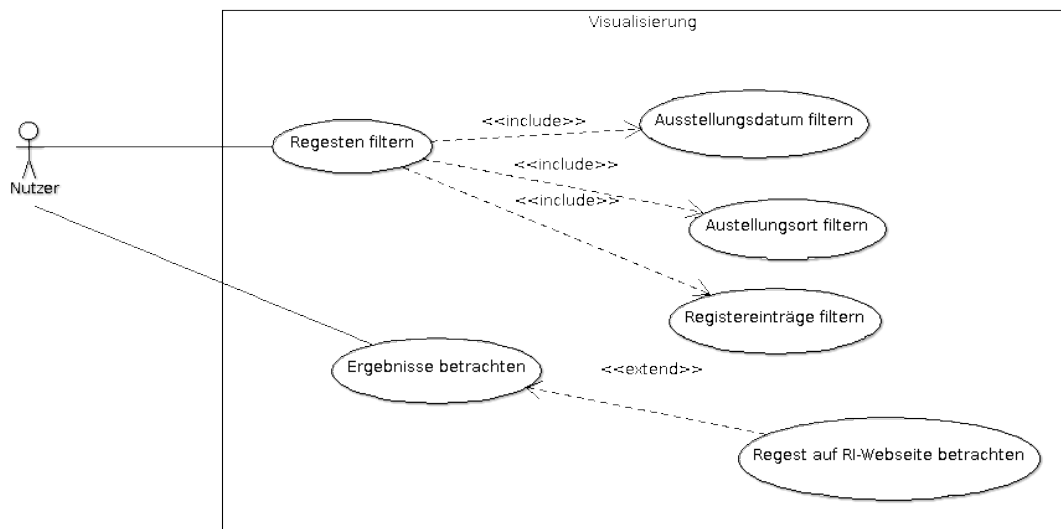


Abbildung 6.1: Use Case Diagram

Abbildung 6.1 zeigt das während der Anforderungsanalyse erstellte Use-Case-Diagramm. Die Funktionalität der Software wird dabei auf die zwei Haupteigenschaften der Suche nach Regesten beschränkt. Erstens kann die Menge der Regesten nach drei Kriterien gefiltert werden. Zweitens kann das Ergebnis genauer untersucht werden. Dafür kann der Nutzer das existierende Online-Angebot der Regesta Imperii nutzen, auf die von der Software gelinkt wird.

6.3 Expertenfeedback

Vor der Entwicklung der Software wurde bereits eine Anforderungsanalyse mit Vertretern der Regesta Imperii durchgeführt. Auf diese Weise soll sichergestellt werden, dass die entwickelte Visualisierung einen Nutzen für die Forschung in den Geisteswissenschaften hat.

Nach Absprache wurde anschließend die Software entwickelt und nach Fertigstellung in einem Workshop an der Akademie der Wissenschaften und der Literatur in Mainz präsentiert. Dabei wurden bereits Fehler in den zur Verfügung gestellten Daten festgestellt. Die entwickelte Visualisierung

stellt die Veröffentlichungsorte der Regesten als Kreise über eine Karte dar. In Bremen werden dabei besonders viele Urkunden markiert. Wie sich herausstellte, wurden die Namen „Bremen Neustadt“ und „Wiener Neustadt“ verwechselt und aus diesem Grund die falschen Koordinaten in den XML-Dateien hinterlegt.

Von den Vertretern der Regesta Imperii wurde weiterhin angemerkt, dass sich aktuelle Forschungsfragen durch den iterativen Ansatz der Visualisierung schneller und leichter beantworten lassen würden. Die Regesta Imperii planen außerdem weiter fehlerhafte Daten zu korrigieren mit weiteren Informationen zu annotieren. Dazu zählen insbesondere der Typ der Entitäten in den Registern. Auf diese Weise ist es möglich die Registeransicht der Visualisierung mit zusätzlichen Informationen zu versehen und dem Nutzer einen schnelleren Überblick zu ermöglichen.

7 Zusammenfassung und Ausblick

Abschließend werden in diesem Kapitel die Ergebnisse der Arbeit zusammengefasst und ein Fazit gezogen. Dafür werden die größten Schwierigkeiten beim Entwurf und der Entwicklung der Visualisierung aufgeführt und die vorliegende Arbeit in ihrer Vollständigkeit bewertet. Den Abschluss bildet der Ausblick, in dem auf die Möglichkeiten eingegangen wird, mit denen diese Arbeit erweitert werden kann.

7.1 Herausforderungen

In diesem Abschnitt werden noch einmal die größten Herausforderungen von der Analyse bis zur Implementierung aufgeführt und untersucht.

7.1.1 Fehlerhafte Daten

Die von den Regesta Imperii zur Verfügung gestellten Daten enthielten teilweise Syntaxfehler oder waren nicht einheitlich aufgebaut. Beispielhaft wurde dies in Kapitel 2 dargestellt. Teilweise konnten Fehler in den Daten erkannt und automatisch korrigiert werden. Dies sind beispielsweise fehlerhafte Datumsangaben. Wenn nicht anders möglich, mussten fehlerhafte Daten jedoch ignoriert werden. Da meist nicht klar war, welche Daten korrekt und welche fehlerbehaftet sind, konnten diese auch nicht markiert werden, um den Nutzer der Visualisierung darauf hinzuweisen, dass Daten möglicherweise Fehlerhaft sind. Gerade bei der Arbeit mit historischen Quellen, die in der Wissenschaft verwendet werden, wären fehlerbehaftete Daten fatal.

7.1.2 Fehlende Dokumentation

Die Regesta Imperii haben einen großen Datensatz zur Verfügung gestellt und damit diese Arbeit ermöglicht. Die histographischen Daten sind jedoch für den Laien auf diesem Gebiet schwer zugänglich. Weiterhin ist, wie einem Gespräch mit Vertretern der Regesta Imperii zu entnehmen, die Struktur der XML-Dateien historisch gewachsen und deshalb teilweise nicht auf die neuen Anforderungen ausgelegt. Fehlende Dokumentation und ein umständlicher Aufbau der XML-Dateien führte deshalb dazu, dass die Aufbereitung der Daten einen großen Anteil an der Erstellung dieser Arbeit einnahm.

7.1.3 Performance

Das letzte große, wenn auch erwartete Problem war und ist die Performance. Das Durchsuchen der circa 22 tausend Regesten stellt dabei ein kleines Problem dar. Die Datenbank und der Server stellen kein Problem dar. Auf Client-Seite kommt es jedoch teilweise zu Performanceproblemen, wenn das Ergebnis der Suche sehr groß ausfällt. Das Erstellen der Tabelle mit D3 nimmt dabei einige Zeit in Anspruch. Der Grund dafür ist, dass in D3 umständlich über Zeilen und Spalten iteriert werden muss und viele unnötige Schritte den Schleifenrumpf unnötig aufblähen. Auch die indirekte Interaktion mit der DOM API führen beim Hinzufügen vieler Knoten in das Dokument zu langsamen Ergebnissen.

7.2 Zusammenfassung

Mit den Daten der Regesta Imperii lassen sich viele Forschungsfragen der Geschichtswissenschaften beantworten. Diese Daten mussten bisher jedoch entweder manuell oder über die Online-Suchfunktion der Regesta Imperii durchsucht werden. In dieser Arbeit wurde ein neuer Ansatz vorgestellt, der die Daten der Regesta Imperii graphisch darstellt. Dadurch ist es dem Nutzer möglich die Regesten zu explorieren und neue Aspekte der Daten zu erkennen. Im Rahmen der Vorstellung der Visualisierung an der Akademie der Wissenschaften und der Literatur in Mainz wurde bereits ein Fehler in den Daten zu Friedrich III. gefunden. Eine Verwechslung von Stadtnamen führte dabei dazu, dass zu einem Stadtnamen die falschen Koordinaten gespeichert wurden und deshalb viele Urkunden an einem falschen Ort in der Visualisierung dargestellt wurden.

7.3 Ausblick

Derzeit existiert eine lauffähige Basisversion eines MCV-Systems für die Visualisierung von Regesten zu Friedrich III. Dieses System kann jedoch um einige Punkte erweitert werden, um dem Nutzer noch mehr Möglichkeiten zu geben, Informationen zu extrahieren und neue Aspekte der Daten zu erfassen.

7.3.1 Weitere Informationen und Verknüpfungen

Der derzeit existierenden Ansichten visualisieren unterschiedliche Eigenschaften eines Regests und bilden zusammen die Grundlage für die Filter-Query. Um dem Nutzer noch mehr Feedback über seine Auswahl zu geben, ist es vorteilhaft, noch mehr Brushing und Linking Funktionen zu implementieren. Ein Vorbild dafür ist das in Kapitel 2 vorgestellte Tool PatViz. Das Highlighten und Anpassen der Visualisierung bei der Interaktion mit den Daten, gibt dem Nutzer die Möglichkeit noch mehr Informationen aus den Daten zu extrahieren.

Geplant ist deshalb das Abtragen der relativen Verteilung von Regesten in der Timeline mit Hilfe von Barcharts oder Ähnlichem. Abbildung 7.1 zeigt, wie diese relative Verteilung dargestellt werden kann.

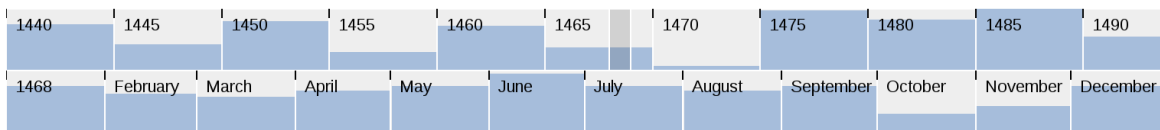


Abbildung 7.1: Relative Verteilung von Regesten über die Zeit

Die Timeline wird es in Bins eingeteilt und in diesen dann Balken dargestellt, die die relative Verteilung der Regesten repräsentieren. Im unteren Teil der Timeline wird dann die relative Verteilung der Regesten angezeigt, die im oberen Teil der Timeline selektiert wurden.

Derzeit ist es im Register-View nicht möglich zu erkennen, welche Einträge bisher selektiert wurden. Aus diesem Grund soll auch hierfür ein visuelles Feedback, beispielsweise durch Icons ergänzt werden. Auf diese Weise erhält der Nutzer auf den ersten Blick eine Übersicht über die bereits selektierten Registereinträge.

Zusätzlich soll im Register-View dargestellt werden, um welchen Typ von Registereintrag es sich handelt. Dabei soll zwischen Personen und Orten unterschieden werden. Sobald diese in den XML-Dateien entsprechend annotiert wurden, können diese wie in Abbildung 7.2 dargestellt farbkodiert werden. Beispielsweise können wie in Abbildung 7.2 Personen blau und Ort rot hinterlegt werden.

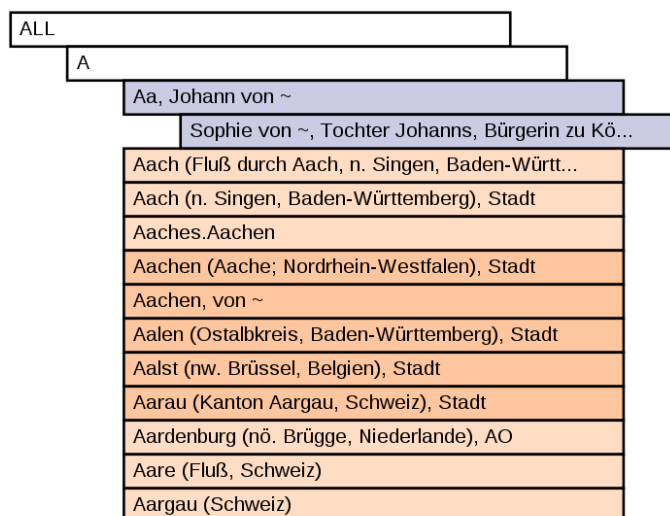


Abbildung 7.2: Colorcoding für Personen und Orte im Register

Um die einzelnen Views weiter miteinander zu verknüpfen, können die Vorkommen von Register-einträgen und Ausstellungsorten in der Timeline markiert werden. Dadurch erhält der Nutzer eine schnelle Übersicht über interessante Zeiträume und muss nicht umständlich, ohne weitere Hinweise nach diesen suchen.

7.3.2 Weitere Einstellungsmöglichkeiten

Bei der Timeline gibt es derzeit die Einschränkung, dass nur ein Start- und Enddatum ausgewählt werden kann. Dies könnte durch beliebig viele Zeiträume ergänzt werden und auch die Granularität der Timeline weiter verbessert werden, indem weitere Ebenen dynamisch zur Timeline hinzugefügt werden, um feinere Datumsangaben durchzuführen.

7.3.3 Bessere Usability

Bei der Auswahl von Registereinträgen muss teilweise noch ein großer Teil der Einträge durchsucht werden, um den gewünschten Eintrag zu finden und auszuwählen. Dies könnte durch eine Suchfunktion verhindert werden, wie sie etwa in der LensBar implementiert ist. Derzeit ist dies jedoch nicht sinnvoll möglich, da die Namen der Registereinträge noch nicht ausreichend eindeutig sind. Die Namen der Registereinträge ergeben sich teilweise aus der Hierarchie und sind nicht explizit gespeichert. Im Rahmen weiterer Projekte möchten die Regesta Imperii jedoch die Daten aufbereiten und zusätzlich mit Annotationen versehen, ob es sich bei einem Registereintrag um eine Person, einen Ort oder eine Institution handelt. Mit dieser Information kann eine weitere visuelle Hilfestellung in die Visualisierung des Registers implementiert werden.

7.4 Fazit

Die Ergänzung des Angebots der Regesta Imperii durch eine Visualisierung ist eine geeignete Möglichkeit, weitere Informationen aus den gegebenen Daten zu extrahieren. Wie sich bereits in Gesprächen mit Vertretern der Regesta Imperii herausstellte, können damit Forschungsfragen einfacher beantwortet werden und es ergeben sich neue Fragen, die vorher noch nicht in Betracht gezogen wurden. Aus diesem Grund ist es wichtig im Kontakt mit Historikern zu bleiben und ihre Anregungen in Erweiterungen der vorgestellten Visualisierung umzusetzen. Weiterhin können andere Ansätze diese Arbeit bereichern und die Arbeit im Rahmen der Digital Humanities weiter vorantreiben.

Die Regesta Imperii planen ihre Daten weiter zu digitalisieren, auszubauen und zu bereinigen. Mit diesen neuen Daten wird es möglich sein, Visualisierungen zu entwerfen, die stärker Bezug auf die Einzigartigkeit des Datensatzes nehmen.

In einem Workshop der Regesta Imperii an der Akademie der Wissenschaften und der Literatur in Mainz wurden bereits weitere Arbeiten, unter anderem auch ein weiterer Visualisierungsansatz vorgestellt, der im Rahmen eines von der Regesta Imperii durchgeführten Hackathons entwickelt wurde. Diese Verbindung von interessanten und hochwertigen Daten mit der Open Source Community und dem Input aus dem Bereich der Visualisierung können dann in mächtigeren Tools resultieren.

Literaturverzeichnis

- [AS94] C. Ahlberg, B. Shneiderman. The alphalider: a compact and rapid selector. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, S. 365–371. ACM, 1994. (Zitiert auf Seite 26)
- [AT95] C. Ahlberg, S. Truvé. Tight coupling: guiding user actions in a direct manipulation retrieval system. In *BCS HCI*, S. 305–321. 1995. (Zitiert auf Seite 26)
- [Ben83] H. D. Benington. Production of large computer programs. *IEEE Annals of the History of Computing*, (4):350–361, 1983. (Zitiert auf Seite 11)
- [BMMS91] A. Buja, J. A. McDonald, J. Michalak, W. Stuetzle. Interactive data visualization using focusing and linking. In *Visualization, 1991. Visualization '91, Proceedings., IEEE Conference on*, S. 156–163. IEEE, 1991. (Zitiert auf Seite 30)
- [BOH11] M. Bostock, V. Ogievetsky, J. Heer. D³ data-driven documents. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2301–2309, 2011. (Zitiert auf den Seiten 36 und 45)
- [Fie00] R. T. Fielding. *Architectural styles and the design of network-based software architectures*. Dissertation, University of California, Irvine, 2000. (Zitiert auf Seite 35)
- [fis] Innovative user interface design. URL <http://www.webcredible.com/i/fisheye.gif>. Abgerufen am 18.10.2015. (Zitiert auf Seite 28)
- [GLP⁺07] C. Gorg, Z. Liu, N. Parekh, K. Singhal, J. Stasko. Visual analytics with Jigsaw. 2007. (Zitiert auf Seite 31)
- [ISO04] ISO. Data elements and interchange formats – Information interchange – Representation of dates and times. ISO 8601:2004, International Organization for Standardization, Geneva, Switzerland, 2004. (Zitiert auf Seite 16)
- [JHSS12] S. Jänicke, C. Heine, R. Stockmann, G. Scheuermann. Comparative Visualization of Geospatial-temporal Data. In *GRAPP/IVAPP*, S. 613–625. 2012. (Zitiert auf den Seiten 25 und 31)
- [KBGE09] S. Koch, H. Bosch, M. Giereth, T. Ertl. Iterative integration of visual insights during patent search and analysis. In *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on*, S. 203–210. IEEE, 2009. (Zitiert auf Seite 33)
- [len] LensBar. URL <http://i.gyazo.com/249537c2115edcb821df7fe689ae3e79.png>. Abgerufen am 18.10.2015. (Zitiert auf Seite 27)

- [Mas98] T. Masui. LensBar-visualization for browsing and filtering large lists of data. In *Information Visualization, 1998. Proceedings. IEEE Symposium on*, S. 113–120. IEEE, 1998. (Zitiert auf Seite 27)
- [Nat00] National Imagery and Mapping Agency. Department of Defense World Geodetic System 1984. Technischer Bericht, 2000. (Zitiert auf Seite 16)
- [OLS93] M. Osada, H. S. Liao, B. Shneiderman. *AlphaSlider: Searching Textual Links with Sliders*. University of Maryland, Center for Automation Research, Human-Computer Interaction Laboratory, 1993. (Zitiert auf Seite 26)
- [RIh] Hilfe: Regesta Imperii. URL <http://www.regesta-imperii.de/regesten/hilfe.html>. Abgerufen am 18.10.2015. (Zitiert auf Seite 8)
- [RIo] Regesta Imperii RI Opac. URL http://regesta-imperii.digitale-sammlungen.de/seite/ri13_wil1999_0055. Abgerufen am 18.10.2015. (Zitiert auf Seite 14)
- [Rob07] J. C. Roberts. State of the art: Coordinated & multiple views in exploratory visualization. In *Coordinated and Multiple Views in Exploratory Visualization, 2007. CMV'07. Fifth International Conference on*, S. 61–71. IEEE, 2007. (Zitiert auf Seite 33)
- [RR12] S. Robertson, J. Robertson. *Mastering the requirements process: Getting requirements right*. Addison-wesley, 2012. (Zitiert auf Seite 10)
- [Sch08] M. Scherr. Multiple and coordinated views in information visualization. *Trends in Information Visualization*, 38, 2008. (Zitiert auf Seite 33)
- [Shn96] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, S. 336–343. IEEE, 1996. (Zitiert auf Seite 21)
- [Sim] Simile Timeline. URL <http://www.simile-widgets.org/timeline/>. Abgerufen am 18.10.2015. (Zitiert auf Seite 22)
- [SQT07] H. Song, Y. Qi, X. Tian, D. Xu. Navigating and visualizing long lists with fisheye view and graphical representation. In *Digital Media and its Application in Museum & Heritages, Second Workshop on*, S. 123–128. IEEE, 2007. (Zitiert auf Seite 28)
- [WBWK00] M. Q. Wang Baldonado, A. Woodruff, A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the working conference on Advanced visual interfaces*, S. 110–119. ACM, 2000. (Zitiert auf Seite 33)

Alle URLs wurden zuletzt am 19. Oktober 2015 geprüft.

Erklärung

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

Ort, Datum, Unterschrift