

1 Ansatz

Die inhaltliche Aufgabe der *Textextraktion-18* war es alle Reden der 1. - 18. Wahlperiode zu extrahieren, hierbei sollten die vom *Crawler* übergebenen XML-Dokumente extrahiert und in JSON Dateien ausgegeben werden. Der *Crawler* realisierte dies, durch die temporäre Persistierung der Plenarprotokolle als XML Dateien[1] und Übergabe als Parameter an die *Textextraktion-18*.

Gegensatz zu den Plenarprotokollen der 19. Legislaturperiode liegen die Protokolle der 18. Legislaturperiode nicht in reinem XML Format vor (siehe Abbildung 1.1), sondern es gibt am Anfang der XML Dateien die nötigsten XML Tags und das Protokoll selbst befindet sich in einem einzigen großen Textblock.

```
<?xml version="1.0" encoding="UTF-8"?>
<DOKUMENT>
  <WAHLPERIODE>18</WAHLPERIODE>
  <DOKUMENTART>PLENARPROTOKOLL</DOKUMENTART>
  <NR>18/3</NR>
  <DATUM>28.11.2013</DATUM>
  <TITEL>Plenarprotokoll vom 28.11.2013</TITEL>
  <TEXT>Plenarprotokoll 18/3
Deutscher Bundestag
Stenografischer Bericht

3. Sitzung

Berlin, Donnerstag, den 28. November 2013

I n h a l t :
Nachruf auf den ehemaligen Bundestagsvize-
präsidenten Dieter-Julius Cronenberg . . . . . 75 A
Erweiterung und Abwicklung der Tagesord-
nung . . . . . 75 C
Absetzung des Tagesordnungspunktes 13 . . . . 75 D
```

Abbildung 1.1: 18003.xml

Die Grundidee war nun also den Textblock semantisch und syntaktisch zu Untersuchen um einen Suchalgorithmus zu finden, welcher innerhalb des Textblockes:

- Titel der Rede
- Zugehörigkeit
- Rede
- Name des Sprechenden
- Datum

zu einer Person findet und extrahiert.

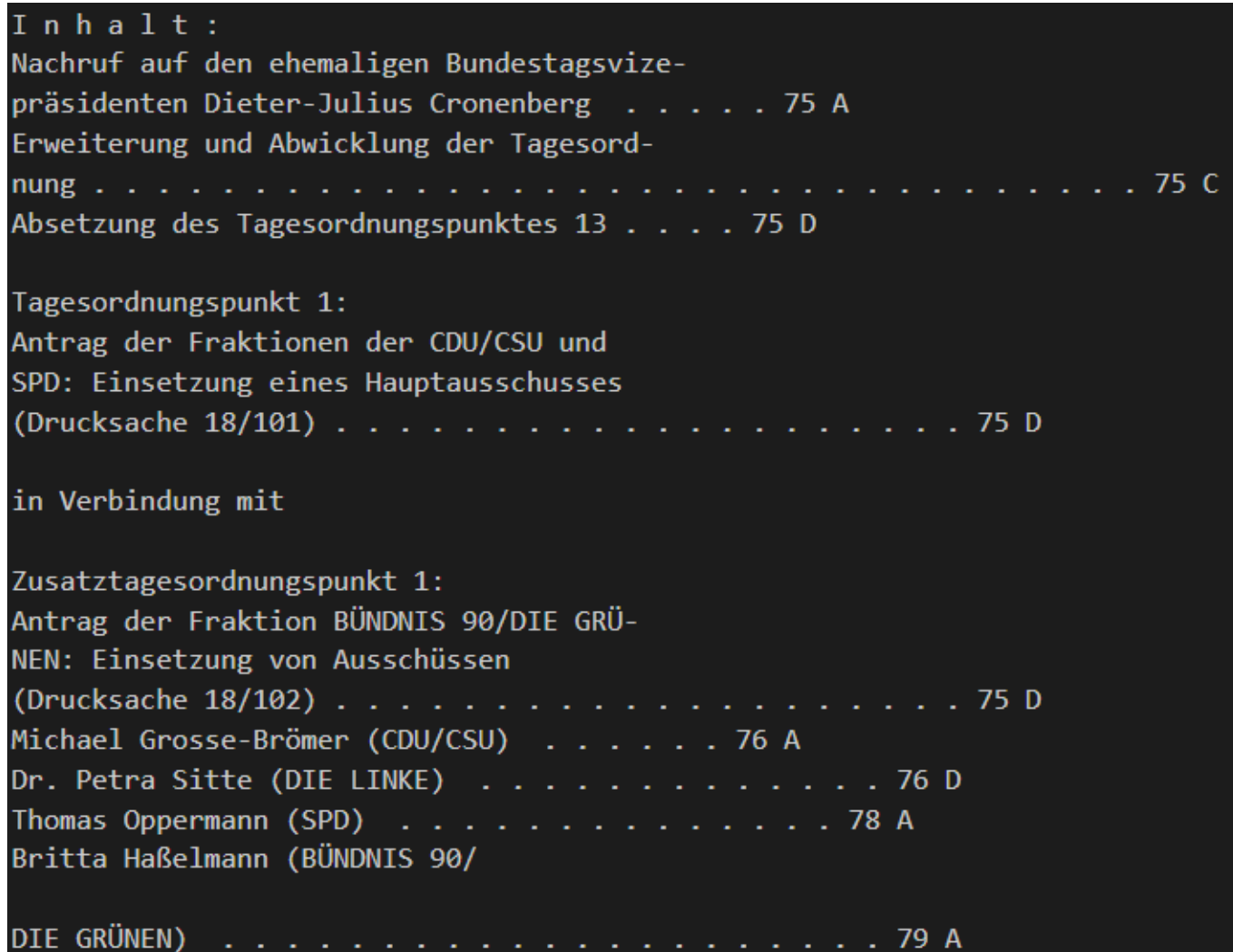
2 Lösungsansatz

Bei den Dokumenten handelt es sich um stenografische Berichte des deutschen Bundestages, weswegen angenommen wurde, dass alle Dokumente eine Konsistenz in der Formatierung des Textblockes besitzen. Somit befasste sich das Teilprojekt *Textextraktion-18* vorerst mit der Extraktion eines Dokumentes, beispielhaft sollte dies das XML-Dokument 18003.xml sein.

2.1 Analyse der XML-Dokumente

2.1.1 Inhaltsverzeichnis

Bereits nach kurzer Analyse des Dokuments stellte sich eine Auffälligkeit innerhalb des Inhaltsverzeichnisses heraus. Alle Kapitel wurden mit ihrem Titel, dem dazugehörigem Abschnitt im Protokoll und den beteiligten Sprechern gelistet (siehe Abbildung 2.1).

The image shows a screenshot of a document's table of contents, rendered in a monospaced font on a dark background. The text is yellow and green. It lists various sections of a stenographic report, including a prelude, a main agenda item, and additional agenda items, each followed by a page number and a speaker's initials. The structure is as follows:
I n h a l t :
Nachruf auf den ehemaligen Bundestagsvizepräsidenten Dieter-Julius Cronenberg 75 A
Erweiterung und Abwicklung der Tagesordnung 75 C
Absetzung des Tagesordnungspunktes 13 75 D

Tagesordnungspunkt 1:
Antrag der Fraktionen der CDU/CSU und SPD: Einsetzung eines Hauptausschusses (Drucksache 18/101) 75 D

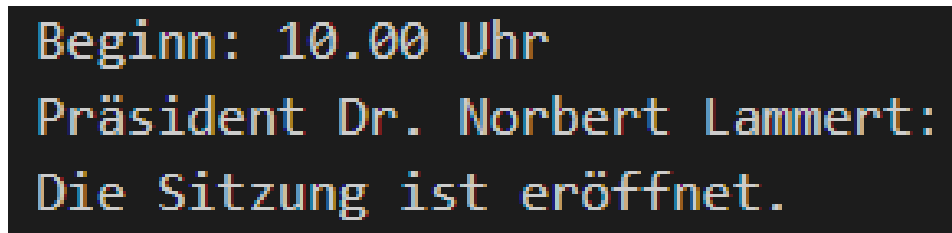
in Verbindung mit

Zusatztagesordnungspunkt 1:
Antrag der Fraktion BÜNDNIS 90/DIE GRÜNEN: Einsetzung von Ausschüssen (Drucksache 18/102) 75 D
Michael Grosse-Brömer (CDU/CSU) 76 A
Dr. Petra Sitte (DIE LINKE) 76 D
Thomas Oppermann (SPD) 78 A
Britta Haßelmann (BÜNDNIS 90/DIE GRÜNEN) 79 A

Abbildung 2.1: Ansicht des Inhaltsverzeichnisses

2.1.2 Eröffnungsrede

Zusätzlich wurde bei kurzem Vergleich mit wenigen anderen Dokumenten festgestellt, dass die Eröffnungsrede des Alterspräsident(siehe Abbildung 2.2) scheinbar immer gleich eingeleitet wurde: „Beginn: 10.00 Uhr“ (natürlich mit abweichender Uhrzeit)

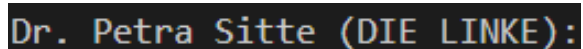


Beginn: 10.00 Uhr
Präsident Dr. Norbert Lammert:
Die Sitzung ist eröffnet.

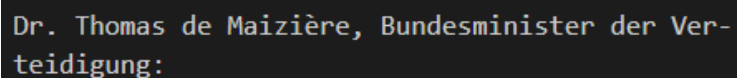
Abbildung 2.2: Einleitung der Eröffnungsrede

2.1.3 Redner

Darüber hinaus beginnt jede Rede mit dem Namen des Sprechenden und dessen Zugehörigkeit



Dr. Petra Sitte (DIE LINKE):



Dr. Thomas de Maizière, Bundesminister der Verteidigung:

2.2 Extraktion mit Regulären Ausdrücken

Es entwickelte sich die Idee anhand der Analyse reguläre Ausdrücke zu entwickeln die innerhalb des Textblockes passen (matchen). Für die einzelnen Teile der Analyse wurde dies wie folgt realisiert:

Analyseteil	regulärer Ausdruck in der Aufzählungsklasse
Inhaltsverzeichnis	TOC_NAMES(<code>Pattern.compile("\\d\\s[A-Z]\\n+")</code>)
Eröffnungsrede	OPENING(<code>Pattern.compile("(Uhr(\\r \\n (\\n)*)?(?=(Vize Alters)?(P p)räsident(en in innen)?))")</code>)
Redner	BREAKINGPOINT(<code>Pattern.compile("(\\s+minister.+\\r*\\n*+:\\n) ((\\s+p P)räsident.+:\\n) (\\s+(\\s+):\\n) (\\s+(\\s+/\\s+\\r*\\n*+):\\n) (\\s+(\\s+\\r*\\n*+):\\n) (\\s+\\.+kanzler.+:\\n)")</code>)

2.3 Theoretischer Ablauf

Das zu erstellende Programm sollte nun grob wie folgt ablaufen:

1. Einlesen des Dokuments, einzig relevante XML Tags sind `<DATUM>` und `<TEXT>`
2. Alle Suchalgorithmen laufen innerhalb des Textblockes ab, das heißt im `<TEXT>` Tag
3. Durchsuche jeden Eintrag im Inhaltsverzeichnis nach Titeln und den dazugehörigen Rednern.
4. Das Ende des Inhaltsverzeichnisses ist der Beginn der Eröffnungsrede des Alterspräsidenten.
5. Alle nachträglichen Reden sollen dem Titel aus dem Inhaltsverzeichnis zugeordnet werden.
6. Das Ende jeder Rede ist der Beginn einer neuen und kann mit dem `BREAKINGPOINT` gefunden werden.
7. Wurden alle Reden des Dokumentes gefunden sollen sie als JSON Datei gespeichert werden.

im GitHub erreichbar unter:

<https://github.com/htw-projekt-p2p-volltextsuche/textextraktion-18>

3 Probleme

Die meisten Probleme entstanden bei der Annahme es gäbe Konsistenz in der Formatierung der Dokumente (siehe Kapitel 2, Seite 2).

3.1 Dokumenttyp

Erste Probleme tauchten noch vor der eigentlichen Extraktion auf, nämlich beim Einlesen der Plenarprotokolle in XML Form auf.

Mit Angabe des Dokumententyps

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DOKUMENT SYSTEM "BUNDESTAGSDOKUMENTE.dtd">
<DOKUMENT>
  <WAHLPERIODE>1</WAHLPERIODE>
```

Ohne Angabe des Dokumententyps

```
<?xml version="1.0" encoding="UTF-8"?>
<DOKUMENT>
  <WAHLPERIODE>17</WAHLPERIODE>
```

Abbildung 3.1: Unterschiede in der Angabe des Dokumententyps

In der 1. - 14. Wahllegislaturperiode wurden den XML-Dateien ein Dokumententyp angegeben, ab der 15. Wahllegislaturperiode war die nicht mehr der Fall. Beipielhaft wird dies in Abbildung 3.1 durch ein Ausschnitt eines Protokolls der 1. und eines der 17. Wahllegislaturperiode verdeutlicht.

3.2 Formatierung

```
Befragung der Bundesregierung: Entwurf ei-
ner Verordnung über die Bezugsfrist für
das Kurzarbeitergeld; weitere Fragen zur
Kabinettsitzung . . . . .

Dr. Franz Josef Jung, Bundesminister
BMAS . . . . .
```

```
und Folgeresolutionen, zuletzt 2109 (2013)
vom 11. Juli 2013
(Drucksache 18/71) . . . . . 80 C
Dr. Thomas de Maizière, Bundesminister
BMVg . . . . . 80 D
Christoph Strässer (SPD) . . . . . 81 D
```

Abbildung 3.2: Unterschiede in der Darstellung des Inhaltsverzeichnisses

Durch weitere Testdurchläufe wurde ein deutlicher Unterschied in den Inhaltsverzeichniseinträgen gefunden (siehe Abbildung 3.2), womit die Benutzung des trennenden regulären Ausdruckes nicht für alle Dokumente möglich war. Des Weiteren ist die scheinbare Gemeinsamkeit der Punkte nicht in alle Dokumenten gegeben.

3.3 Benamung der Zugehörigkeit

Auch bei Benamung von Zugehörigkeiten gab es viele Unterschiede in der Schreibweise. Bestes Beispiel ist hierfür die Partei „BÜNDNIS 90/DIE GRÜNEN“. Allein für die unterschiedlichen Schreibweisen dieser Partei:

- BÜNDNIS 90/DIE GRÜNEN
- BÜNDNIS 90/
DIE GRÜNEN
- BÜNDNIS 90/
DIE GRÜNEN
- BÜNDNIS 90/DIE GRÜ-
NEN

mussten drei reguläre Ausdrücke abgeändert werden damit sie funktionieren.

3.4 Personensuche im Inhaltsverzeichnis

Der entscheidende Punkt bei der Extraktion war die Suche im Inhaltsverzeichnis nach Personen. Innerhalb der Suche selbst gab es aber Unterschiede die separat abgearbeitet werden mussten. Die Suche wurde mit der Methode `createMap()` in der `SpeechSearch`-Klasse implementiert. Der wörtlich Ablauf der Methode sieht grob wie folgt aus:

Wenn es einen Titel „Geschäftsordnung“ gibt,

dann ist im selben Eintrag auch der erste Name enthalten und muss gespeichert werden

 danach ist jeder Eintrag ein Name,

 bis zum Eintrag der einen Titel enthält.

 nächster Schleifendurchlauf ab dem Titel.

sonst ,

wenn nach einem Titel ein Name folgt

dann ist jeder nachfolgende Eintrag ein Name und muss gespeichert werden,

 bis ein Eintrag einen Titel enthält

 nächster Schleifendurchlauf ab dem Titel.

Wurden zu jedem Titel die Redner gefunden, werden diese Einträge in einer `Map` gespeichert.

4 Schlussbetrachtung

Die grundlegende Anforderung alle Plenarprotokolle der 1.-18. Wahllegislaturperiode wurde in einem Sprint Review Meeting abgeändert, sodass vorrangig die Plenarprotokolle der 18. Wahllegislaturperiode extrahiert werden sollten. Dieses Ziel wurde erreicht, allerdings gibt es Abstriche in Bezug auf die Form der einzelnen Bestandteile der extrahierten Reden im JSON Format.

4.1 split versus substring

In den ersten Versuchen wurden Strings, mithilfe von regulären Ausdrücken, in mindestens zwei Teile zerlegt. Dies hat allerdings zur Folge, dass nach mehreren solcher Zerlegungsprozesse ein Urzustand fast nicht mehr erreicht werden konnte. Daraus folgte die Benutzung von `substring` welcher den Teilstring von Position X-Y zurückgibt ohne dessen Original zu überschreiben.

4.2 Fazit

Prinzipiell wurde innerhalb des Projektes eine spezifizierte KI geschrieben, welche Reden aus der 18. Wahllegislaturperiode anhand der frei zugänglichen XML-Dateien[1] verarbeitet und in JSON-Dateien umwandelt. Das Ergebnis ist verwertbar, kann aber noch verbessert werden in Bezug auf Performanz und Formatierung.

Alles in allem wurde das Projektziel erreicht.

Quellenverzeichnis

- [1] Deutscher Bundestag - Open Data;
URL: <https://www.bundestag.de/services/opendata> (besucht am 19.07.2021).