Kollationen und Transformationen für XML–Dokumente

Dietmar Seipel, Klaus Prätor

Universität Würzburg,
Berlin–Brandenburgische Akademie der Wissenschaften

Kollation von XML–Dokumenten

Michael Wenger hat einen XML–Vergleichsalgorithmus für die philologische Arbeit als Eclipse–Plugin entwickelt.

- Elemente werden basierend auf einem *Ähnlichkeitsmaß* zugeordnet.
- Exakte Übereinstimmungen werden mittels einer Hashsumme ermittelt; falls der Textinhalt gleich geblieben ist, aber neue Auszeichnungen hinzugekommen sind, so kann dies gut mit Hilfe einer *toleranten Hashsumme* festgestellt werden.

Kollation von XML–Dokumenten

- Verfeinerung des Grundalgorithmus durch beliebige *Ähnlichkeitsfilter* und *Strukturfilter*.
- Falls ein Strukturfilter zwei Knoten einander zuordnet, so wird auch versucht die Nachbarknoten innerhalb eines eingeschränkten Teilbereichs des zweiten Dokuments zuzuordnen.
- Die Reihenfolge der Knoten innerhalb des Dokuments wird dabei berücksichtigt.

Kollation von XML–Dokumenten

- Elemente sind aber nicht so stark an ihre Eltern gebunden wie bei klassischen XML–Daten, so daß *Elementverschiebungen* erkannt werden können. Außerdem werden Elementnamen und Attribute nicht so stark gewichtet, so daß in einem gewissen Umfang sogar *geänderte Strukturauszeichnungen* erkannt werden können.
- Als Vorlage für die Ausgabe der *Differenzlisten (Deltas)* zwischen den beiden Dokumenten wird das Ausgabeformat von *XyDiff* benutzt.

Kollation von XML–Dokumenten – Beispiel

```
<p n="40">
  <s n="1">
    Der Igel ist vor dem Hasen.
  </s>
  <s n="2">
    Der Hase isst schneller
    als der Igel.
  </s>
  <s n="3">
    Der Igel gewinnt dennoch.
  </s>
  <s n="4">
    Sehr seltsam.
  </s>
</p>
```

```
<p n="50">
  <s n="1">
    Der Hase ist schneller
    als der Igel.
  </s>
  <s n="2">
    Trotzdem ist der Igel
    vor dem Hasen.
  </s>
</p>
<p n="51">
  <s n="1">
    Der Igel gewinnt dennoch.
  </s>
</p>
```

Transformation von XML–Dokumenten

- Kritische oder wissenschaftliche Editionen sind ein vielversprechndes Anwendungsfeld für die *deklarative Programmierung*. Diese kann das Parsen und das Markup von Texten sowie die Transformation von XML–Dokumenten vereinfachen.
- FNQUERY ist eine deklarative XML–*Anfrage– und Transformationssprache*, welche voll verschränkt ist mit der Programmiersprache PROLOG.
- Die GUI von FNQUERY ermöglicht die *inkrementelle* Entwicklung von Transformationsmethoden.

Nützliche Features von PROLOG

- Spracherweiterungen: Define a *little language* embedded in PROLOG syntax (O'Keefe).
- Definite Clause Grammars (DCGs)
- Backtracking und Unifikation
- Prädikate höherer Ordnung:
  - Möglichkeit alle Resultate zu berechnen
  - Iteratoren und Filter

Orthogonale Transformationstechniken

Wir präsentieren einen neuen, kompakten Substitutionsformalismus, der mit den bekannten DCGs von PROLOG verwoben werden kann:

- *DCGs* zur *Gruppierung* von Elementen auf derselben Ebene des Dokuments und komplexe, verschachtelte Strukturen zu bilden (sequentieller Scan).
- *Subsititutionsregeln* zur Transformation komplexer, verschachtelter Dokumente; sie transformieren ein (baumstrukturiertes) XML–Dokument rekursiv – mit den Blättern beginnend.

Parsing und Transformation

```
<p>Samstag, 27.05.</p>
<p><em>Das Projekt TextGrid</em></p>
<p><em>Kollationierung und Transformation</em></p>
```

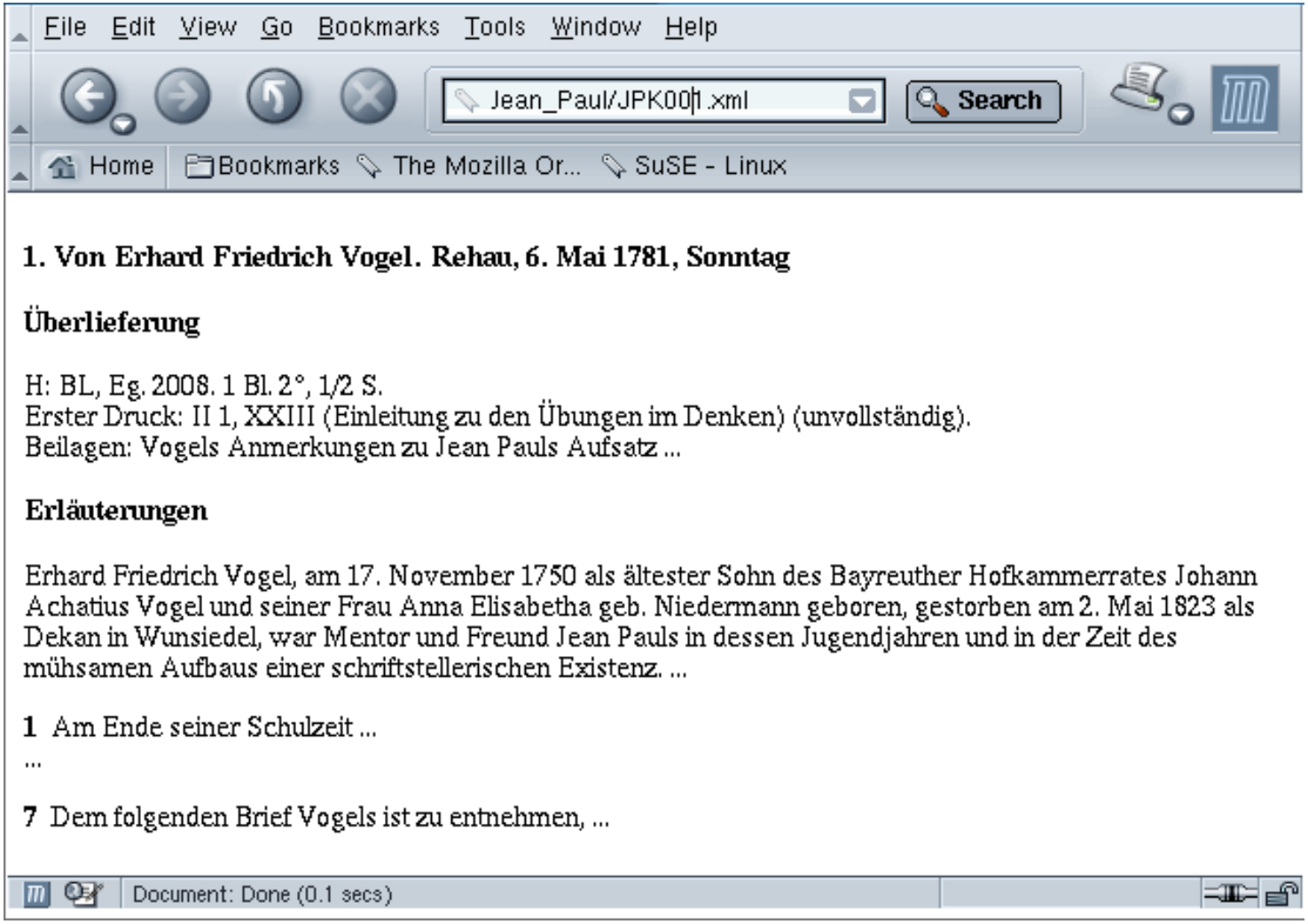
Parsing:

```
<section title="Samstag, 27.05.">
   <p><em>Das Projekt TextGrid</em></p>
   <p><em>...</em></p>
</section>
```

Transformation:

```
<Tag Datum="Samstag, 27.05.">
   <Vortrag>Das Projekt TextGrid</Vortrag>
   <Vortrag>...</Vortrag>
</Tag>
```

Kommentar zu einem Brief aus der Jean Paul–Edition



Konversion: Word $\mapsto$ HTML

```
<p style="margin-top:0;margin-bottom:0;">
   <font face="Times New Roman" size="3">
      <em>202, </em> 6
      <strong>Krebse] </strong>
      Buchhaendlerisch: Remittenden.
   </font>
</p>
```

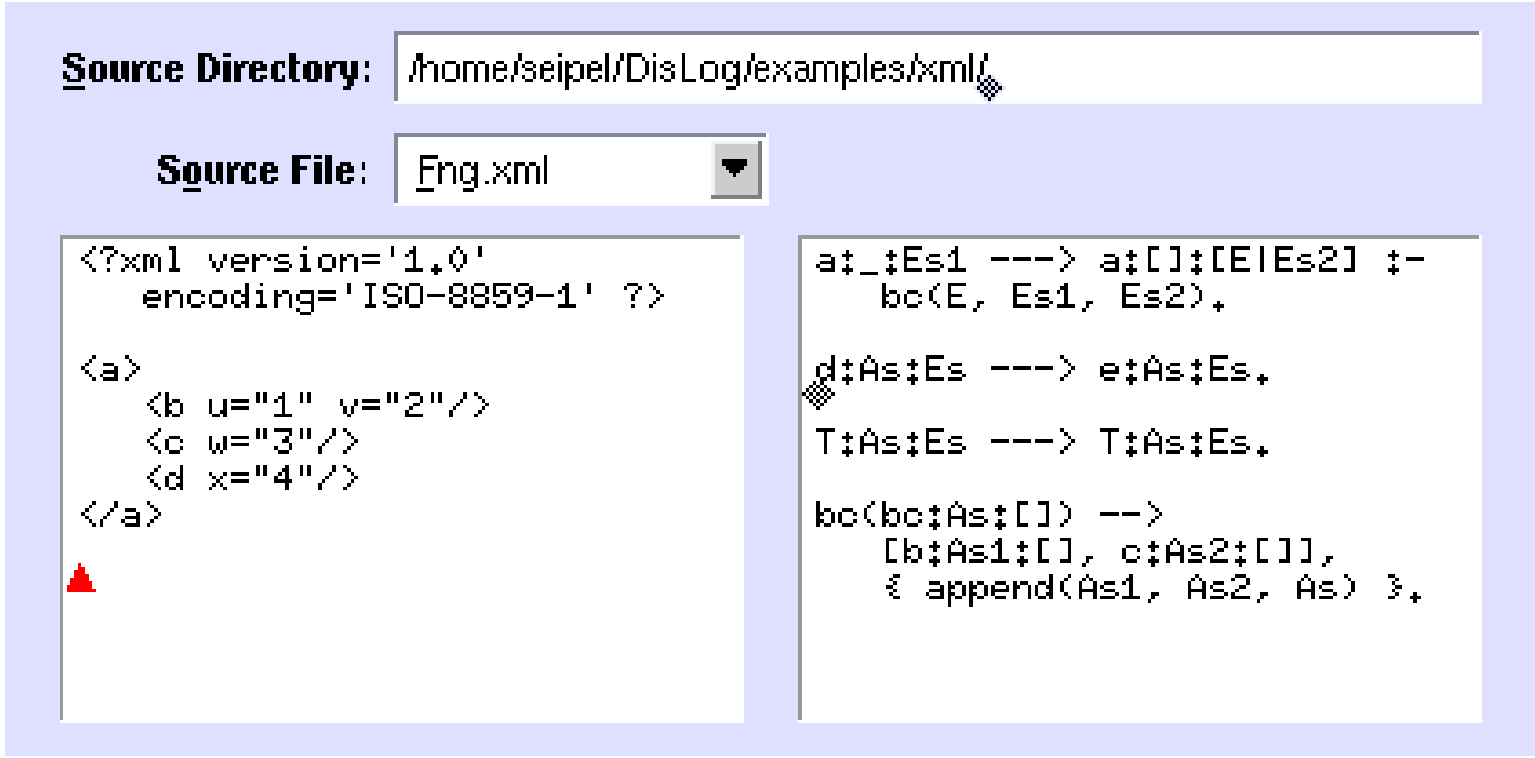
Konversion: HTML ↦ XML / TEI

- die XML–Anfragesprache FNQUERY
  - Pfadausdrücke ⇝ FNPATH
  - Selektion von Elementen/Attributten ⇝ FNSELECT
  - Transformationen wie in XSLT ⇝ FNTRANSFORM
  - Updates ⇝ FNUPDATE
- Parsing mittels DCGs
- graphische Benutzeroberfläche (GUI)

Semantisches Tagging: HTML ↦ XML

```
<comment>
   <commentHead>1. Von Erhard Friedrich Vogel.
      Rehau, 6. Mai 1781, Sonntag
   </commentHead>
   <ednote type="Ueberlieferung">
      <notep>H: BL, Eg. 2008. 1 Bl. 2, 1/2 S.</notep> ...
   </ednote>
   <ednote type="Erlaeuterungen"> ...
      <notep>
         <page>202, </page> 6
         <lemma>Krebse] </lemma>
         Buchhaendlerisch: ...
      </notep>
   </ednote>
</comment>
```

Die GUI von FnQuery (FnTransform)




```
<a>
   <bc u="1" v="2" w="3"/>
   <e x="4"/>
</a>
```

Triple–Repräsentation von XML–Elementen

```
p:[style:'...']:[
   font:[face:'...', size:'3']:[
      em:['202, '], '6', ...]]
```

```
T:As:Es
```

```
<p style="...">
   <font face="..." size="3">
      <em>202, </em> 6 ...
   </font>
</p>
```

Pfadausdrücke in FnUpdate

```
xml_extract_notes(Xml_1, Xml_2) :-
    Xml_2 := Xml_1 <-> [
        ^ednote::[@type='Erlaeuterungen']
        ^note^remark ].
```

```
<ednote type="Erlaeuterungen">
  <note id="7">
    <remark>
      <pos page="1" line="18-19"/>
      <lemma>Gehen</lemma>
      <lemma>sind</lemma>
    </remark>
    Dem folgenden Brief ...
  </note>
</ednote>
```

```
<ednote type="Erlaeuterungen">
  <note id="7">
    Dem folgenden Brief ...
  </note>
</ednote>
```

Zusammenfassung

- Interessant wäre ein Vergleich der verschiedenen Kollationierungsansätze (Wenger, Hemmerich, etc.)
- FnQuery ist eine deklarative Anfrage–, Transformations– und Programmiersprache.
- Wir können das Parsen und Transformieren von Xml–Dokumenten mischen.
- Vor– und Nachbearbeitung bei der Kollationierung