Joachim Gasch

XML Schema driven Database Management of Speech Corpus Metadata

Electronic speech corpora need to bring together several heterogeneous data formats like audio and video data, corpus-, event- and speaker documentation and time aligned media annotations. The metadata management system has to drive data capture, XML native database storage, dynamic publishing and information retrieval processes. This article describes an XML schema based standardization approach where metadata (documentation and annotation information) of different speech corpora is centrally validated and natively stored within an object-relational XML database.

1 XML Data Model

1.1 Introduction

At the beginning of the standardization project, existing metadata standards like the Dublin Core Metadata Initiative (DC)[1], the Open Language Archive Community (OLAC)[2], the Text Encoding Initiative (TEI)[3], the MPEG-7 standard[4] and the ISLE Metadata Initiative (IMDI)[5] as well as the structures of existing in-house speech corpus documentation metadata (DGD)[6] were revised, compared and analyzed. The result set of this analysis was used as the main input during the design phase of two generic XML schemas to define the structures of event- and speaker documentation (cf. Dickgießer, 2008). Media files and time aligned, multi-dimensional media annotations are systematically linked with the documentation metadata (cf. Gasch et al., 2008).

1.2 Generic and project-specific XML Schema Design

For each of the two speech corpus metadata components in chapter 1.1, a generic XML schema was designed modeling a holistic catalogue (repository) of hierarchically well ordered information units. Usability testing and a maximum capacity for future catalogue extensions were two important goals during the XML schema development phase: an XML repository schema includes a base of mandatory elements that guarantee query compatibility across multiple speech corpora (cf. Schiel and Draxler, 2004). Beside these

1 URL: http://dublincore.org
2 URL: http://www.language-archives.org/
3 URL: http://www.tei-c.org/index.xml
4 http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm
5 URL: http://www.mpi.nl/IMDI/
6 http://dsav-oeff.ids-mannheim.de/

mandatory schema elements, a wide set of optional complex elements are introduced in the XML schemas for optional use within different speech corpus projects (cf. figure 1). If an optional complex from the catalogue is activated in the project specific schema of a speech corpus project, the selected complex becomes mandatory in the context of this project.

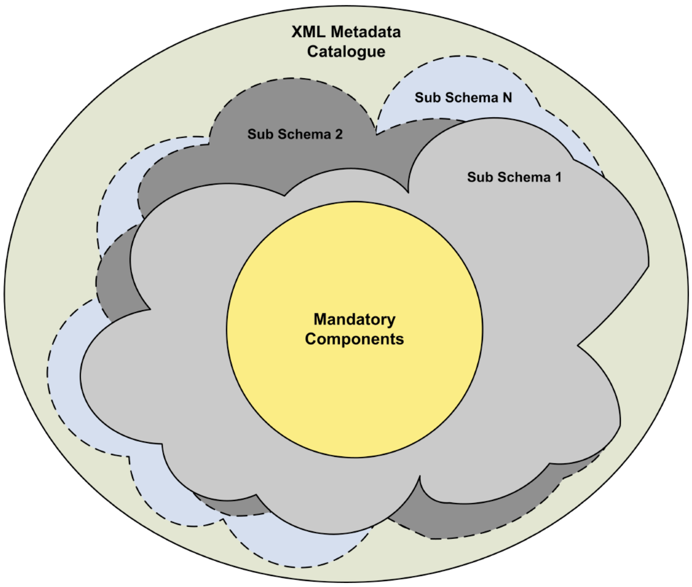


Fig. 1: Derivation of project specific sub schemas from the generic XML schema catalogue (repository)

Once, the project specific subset has been derived from the corresponding generic XML repository schema, the project specific XML schema is fine tuned to ensure maximum data consistency and quality. The following XML schema validation functionalities are implemented and applied during the data entry process:

<u>Enumerations</u>:

Implementation of project specific finite controlled vocabulary lists providing possible field contents using enumerations. Example: the project-specific semantic enrichment provided by this feature guarantees a high level of data quality within the project context. The following example illustrates a project specific country list for the speech corpus project “German Today” (cf. Brinckmann et al., 2008):

```
   <xs:element name="Land">
        <xs:annotation>
                 <xs:documentation>[ids mandatory] country, where the event took place
(taken from the country list ISO 3166-1)</xs:documentation>
        </xs:annotation>
        <xs:simpleType>
                 <xs:restriction base="xs:string">
                          <xs:minLength value="1"/>
                          <xs:enumeration value="Deutschland"/>
                          <xs:enumeration value="Österreich"/>
                          <xs:enumeration value="Schweiz"/>
                          <xs:enumeration value="Italien"/>
                          <xs:enumeration value="Liechtenstein"/>
                          <xs:enumeration value="Belgien"/>
                          <xs:enumeration value="Luxemburg"/>
                 </xs:restriction>
        </xs:simpleType>
   </xs:element>
```

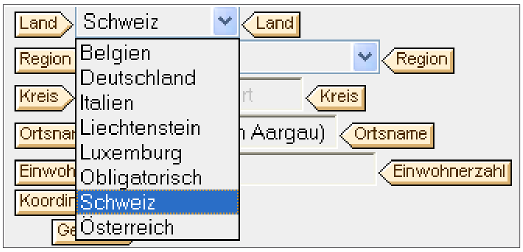
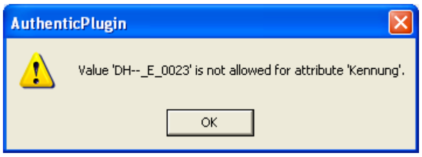


Fig. 2: Graphical editor representation of a controlled vocabulary list

Regular Expressions:

We define patterns for element contents, for example to validate unique event IDs. A regular expression is implemented for the ID attribute of the event element (combined with a unique XML database index on the ID attribute value). The regular expression example defines a pattern for the event IDs expecting a zero leading five digit number at the end. In the example the XML parser of the editor returns a validation error because the ending number of the event ID contains only four digits:

```
   <xs:attribute name="Kennung" use="required">
        <xs:simpleType>
                 <xs:restriction base="xs:string">
                          <xs:pattern value="DH--_E_\d{5}"/>
                 </xs:restriction>
        </xs:simpleType>
   </xs:attribute>
```


AuthenticPlugin

Value 'DH--_E_0023' is not allowed for attribute 'Kennung'.

Fig. 3: Regular expression validation error

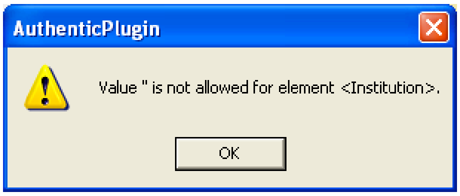
Recursive content (multiple occurrences):

The possible number of occurrences of complex elements is controlled by the element attributes “minOccurs” respectively “maxOccurs”. The element <Sprecher> (speaker) is mandatory, and can be repeated as many times as needed:

```
<xs:element name="Sprecher" minOccurs="1" maxOccurs="unbounded">
```

Mandatory non-empty elements:

New XML document instances are generated using default values for element contents. In this way the XML editor permits the user to validate the document against the XML schema to identify possible errors at any time during data entry. For example an element that is not allowed by the schema to be empty would produce an error message when the document is validated having the element empty:


AuthenticPlugin

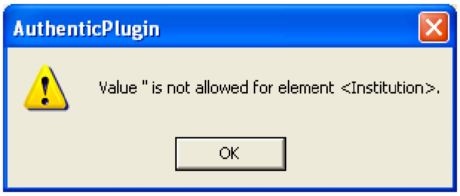
Value '' is not allowed for element <Institution>.

Fig. 4: Empty element validation error

Content data types:

We implementation exact data format definitions like for example date format specifications for date fields (YYYY-MM-DD) or the numeric range of the decimal format for geocode information. The following is an example of an XML parser validation error for a geographic latitude value that contains characters not permitted by the field format:

```
<xs:element name="Geografische_Breite" default="000.0">
     <xs:annotation>
             <xs:documentation>[mandatory] geografic latitude of the event place in
decimal degree</xs:documentation>
     </xs:annotation>
     <xs:simpleType>
             <xs:restriction base="xs:decimal">
                     <xs:minInclusive value="-180.0"/>
                     <xs:maxInclusive value="180.0"/>
             </xs:restriction>
     </xs:simpleType>
</xs:element>
```

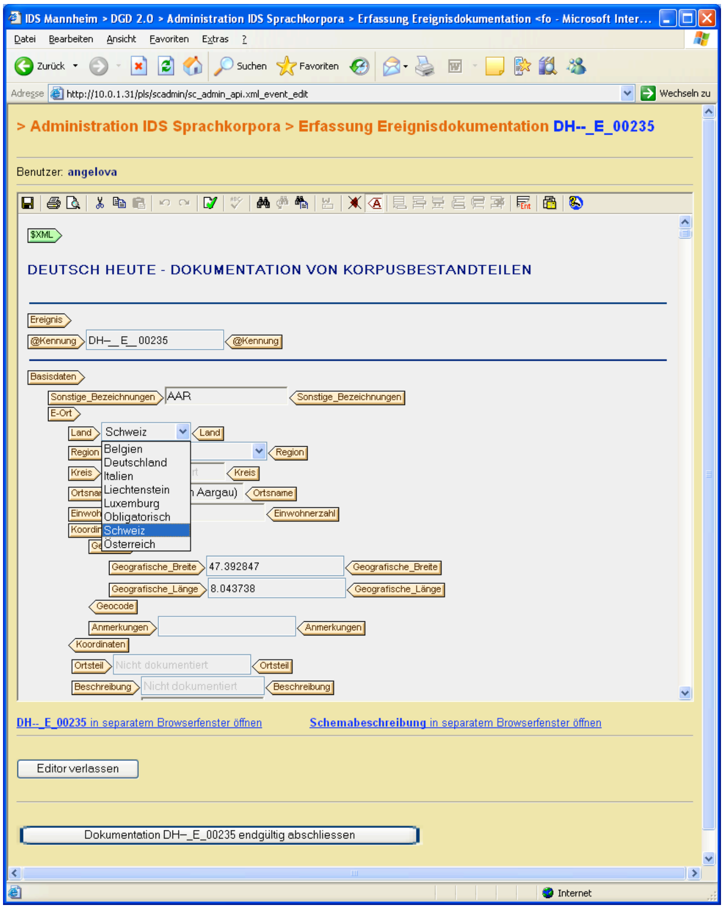

AuthenticPlugin

Value '' is not allowed for element <Institution>.

Fig. 5: Data type validation error

2 Native XML Database Storage and Processing

2.1 XML Data Entry


IDS Mannheim > DGD 2.0 > Administration IDS Sprachkorpora > Erfassung Ereignisdokumentation <fo - Microsoft Inter...
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück
Suchen
Favoriten
Adresse http://10.0.1.31/pls/scadmin/sc_admin_api.xml_event_edit
Wechseln zu
Internet

> Administration IDS Sprachkorpora > Erfassung Ereignisdokumentation DH--_E_00235

Benutzer: angelova

$XML

DEUTSCH HEUTE - DOKUMENTATION VON KORPUSBESTANDTEILEN

Ereignis
@Kennung DH--_E_00235 @Kennung

Basisdaten
Sonstige_Bezeichnungen AAR Sonstige_Bezeichnungen
E-Ort
Land Schweiz Land
Belgien
Deutschland
Italien
Liechtenstein
Luxemburg
Obligatorisch
Schweiz
Österreich
Region Region
Kreis rt Kreis
Ortsnar Aargau) Ortsname
Einwoh Einwohnerzahl
Koordin
Ge
Geografische_Breite 47.392847 Geografische_Breite
Geografische_Länge 8.043738 Geografische_Länge
Geocode
Anmerkungen Anmerkungen
Koordinaten
Ortsteil Nicht dokumentiert Ortsteil
Beschreibung Nicht dokumentiert Beschreibung

DH--_E_00235 in separatem Browserfenster öffnen   Schemabeschreibung in separatem Browserfenster öffnen

Editor verlassen

Dokumentation DH--_E_00235 endgültig abschliessen

Fig. 6: Web based XML editor client/server application

XML instances basically can originate from two different sources. They can be manually created using an XML editor application. Or the metadata might already be available in another machine readable source format. In case of pre-existing electronic metadata, an automated bulk data mapping process can be set up to transform, validate and import the generated XML data into the XML database. For metadata exchange with other institutions an automated export mapping process can be set up.

The XML instances are locally validated against their corresponding project-specific XML sub schema using the validating XML editor application or directly inside the source data mapping process. During the import process of the Oracle XML database, the schema references of the XML instances are changed from the project-specific sub schema to the catalogue schema location. In this way XML instances that were created with different project-specific schemas are all validated against the same XML catalogue schema.

We use a browser-based schema validating XML authoring solution[7] to edit the XML documentation instances. The XML editor browser plug-in is driven by a web based client/server application implemented in Oracle PL/SQL. The application controls personalized user access, unique document ID assignment and document workflow. The XML Editor provides an ergonomic user support during XML data entry and also guarantees syntactic and semantic data quality due to the customization of the underlying project specific XML schemas (cf. figure 6).

2.2 Oracle XML Database Storage

Once the XML instances have been finally revised, the documents are ready for import into the XML database. The Oracle 11g object relational XML database[8] basically distinguishes three different types of internal processing and storage of XML instances: unstructured, binary and structured XML database storage[9].

In the unstructured storage case, the XML database only checks if a document is syntactically well formed without validating it against an XML schema and writes the document instance into an XMLType CLOB (character large object) field. When retrieving document content, the Document Object Model (DOM) has to be built first in memory for each XML instance. This makes this alternative very time consuming for large amounts of data and complex XML structures so that it should be used especially for document-centric cases.

With the binary XML storage approach, the xml data is first syntactically parsed and the result of the parsing process is then stored in a post-parse, binary format. The binary stored DOM images of the parsing results highly increase the performance of XML information retrieval processes for document-centric scenarios.

We decided to use the structured storage approach (schema based processing of the XML data) to store documentation instances (cf. figure 7). With this approach, first the XML schema is registered in the database and an XMLType table is generated according to the XML schema. Then an internal object relational database structure is created that models the DOM types of the XML instances as described in the XML schema. The XML instances are semantically validated against their XML schema during the import process (XML parser). Using this object relational approach, content retrieval for data-centric structures is much faster than with unstructured storage because the DOMs do not

7 Altova Authentic (Browser Edition): http://www.altova.com/products/authentic/web_based_xml_forms.html

8 Oracle XML DB, URL: http://www.oracle.com/technology/ tech/xml/xmldb/index.html

9 Oracle XML DB storage options - unstructured, binary and structured storage: http://download.oracle.com/docs/cd/B28359_01/appdev.111/b28369/xdb01int.htm#BABECDCF

have to be built at runtime. In contrary to the binary approach, with structured storage all XML instances must be compliant with the underlying XML schema.

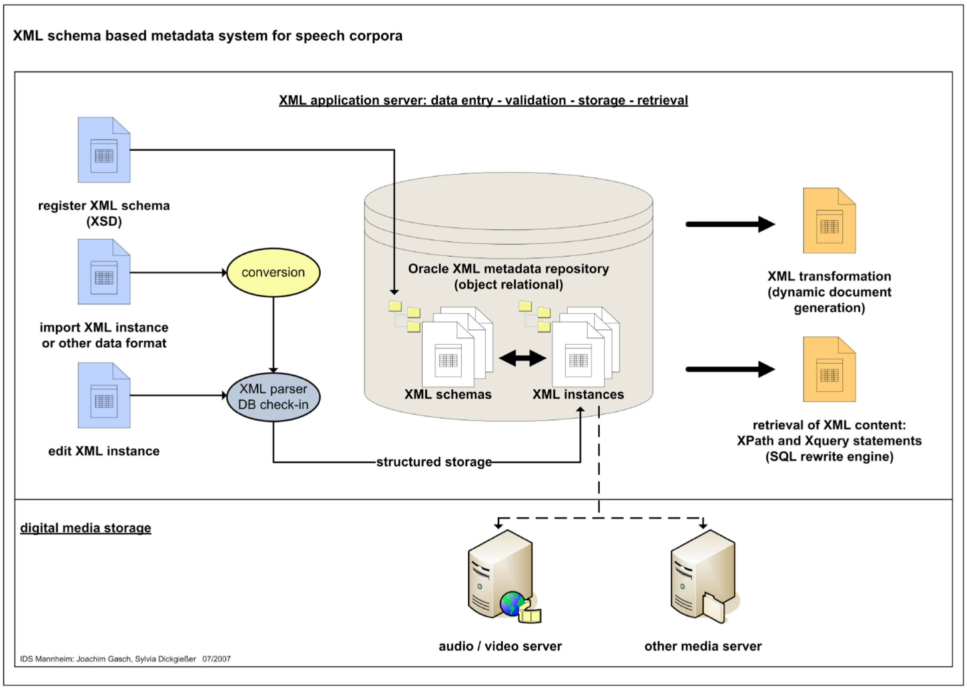

XML schema based metadata system for speech corpora
XML application server: data entry - validation - storage - retrieval
register XML schema (XSD)
import XML instance or other data format
edit XML instance
conversion
XML parser DB check-in
Oracle XML metadata repository (object relational)
XML schemas
XML instances
structured storage
XML transformation (dynamic document generation)
retrieval of XML content: XPath and Xquery statements (SQL rewrite engine)
digital media storage
audio / video server
other media server
IDS Mannheim: Joachim Gasch, Sylvia Dickgießer 07/2007

Figure 7: Oracle 11g XML database (structured storage model)

2.3 XML Information Retrieval

2.3.1 XQuery Language

The XML schema aware processing of XML instances enables the implementation of context sensitive query interfaces. As the complete information structure is well known, informational units can be precisely retrieved in their semantic contexts. XQuery works with XML documents in a similar way as the Standard Query Language (SQL) is used to retrieve information from table structures in the relational database world.

Within XML technology, XQuery[10] (as an extension of XPath) is the W3C recommendation providing the correspondent structure navigation and retrieval functionalities to extract information from complex XML structures. XPath queries are kind of limited in the case of XML documents containing iterative structures, cf. the following example XML fragment:

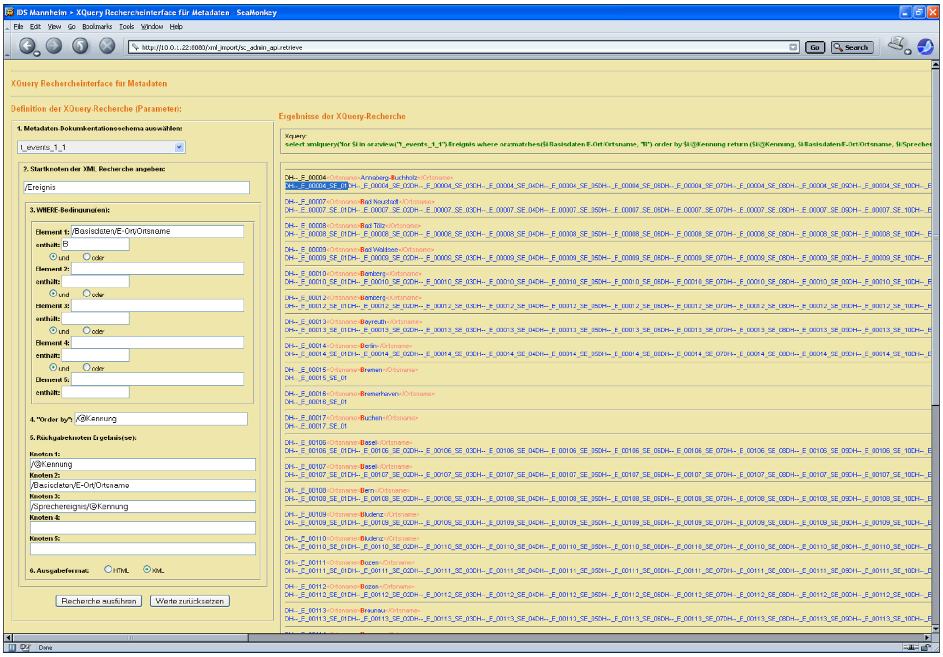
10 W3C, XQuery 1.0: An XML Query Language, URL: http://www.w3.org/TR/xquery/

```
<node_1>
     <element_1/>
     <element_1/>
     <element_1/>
     ...
</node_1>
```

Getting the content of all <element_1> tags using an XPath query is only possible, if there exists a unique distinctive attribute value inside the <element_1> tag specifying the concrete tag the query is referring to. Otherwise the XPath expression returns an error message saying that the query returns multiple not distinctive results. It is at this point where XQuery FLWOR[11] expressions get of special interest. A FLWOR XQuery statement allows us to loop through iterative even non distinctive XML substructures. The "for" construct produces intermediate results as temporary tuples of bound variables called a tuple stream, even if the repetitive structures are not distinct on element level.

The following describes an XQuery example containing the iterative structure speech event taken from the speech corpus DH:

```
select xmlquery('
     for $i in ora:view("t_events_1_1")/Ereignis
     order by $i/@Kennung
     return ($i/@Kennung, $i/Basisdaten/E-Ort/Ortsname,$i/Sprechereignis/@Kennung
)' returning content) from dual;
```


IDS Mannheim > XQuery Rechercheinterface für Metadaten - SeaMonkey
File Edit View Go Bookmarks Tools Window Help
http://10.0.1.22:8080/xml_import/sc_admin_api.retrieve
Go
Search
Done

XQuery Rechercheinterface für Metadaten

Definition der XQuery-Recherche (Parameter):

1. Metadaten-Dokumkentationsschema auswählen: t_events_1_1

2. Startknoten der XML Recherche angeben: /Ereignis

3. WHERE-Bedingung(en):

Element 1: /Basisdaten/E-Ort/Ortsname
enthält: B
[✓ und] oder
Element 2:
enthält:
[✓ und] oder
Element 3:
enthält:
[✓ und] oder
Element 4:
enthält:
[✓ und] oder
Element 5:
enthält:

4. "Order by": /@Kennung

5. Rückgabeknoten Ergebnis(se):

Knoten 1: /@Kennung
Knoten 2: /Basisdaten/E-Ort/Ortsname
Knoten 3: /Sprechereignis/@Kennung
Knoten 4:
Knoten 5:

6. Ausgabeformat: HTML [✓ XML]

Recherche ausführen | Werte zurücksetzen

Ergebnisse der XQuery-Recherche

Xquery:
select xmlquery('for $i in ora:view("t_events_1_1")/Ereignis where ora:matches($i/Basisdaten/E-Ort/Ortsname, "B") order by $i/@Kennung return ($i/@Kennung, $i/Basisdaten/E-Ort/Ortsname, $i/Sprecher

```
DH--_E_00004<Ortsname>Arnaberg-Buchholz</Ortsname>
DH--_E_00004_SE_01DH--_E_00004_SE_02DH--_E_00004_SE_03DH--_E_00004_SE_04DH--_E_00004_SE_05DH--_E_00004_SE_06DH--_E_00004_SE_07DH--_E_00004_SE_08DH--_E_00004_SE_09DH--_E_00004_SE_10DH--_E

DH--_E_00007<Ortsname>Bad Neustadt</Ortsname>
DH--_E_00007_SE_01DH--_E_00007_SE_02DH--_E_00007_SE_03DH--_E_00007_SE_04DH--_E_00007_SE_05DH--_E_00007_SE_06DH--_E_00007_SE_07DH--_E_00007_SE_08DH--_E_00007_SE_09DH--_E_00007_SE_10DH--_E

DH--_E_00008<Ortsname>Bad Tölz</Ortsname>
DH--_E_00008_SE_01DH--_E_00008_SE_02DH--_E_00008_SE_03DH--_E_00008_SE_04DH--_E_00008_SE_05DH--_E_00008_SE_06DH--_E_00008_SE_07DH--_E_00008_SE_08DH--_E_00008_SE_09DH--_E_00008_SE_10DH--_E

DH--_E_00009<Ortsname>Bad Waldsee</Ortsname>
DH--_E_00009_SE_01DH--_E_00009_SE_02DH--_E_00009_SE_03DH--_E_00009_SE_04DH--_E_00009_SE_05DH--_E_00009_SE_06DH--_E_00009_SE_07DH--_E_00009_SE_08DH--_E_00009_SE_09DH--_E_00009_SE_10DH--_E

DH--_E_00010<Ortsname>Bamberg</Ortsname>
DH--_E_00010_SE_01DH--_E_00010_SE_02DH--_E_00010_SE_03DH--_E_00010_SE_04DH--_E_00010_SE_05DH--_E_00010_SE_06DH--_E_00010_SE_07DH--_E_00010_SE_08DH--_E_00010_SE_09DH--_E_00010_SE_10DH--_E

DH--_E_00012<Ortsname>Bamberg</Ortsname>
DH--_E_00012_SE_01DH--_E_00012_SE_02DH--_E_00012_SE_03DH--_E_00012_SE_04DH--_E_00012_SE_05DH--_E_00012_SE_06DH--_E_00012_SE_07DH--_E_00012_SE_08DH--_E_00012_SE_09DH--_E_00012_SE_10DH--_E

DH--_E_00013<Ortsname>Bayreuth</Ortsname>
DH--_E_00013_SE_01DH--_E_00013_SE_02DH--_E_00013_SE_03DH--_E_00013_SE_04DH--_E_00013_SE_05DH--_E_00013_SE_06DH--_E_00013_SE_07DH--_E_00013_SE_08DH--_E_00013_SE_09DH--_E_00013_SE_10DH--_E

DH--_E_00014<Ortsname>Berlin</Ortsname>
DH--_E_00014_SE_01DH--_E_00014_SE_02DH--_E_00014_SE_03DH--_E_00014_SE_04DH--_E_00014_SE_05DH--_E_00014_SE_06DH--_E_00014_SE_07DH--_E_00014_SE_08DH--_E_00014_SE_09DH--_E_00014_SE_10DH--_E

DH--_E_00015<Ortsname>Bremen</Ortsname>
DH--_E_00015_SE_01

DH--_E_00016<Ortsname>Bremerhaven</Ortsname>
DH--_E_00016_SE_01

DH--_E_00017<Ortsname>Buchen</Ortsname>
DH--_E_00017_SE_01

DH--_E_00106<Ortsname>Basel</Ortsname>
DH--_E_00106_SE_01DH--_E_00106_SE_02DH--_E_00106_SE_03DH--_E_00106_SE_04DH--_E_00106_SE_05DH--_E_00106_SE_06DH--_E_00106_SE_07DH--_E_00106_SE_08DH--_E_00106_SE_09DH--_E_00106_SE_10DH--_E

DH--_E_00107<Ortsname>Basel</Ortsname>
DH--_E_00107_SE_01DH--_E_00107_SE_02DH--_E_00107_SE_03DH--_E_00107_SE_04DH--_E_00107_SE_05DH--_E_00107_SE_06DH--_E_00107_SE_07DH--_E_00107_SE_08DH--_E_00107_SE_09DH--_E_00107_SE_10DH--_E

DH--_E_00108<Ortsname>Bern</Ortsname>
DH--_E_00108_SE_01DH--_E_00108_SE_02DH--_E_00108_SE_03DH--_E_00108_SE_04DH--_E_00108_SE_05DH--_E_00108_SE_06DH--_E_00108_SE_07DH--_E_00108_SE_08DH--_E_00108_SE_09DH--_E_00108_SE_10DH--_E

DH--_E_00109<Ortsname>Bludenz</Ortsname>
DH--_E_00109_SE_01DH--_E_00109_SE_02DH--_E_00109_SE_03DH--_E_00109_SE_04DH--_E_00109_SE_05DH--_E_00109_SE_06DH--_E_00109_SE_07DH--_E_00109_SE_08DH--_E_00109_SE_09DH--_E_00109_SE_10DH--_E

DH--_E_00110<Ortsname>Bludenz</Ortsname>
DH--_E_00110_SE_01DH--_E_00110_SE_02DH--_E_00110_SE_03DH--_E_00110_SE_04DH--_E_00110_SE_05DH--_E_00110_SE_06DH--_E_00110_SE_07DH--_E_00110_SE_08DH--_E_00110_SE_09DH--_E_00110_SE_10DH--_E

DH--_E_00111<Ortsname>Bozen</Ortsname>
DH--_E_00111_SE_01DH--_E_00111_SE_02DH--_E_00111_SE_03DH--_E_00111_SE_04DH--_E_00111_SE_05DH--_E_00111_SE_06DH--_E_00111_SE_07DH--_E_00111_SE_08DH--_E_00111_SE_09DH--_E_00111_SE_10DH--_E

DH--_E_00112<Ortsname>Bozen</Ortsname>
DH--_E_00112_SE_01DH--_E_00112_SE_02DH--_E_00112_SE_03DH--_E_00112_SE_04DH--_E_00112_SE_05DH--_E_00112_SE_06DH--_E_00112_SE_07DH--_E_00112_SE_08DH--_E_00112_SE_09DH--_E_00112_SE_10DH--_E

DH--_E_00113<Ortsname>Braunau</Ortsname>
DH--_E_00113_SE_01DH--_E_00113_SE_02DH--_E_00113_SE_03DH--_E_00113_SE_04DH--_E_00113_SE_05DH--_E_00113_SE_06DH--_E_00113_SE_07DH--_E_00113_SE_08DH--_E_00113_SE_09DH--_E_00113_SE_10DH--_E
```

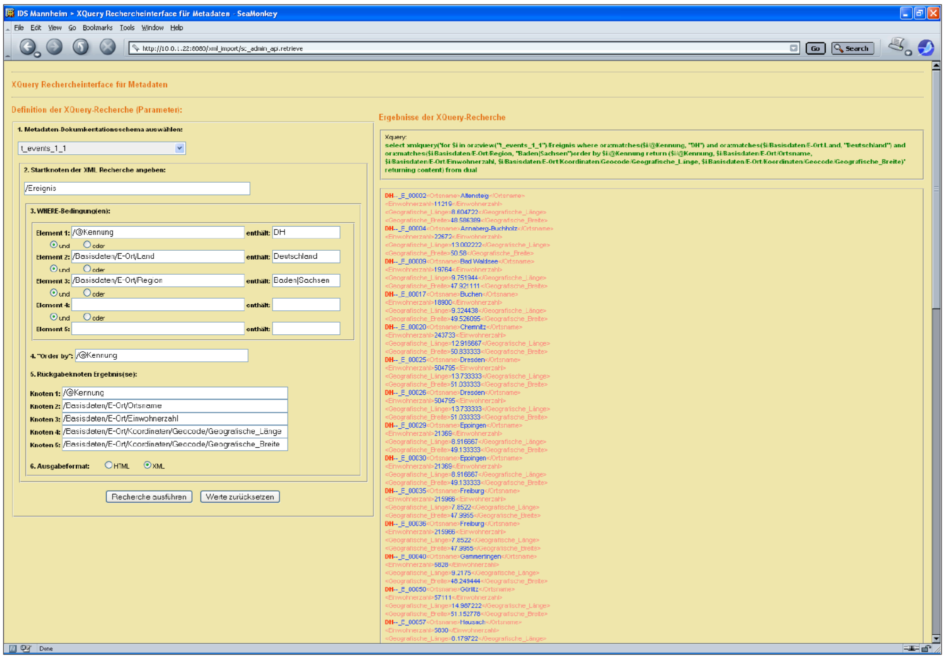
Fig. 8: XQuery retrieval interface (iterative XML structures)

11 FLWOR (abbreviation): for, let, where, order by, return

In the above example we select the event IDs, the city names and the iterative speech event IDs for all events that have city names starting with “B”. This example illustrates that iterative speech event IDs are retrieved by the XQuery statement in the same way as it is the case for the unique event IDs.

As already mentioned, XQuery statements provide very powerful functionality for a precise extraction of information pieces out of their semantic contexts. The following XQuery example extracts a complete geographic event information subset from the speech corpus DH:

```
select xmlquery(‘
     for $i in ora:view(“t_events_1_1”)/Ereignis
     where ora:matches($i/@Kennung, “DH”)
     and ora:matches($i/Basisdaten/E-Ort/Land, “Deutschland”)
     and ora:matches($i/Basisdaten/E-Ort/Region, “Baden|Sachsen”)
     order by $i/@Kennung
     return ($i/@Kennung, $i/Basisdaten/E-Ort/Ortsname,
     $i/Basisdaten/E-Ort/Einwohnerzahl,
     $i/Basisdaten/E-Ort/Koordinaten/Geocode/Geografische_Länge,
     $i/Basisdaten/E-Ort/Koordinaten/Geocode/Geografische_Breite
)‘ returning content) from dual
```


IDS Mannheim > XQuery Rechercheinterface für Metadaten - SeaMonkey
File Edit View Go Bookmarks Tools Window Help
http://10.0.1.22:8080/xml_import/sc_admin_api.retrieve
Go
Search
Recherche ausführen
Werte zurücksetzen
Done

XQuery Rechercheinterface für Metadaten

Definition der XQuery-Recherche (Parameter):

1. Metadaten-Dokumkentationsschema auswählen: t_events_1_1

2. Startknoten der XML Recherche angeben: /Ereignis

3. WHERE-Bedingung(en):

Element 1: /@Kennung enthält: DH
[✓ und] oder
Element 2: /Basisdaten/E-Ort/Land enthält: Deutschland
[✓ und] oder
Element 3: /Basisdaten/E-Ort/Region enthält: Baden|Sachsen
[✓ und] oder
Element 4: enthält:
[✓ und] oder
Element 5: enthält:

4. "Order by": /@Kennung

5. Rückgabeknoten Ergebnis(se):

Knoten 1: /@Kennung
Knoten 2: /Basisdaten/E-Ort/Ortsname
Knoten 3: /Basisdaten/E-Ort/Einwohnerzahl
Knoten 4: /Basisdaten/E-Ort/Koordinaten/Geocode/Geografische_Länge
Knoten 5: /Basisdaten/E-Ort/Koordinaten/Geocode/Geografische_Breite

6. Ausgabeformat: HTML [✓ XML]

Ergebnisse der XQuery-Recherche

Xquery:
select xmlquery('for $i in ora:view("t_events_1_1")/Ereignis where ora:matches($i/@Kennung, "DH") and ora:matches($i/Basisdaten/E-Ort/Land, "Deutschland") and ora:matches($i/Basisdaten/E-Ort/Region, "Baden|Sachsen")order by $i/@Kennung return ($i/@Kennung, $i/Basisdaten/E-Ort/Ortsname, $i/Basisdaten/E-Ort/Einwohnerzahl, $i/Basisdaten/E-Ort/Koordinaten/Geocode/Geografische_Länge, $i/Basisdaten/E-Ort/Koordinaten/Geocode/Geografische_Breite)' returning content) from dual

```
DH--_E_00002<Ortsname>Altensteig</Ortsname>
<Einwohnerzahl>11219</Einwohnerzahl>
<Geografische_Länge>8.604722</Geografische_Länge>
<Geografische_Breite>48.586389</Geografische_Breite>
DH--_E_00004<Ortsname>Annaberg-Buchholz</Ortsname>
<Einwohnerzahl>22672</Einwohnerzahl>
<Geografische_Länge>13.002222</Geografische_Länge>
<Geografische_Breite>50.58</Geografische_Breite>
DH--_E_00009<Ortsname>Bad Waldsee</Ortsname>
<Einwohnerzahl>19764</Einwohnerzahl>
<Geografische_Länge>9.751944</Geografische_Länge>
<Geografische_Breite>47.921111</Geografische_Breite>
DH--_E_00017<Ortsname>Buchen</Ortsname>
<Einwohnerzahl>18900</Einwohnerzahl>
<Geografische_Länge>9.324438</Geografische_Länge>
<Geografische_Breite>49.526095</Geografische_Breite>
DH--_E_00020<Ortsname>Chemnitz</Ortsname>
<Einwohnerzahl>243733</Einwohnerzahl>
<Geografische_Länge>12.916667</Geografische_Länge>
<Geografische_Breite>50.833333</Geografische_Breite>
DH--_E_00025<Ortsname>Dresden</Ortsname>
<Einwohnerzahl>504795</Einwohnerzahl>
<Geografische_Länge>13.733333</Geografische_Länge>
<Geografische_Breite>51.033333</Geografische_Breite>
DH--_E_00026<Ortsname>Dresden</Ortsname>
<Einwohnerzahl>504795</Einwohnerzahl>
<Geografische_Länge>13.733333</Geografische_Länge>
<Geografische_Breite>51.033333</Geografische_Breite>
DH--_E_00029<Ortsname>Eppingen</Ortsname>
<Einwohnerzahl>21369</Einwohnerzahl>
<Geografische_Länge>8.916667</Geografische_Länge>
<Geografische_Breite>49.133333</Geografische_Breite>
DH--_E_00030<Ortsname>Eppingen</Ortsname>
<Einwohnerzahl>21369</Einwohnerzahl>
<Geografische_Länge>8.916667</Geografische_Länge>
<Geografische_Breite>49.133333</Geografische_Breite>
DH--_E_00035<Ortsname>Freiburg</Ortsname>
<Einwohnerzahl>215966</Einwohnerzahl>
<Geografische_Länge>7.8522</Geografische_Länge>
<Geografische_Breite>47.9955</Geografische_Breite>
DH--_E_00036<Ortsname>Freiburg</Ortsname>
<Einwohnerzahl>215966</Einwohnerzahl>
<Geografische_Länge>7.8522</Geografische_Länge>
<Geografische_Breite>47.9955</Geografische_Breite>
DH--_E_00040<Ortsname>Gammertingen</Ortsname>
<Einwohnerzahl>6828</Einwohnerzahl>
<Geografische_Länge>9.2175</Geografische_Länge>
<Geografische_Breite>48.249444</Geografische_Breite>
DH--_E_00050<Ortsname>Görlitz</Ortsname>
<Einwohnerzahl>57111</Einwohnerzahl>
<Geografische_Länge>14.987222</Geografische_Länge>
<Geografische_Breite>51.152778</Geografische_Breite>
DH--_E_00057<Ortsname>Hausach</Ortsname>
<Einwohnerzahl>5830</Einwohnerzahl>
<Geografische_Länge>8.179722</Geografische_Länge>
```

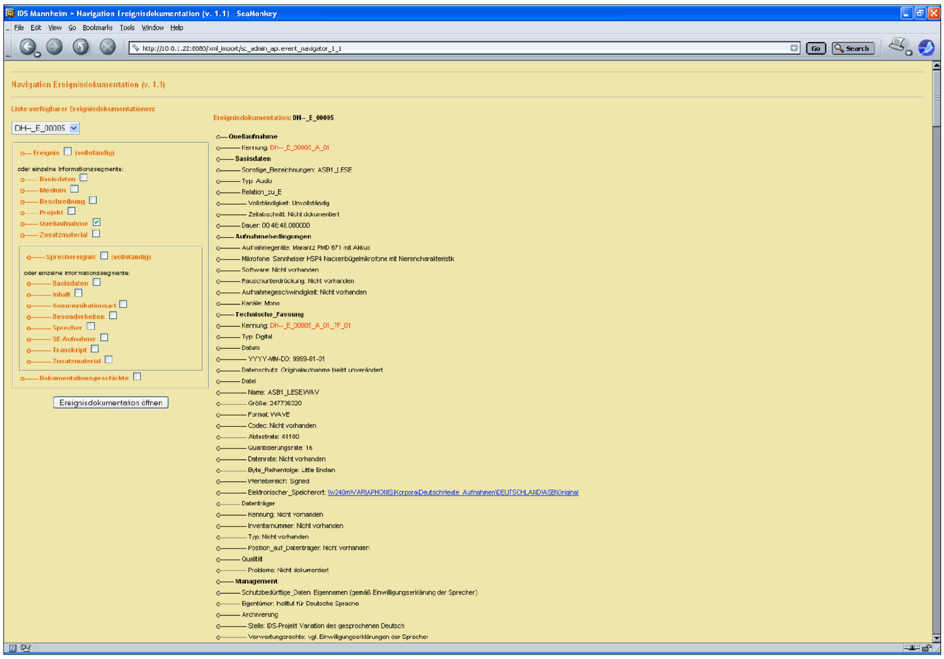
Fig. 9: XQuery retrieval interface (geographic information subsets)

In the example we select the event IDs, the town names, the numbers of inhabitants and the geocodes for all events that have been recorded in Germany in the federal states of Baden-Württemberg or Sachsen and that belong to the speech corpus “DH”. With the result set of this query the dynamic creation of geographic maps is planned for the near future.

2.3.2 XML Full-Text Search

XQuery statements provide powerful retrieval functionality for experienced users. But one great disadvantage of this technology is their requirement that the user has a detailed knowledge of the information structure to perform a query[12]. Users that are new to the system certainly prefer a simple full-text search functionality to start exploring speech corpus metadata documents without knowing their exact structure. The full-text search module provided therefore ignores the XML tag structure and only considers pure element content. In the same way as for relational tables, the Oracle XML database also operates an Oracle Context Index (CTXSYS.CONTEXT) that can be applied for full-text searches over natively stored XML instances.

2.4 Online Publishing


IDS Mannheim > Navigation Ereignisdokumentation (v. 1.1) - SeaMonkey
File Edit View Go Bookmarks Tools Window Help
http://10.0.1.22:8080/xml_import/sc_admin_api.event_navigator_1_1
Go
Search

Navigation Ereignisdokumentation (v. 1.1)

Liste verfügbarer Ereignisdokumentationen:

DH--_E_00005

Ereignis ☐ (vollständig)
oder einzelne Informationssegmente:
Basisdaten ☐
Medium ☐
Beschreibung ☐
Projekt ☐
Quellaufnahme ☑
Zusatzmaterial ☐

Sprechereignis ☐ (vollständig)
oder einzelne Informationssegmente:
Basisdaten ☐
Inhalt ☐
Kommunikationsart ☐
Besonderheiten ☐
Sprecher ☐
SE-Aufnahme ☐
Transkript ☐
Zusatzmaterial ☐

Dokumentationsgeschichte ☐

Ereignisdokumentation öffnen

Ereignisdokumentation: DH--_E_00005

```
o— Quellaufnahme
o———— Kennung: DH--_E_00005_A_01
o——— Basisdaten
o———— Sonstige_Bezeichnungen: ASB1_LESE
o———— Typ: Audio
o———— Relation_zu_E
o————— Vollständigkeit: Unvollständig
o————— Zeitabschnitt: Nicht dokumentiert
o———— Dauer: 00:46:48.000000
o——— Aufnahmebedingungen
o———— Aufnahmegeräte: Marantz PMD 671 mit Akkus
o———— Mikrofone: Sennheiser HSP4 Nackenbügelmikrofone mit Nierencharakteristik
o———— Software: Nicht vorhanden
o———— Rauschunterdrückung: Nicht vorhanden
o———— Aufnahmegeschwindigkeit: Nicht vorhanden
o———— Kanäle: Mono
o——— Technische_Fassung
o———— Kennung: DH--_E_00005_A_01_TF_01
o———— Typ: Digital
o———— Datum
o————— YYYY-MM-DD: 9999-01-01
o———— Datenschutz: Originalaufnahme bleibt unverändert
o———— Datei
o————— Name: ASB1_LESE.WAV
o————— Größe: 247736320
o————— Format: WAVE
o————— Codec: Nicht vorhanden
o————— Abtastrate: 44100
o————— Quantisierungsrate: 16
o————— Datenrate: Nicht vorhanden
o————— Byte_Reihenfolge: Little Endian
o————— Wertebereich: Signed
o————— Elektronischer_Speicherort: \\v240m\VARIAPHONIS\Korpora\DeutschHeute_Aufnahmen\DEUTSCHLAND\ASB\Original
o———— Datenträger
o————— Kennung: Nicht vorhanden
o————— Inventarnummer: Nicht vorhanden
o————— Typ: Nicht vorhanden
o————— Position_auf_Datenträger: Nicht vorhanden
o———— Qualität
o————— Probleme: Nicht dokumentiert
o——— Management
o———— Schutzbedürftige_Daten: Eigennamen (gemäß Einwilligungserklärung der Sprecher)
o———— Eigentümer: Institut für Deutsche Sprache
o———— Archivierung
o————— Stelle: IDS-Projekt Variation des gesprochenen Deutsch
o————— Verwertungsrechte: vgl. Einwilligungserklärungen der Sprecher
```

Fig. 10: Browser application for customizable speech event navigation

The centrally bundled XML database storage of documentation components of speech corpora on corpus, event, global speaker data and media annotation level allows the

12 We plan to use AJAX technology to provide instant support to the user during the XQuery definition process: XML Schema based suggestions will be helpful while filling in the XML path expressions of the XQuery.

publication of personalized views of browsable, well structured corpus information and precise access to media data. Such fine grained views may vary depending on individual user needs. Personalized information representations are generated directly by the database application. First, all necessary XML instances and related XSL transformations are read from the database. Then a specific view is dynamically generated by the Oracle XSLT processor applying the XSL transformations to the underlying XML data collections. Figure 10 shows an example of a user customized event documentation view. Here, too, media resources are dynamically linked with the documentation view during the XSL transformation process.

3 Outlook

For the near future we plan to extend the system to include the management of XML schema based annotations of audio and video signals (e.g. orthographic and phonetic transcriptions). The integration of meta- information and signal annotations into the XML database storage model will allow us to bundle all corpus related information for the retrieval processes and to perform complex queries against the complete XML data collections.

References

Brinckmann Caren, Stefan Kleiner, Ralf Knöbl & Nina Berend (2008): German Today: an areally extensive corpus of spoken Standard German. Proceedings 6th International Conference on Language Resources and Evaluation (LREC 2008), Marrakech, Morocco.

Dickgießer, Sylvia (2008): diasysco - Schemata für die Dokumentation von Korpora der gesprochenen Sprache (in prep.)

Gasch Joachim, Brinckmann Caren, Dickgießer Sylvia (2008): memasysco: XML schema based metadata management system for speech corpora. Proceedings 6th International Conference on Language Resources and Evaluation (LREC 2008), Marrakech, Morocco.

Schiel, F. and Draxler, C. (2004): *The production of speech corpora. Version 2.5*. URL: http://www.phonetik.uni-muenchen.de/forschung/BITS/TP1/Cookbook/ (accessed Mai 15, 2008).