ZENTRUM FÜR INFORMATIONSMODELLIERUNG AUSTRIAN CENTRE FOR DIGITAL HUMANITIES

KARL-FRANZENS-UNIVERSITÄT GRAZ
UNIVERSITY OF GRAZ

Beyond TEI

Digital Editions with XPath & XSLT for the Web & in LaTeX

Sarah Lang

Harvard, April/May 2022

Overview

1. The workshop
2. Markup & annotation
3. Markup is all around you!
4. Primer on data structures & metadata formats

The workshop

Goals

1. get to know XPath & XSLT (and learn how to use it)
2. understand the role of XML/TEI, XPath and XSLT in Digital Editing
3. be able to use XSLT to generate HTML and LaTeX output from TEI
4. Two days isn't enough for you to master XSLT!

Schedule

Day 1, morning	XML, TEI and Digital Editing → repetition of the basics, making sure we're all on the same page, understanding why we're even learning XSLT.
Day 1, afternoon	Navigating XML documents using XPath, introduction to HTML (& Bootstrap) and LaTeX (& `reledmac`)
Day 2	Transforming XML documents into HTML & LaTeX output formats using XSLT

Single point of entry for all workshop-related materials: LaTeX Ninja blogpost & Github Repository (‘additional resources’ directory)

Introductions

Please introduce yourselves!

1. Name, pronouns, field/topic of study
2. Why did you come to this workshop?
3. Previous experience with Digital Humanities (DH) or editing?

Contact

@SarahALang_
@latex_ninja

sarahalang.com
latex-ninja.com

sarah.lang@uni-graz.at

Sarah Lang (she/they)

- originally from Germany, now in Graz (Austria)
- Studied Latin, French & History (teacher's education) in Graz & Montpellier (France), then Archaeology Bachelor, Master in Religious Studies & Philosophy
- got a DH certificate & started working at Zentrum für Informationsmodellierung (ZIM) / Centre for Information Modelling in Graz
  - Moral Weeklies/Spectators → *gams.uni-graz.at/mws*
  - Graz Repository of Ancient Fables (GRaF) → *gams.uni-graz.at/graf*
  - *PhD thesis:* Decoding alchemical *Decknamen* digitally. A Polysemantic Annotation and Machine Reasoning Algorithm for the Corpus of Iatrochymist Michael Maier (1568–1622)
- *Now:* teaching in Graz, Passau & Vienna; PostDoc in Graz. *Research interests:* history of science (alchemy), Neo-Latin, text mining and computer vision

Markup & annotation

Annotating in the Humanities

→ Typical practice for the Humanities. Can be done on paper.
You can go about it in many ways:

- collect specific metadata to enrich your primary data with (research-driven modelling)
- provide metadata as general as possible to facilitate reuse (curation-driven modelling)
- we can provide administrative or technical metadata but also encode semantic information (implicit structures) for the machine.
- Often: describe logical structure of documents for computers ('This is a heading. This is a paragraph.')

Example:

```
<name type="person">J. W. v. Goethe</name>
```

In the digital realm

Adding additional data to source document in a machine-readable format.

Formal models

If you model is machine-processable, it's a formal model – i.e. markup creates a formal model of your data.

Many names for the same thing

markup, encoding, annotating, ...

Different levels of annotation

Base annotation Encode formal criteria of the text structure (i.e. headings, paragraphs), adding some metadata. Mostly *presentational*.

Data enrichment Going from formal aspects to semantics: Annotating personal names, places (*Named Entities*), linking those to norm data, thus creating *Linked (Open) Data* (LOD).

Examples for norm data

- GND (*Gemeinsame Normdatei*, Integrated Authority File)
- GeoNames
- ...

Markup languages i

Annotation is also called mark-up.

> **Markup** refers to data included in an electronic document which is distinct from the document's content in that it is typically not included in representations of the document for end users, for example on paper or a computer screen, or in an audio stream. **Markup is often used to control the display of the document or to enrich its content to facilitate automated processing.**
>
> Older markup languages [...] typically **focus on typography and presentation**, [...] most modern markup languages, for example XML, **identify document components** (for example headings, paragraphs, and tables), with the expectation that technology such as stylesheets will be used to apply formatting or other processing.
>
> Some markup languages, such as the widely used HTML, have **pre-defined presentation semantics**, meaning that their specification prescribes some aspects of how to present the structured data on particular media. (Wikipedia)

Markup languages ii

machine readable

SMGL HTML XML ...

presentational vs. descriptive / semantic:

- e.g. ‘font size 14pt’ vs. ‘heading’ (=type). (explicit vs. implicit)
- text formatting vs. meaning of the text
- procedural, representative, descriptive / conceptual (semantic)
- WYSIWYG text processing vs. WYSIWYM
- Advantages of using macros in WS Word: change the settings once for the whole document
- this is achieved when we separate content from its presentation
- there are different ‘views’ on markup documents
- browser ‘renders’ HTML: I can see it in two different ways – rendered or as the HTML code

Markup languages iii

- There are tools to switch documents between different types of markup! (not all formats work equally well): **Pandoc** or **OxGarage** (for TEI mostly).
- Binary document formats, such as *.doc*, *.pdf*, *.dvi* (TEX output format) ≠ markup: You can tell by the fact that you cannot look at their 'code view'.
- A `.docx` file is a `.zip` archive (try unzipping it!) which contains XML files (but it's complicated).
- **Goal of markup:** Make the implicit (you know it's a heading) explicit for the computer (doesn't know otherwise).

Markup is all around you!

SGML

Standard Generalized Markup Language

.SGML

Introduces the principles of

the separation of form and content

Is a metalanguage (like XML)

HTML & XML are both derived from the older SGML (thus the similarity) – but HTML has a fixed tag set whereas XML is by definition *extensible*.

RTF

Rich Text Format Microsoft 1987 ● exchange format between text editors (different operating systems, such as Mac and Windows, didn't create interchangeable output formats).

Unlike plain text, RTF contains markup for formatting text which can then be 'retranslated' into the native editors.

.RTF

```
{\rtf1
Hello!
\line
{\i This} is \b{\i a
\i0 formattted \b0text}.
\par
\b THE \b0END.
}
```

JSON

.JSON JavaScript Object Notation pronounced like 'Jason' ● compact data format ● human readable ● set up in key-value pairs ● nested (like XML)

JSON vs. XML differences?

XML = describes structure, JSON = non-declarative syntax convention ● JSON: defines instances of structured data ● very flexible ● 'lightweight': little overhead, easier to read for data in key-value format ● valid javascript: can be instantiated into a javascript object via the *eval()* function.

Conclusion

JSON has advantages if you just need simple key-value pairs (**'simplicity'**). XML has more and allows for more **complexity**.

XML = mark up language

JSON = data exchange format

```
{
  "publisher": "Xema",
  "number": "1234-5678-9012-3456",
  "owner":
  {
    "Name": "Mustermann",
    "Vorname": "Max",
    "male": true,
    "hobbies": ["surfing", "chess"],
    "age": 42,
    "kids": [],
    "spouse": null
  }
}
```

PostScript

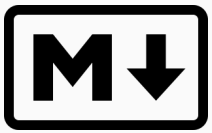
Page description language ● 1980s (Adobe Systems) ● vector graphic format for printers ● but also: Turing-complete, stack-oriented programming language ● used to be the standard in the printing industry ● today PDF (*Portable Document Format*) (also by Adobe, developed from PS) has become the standard ● could be generated via postscript printer drivers from all sorts of documents ● processed via 'Ghostscript' in UNIX ● decribes documents as scalable vector graphics which allows for loss-less zooming / scaling .ps // presentational

Example

This example program writes 'Hello World!' to position 50,50. By default, the PS coorindate system starts from the bottom left corner.

```
%!
/Courier findfont     % font type
20 scalefont          % font size 20
setfont               % set it
50 50 moveto          % (50, 50)
                      % = writing pos
(Hello World!) show   % print text
showpage              % show page
```

Markdown



→ Markdown in 60s

simple text formatting

.md // presentational

`*Italic*` ..... *Italic*
`**Bold**` ..... **Bold**
`## Heading 1` ..... Heading 1
`#### Heading 2` ..... etc.
`[Link]{http://a.com}` 'hidden' link
`![Image][http://url/a.png` image
`> Blockquote` ..... quote
`- List` ... list (can be nested, 2 spaces)
`* List` ..... alternative list
`1. enumerate` ..... numbered list
`---` ..... separator line
`` `Inline code` `` ..... Code ('backticks')
```` ```code block``` ```` ..... code block
````

Practice!

Try Markdown in 60s or 10min exercise

Primer on data structures & metadata formats

TEI, now what?

Why are we doing this workshop? The motivation from our abstract:

- [...] the Text Encoding Initiative (TEI) for XML has become the gold standard for scholarly editions of texts.
- But what happens after an edition is encoded in TEI?

Goals for the next session

1. There are many different types of data. They get stored in different structures.
2. Why XML and not something else?
3. What else is there?
4. **Also:** some metadata literacy!

Digital data representation

→ i.e. machine processable

Digital representations

- Images
  - raster graphics (*.png*, *.jpeg*)
  - vector graphics (*.svg*)
- Text
  - plain text (*.txt*)
  - formatted text (*.docx*, *.rtf*, *.xml*, *.tex*)
- Lists & tables (*.csv*, *.xlsx*)
- Sound (*.wav*, *.midi*)
- **Objects:** (Simulated) 3D view, abstracted representation by description and images

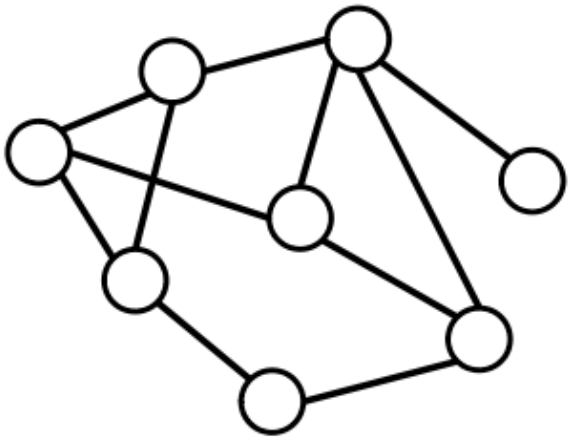
Further types

- markup languages (*.xml*, *.html*, etc.)
- data objects (*.json*, etc.)
- graphs / graph databases (*.rdf*, etc.)
- relational databases → SQL

Data structures: Graphs

Applications

- **Resource Description Framework (RDF):**
  - e.g. Blazegraph (Graph-DB)
  - Query: SPARQL
- Labeled Property Graphs
  - e.g. Neo4j (Graph-DB)
  - Query: Cypher



RDF/Turtle-Notation (`.ttl`)

```
@prefix ex: <http://example.com/#> .

ex:Graz a ex:city;
ex:name "Graz" ;
ex:inhabitants 288806 ;
ex:location [ ex:lat 47.4; ex:long 5.26 ] .

ex:Wien a ex:city;
ex:name "Wien" ;
ex:inhabitants 1897491 ;
ex:location [ ex:lat 47.12; ex:long 16.22 ] .
```

SPARQL query

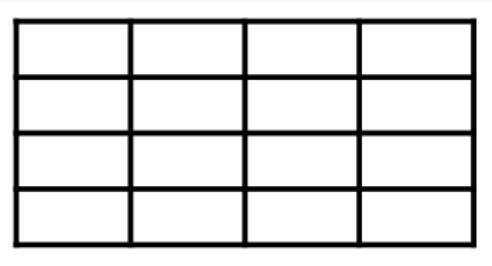
```
@prefix ex: <http://example.com/#> .

SELECT ?name, ?population
WHERE {
    ?city ex:inhabitants ?population .
}
```

Data structures: tabular data / relational databases

Applications

- e.g. SQLite, MySQL (relational dbs)
- Query: SQL



SQL query

```sql
CREATE TABLE "places" (
 "name" TEXT,
 "population" INTEGER,
 "longitude" REAL,
 "latitude" REAL,
 PRIMARY KEY("name")
);

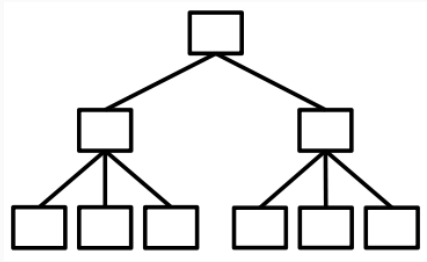
INSERT INTO places
VALUES
('Graz', 288806, 47.066667, 15.433333),
('Wien', 1897491, 48.208174, 16.373819);
```

(name, population, longitude, latitude)

Data structures: tree hierarchy 1 / JSON

Applications: **JavaScript Object Notation (JSON)**

- e.g. MongoDB
- no standardized query language, just JavaScript (*.js*)



JSON & JS (w3s)

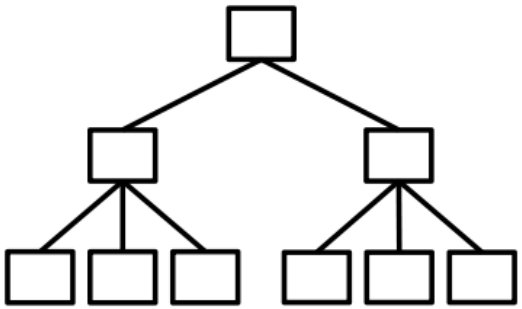
```
'{"name": "John", "age": 30, "car": null,
  "tree" : [
    "key": "value"
  ]
}'
const obj = JSON.parse('{"name":"John", "age":30, "city":"New York"}');
obj.age = obj.age.toString();
```

Data structures: tree hierarchy 2 / XML

Applications:

eXtensible Markup Language (XML):

- DBs: eXist, BaseX
- Query: XPath (w3s) and XQuery (w3s)



```
<!-- books.xml -->
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book>
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book>
    <title lang="en">Learning XML</title>
    <price>39.95</price>
  </book>
</bookstore>

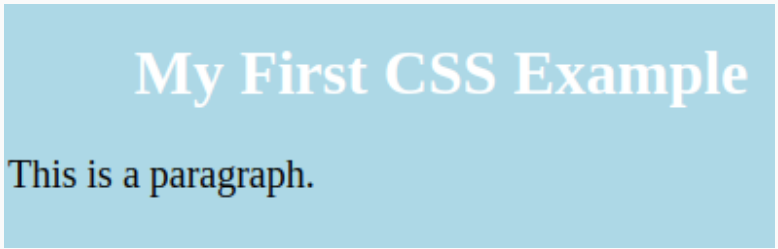
<!-- XQuery -->
for $x in doc("books.xml")/bookstore/book
where $x/price>30
order by $x/title
return $x/title

<!-- XPath -->
//title[@lang='en']
/bookstore/book[price>35.00]
```

Data structures: tree hierarchy 3 / web pages (HTML)

HTML (w3s) – structure

```
<!DOCTYPE html>
<html>
 <head>
   <title>Page Title</title>
 </head>
 <body>
  <h1>This is a Heading</h1>
  <p>This is a paragraph.</p>
 </body>
</html>
```



CSS in HTML (w3s) – rendering

```
<!DOCTYPE html>
<html>
  <head>
    <style>
body {
  background-color: lightblue;
}
h1 {
  color: white;
  text-align: center;
}
p {
  font-family: verdana;
  font-size: 20px;
}
    </style>
  </head>
  <body>
    <h1>My First CSS Example</h1>
    <p>This is a paragraph.</p>
  </body>
</html>
```

Why so many data formats?

Different data formats (& standards) focus on different aspects & have different goals:

1. **text-based**
   - Text Encoding Initiative (**TEI**)
   - Extensible Hypertext Markup Language (**XHTML** = XML-compliant HTML)
   - Open Document Format for Office Applications (**ODF**)
2. **page-based**
   - TEX / LATEX
   - **XSL-FO** (XSL Formatting Objects, discontinued)
3. **ontology-based**
   - Resource Description Framework (**RDF**) & RDF Schema (
   - Machine-Readable Cataloging (MARC))
   - Web Ontologie Language (**OWL**)
   - Simple Knowledge Organisation System (**SKOS**)
   - Conceptual Reference Model (**CIDOC-CRM**)
4. **digital archiving / digital objects**
   - Dublin Core Metadata Initiative (**DCMI**), known as Dublin Core (**DC**)
   - Metadata Encoding and Transmission Standard (**METS**)
   - Metadata Object Description Schema (**MODS**)
   - Encoded Archival Description (**EAD**)
   - Charters Encoding Initiative (**CEI**)

A primer on metadata

What are metadata?

- „data about data“
  1. data about containers of data = **structural metadata**
  2. data about the content represented by data = **descriptive metadata**
- functions:
  1. descriptive
  2. administrative
  3. technical
  4. use

There are standards for the description of metadata (and many are XML-based), e.g.

- Machine-Readable Cataloging (**MARC**)
- Metadata Object Description Schema (**MODS**)
- Encoded Archival Description (**EAD**)
- Lightweight Information Describing Objects (**LIDO**)
- Collective Description of Works of Art (**CDWA**) / Visual Research Association (**VRA**)
- Europeana Metadata Model (**EDM**)
- Resource Description Framework (**RDF**)
- Metadata Encoding & Transmission Standard (**METS**)
- Dublin Core (**DC**)
- Functional Requirements for Bibliographic Records (**FRBR**)
- the *<teiHeader>* has metadata…

Slides on metadata in DH with more information

Dublin Core (DC) i

What is the DC?

- founded in Dublin (Ohio) in 1995
- **two levels:** simple (15 elements) & qualified (additional *`Audience`*, *`Provenance`* and *`RightsHolder`*)
- **classes of terms:** elements (nouns) & qualifiers (adjectives).
- can be expressed in RDF/XML
- each element is optional & can be repeated
- also: dc:terms

> The Dublin Core™ metadata standard is a simple yet effective **element set for describing a wide range of networked resources.** [...] Another way to look at Dublin Core™ is as a "small language for making a particular class of statements about resources". In this language, there are two classes of terms – *elements* (nouns) and *qualifiers* (adjectives) – which can be arranged into a simple pattern of statements.
> (source)

Dublin Core (DC) ii

The Core

i.e. the elements:

1. title
2. subject
3. description
4. type
5. source
6. relation
7. coverage
8. creator
9. publisher
10. contributor
11. rights
12. date
13. format
14. identifier
15. language

Qualified

1. (audience)
2. (provenance)
3. (rights holder)

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">

   <rdf:Description rdf:about="http://media.example.com
                               /audio/guide.ra">
      <dc:creator>Rose Bush</dc:creator>
      <dc:title>A Guide to Growing Roses</dc:title>
      <dc:description>Describes process for
        planting and nurturing different kinds
        of rose bushes.</dc:description>
      <dc:date>2001-01-20</dc:date>
   </rdf:Description>
</rdf:RDF>
```

Note the two namespaces *rdf:* and *dc:*.

Metadata Encoding and Transmission Standard (METS)

METS

- tool for encoding digital library objects
- container format for documents in which contents of different formats can be integrated
- also describes relationships between objects
- describes logical and physical structure of an object
- also contains descriptive (bibliographical) and administrative metadata
- relatively simple and straightforward
- supports a wide range of materials
- <website> & more info here

Only structural map is required.

```
<mets>
  <metsHdr/>
  <dmdSec/>
  <amdSec/>
  <fileSec/>
  <structMap/>
  <structLink/>
  <behaviorSec/>
</mets>
```

Metadata Object Description Schema (MODS)

MODS

- can represent the major elements from a MARC record

→ represents key bibliographic data in easily understandable names

→ easier to understand than MARC (for the uninitiated)

- bridges the gap between library application and bibliographic source that don't make use of cataloging metadata formats
- richer than the Dublin Core (DC) but less detailed than MARC
- partially backwards compatible with MARC

```
<mods:mods xmlns:mods="http://www.loc.gov/mods/v3">
  <mods:titleInfo>
    <mods:nonSort>The </mods:nonSort>
    <mods:title>
      1946 Library of Congress recital
    </mods:title>
  </mods:titleInfo>
  <mods:relatedItem type="constituent"
                    ID="DMD_disc01_tr001">
    <mods:titleInfo type="uniform">
      <mods:partName>Chaconne von Vitali
        </mods:partName>
    </mods:titleInfo>
  </mods:relatedItem>
  <mods:identifier type="lccn">99594334
    </mods:identifier>
</mods:mods>
```

Code example from here.

Encoded Archival Description (EAD)

EAD

is a standard for encoding descriptive information regarding archival records

example EAD & Wikipedia (source of the example)

```
<eadheader>
   <eadid countrycode="us" identifier="bachrach_lf">
     bachrach_lf</eadid>
   <filedesc>
      <titlestmt>
         <titleproper encodinganalog="Title">
          Louis Fabian Bachrach Papers</titleproper>
         <subtitle>An inventory of his papers at
           Blank University</subtitle>
         <author encodinganalog="Creator">Mary Smith</author>
      </titlestmt>
      <publicationstmt>
         <publisher encodinganalog="Publisher">
           Blank University</publisher>
         <date encodinganalog="Date" normal="1981">
           1981</date>
      </publicationstmt>
   </filedesc>
   <profiledesc>
      <creation>John Jones
         <date normal="2006-09-13">13 Sep 2006</date>
      </creation>
   </profiledesc>
</eadheader>
```

Charters Encoding Initiative (CEI)

CEI

considers the possibilities of a standard to encode medieval and early modern charters with XML.

- implements the *Vocabulaire Internationale de Diplomatique*
- founded in 2004
- MOM-CA, the collaborative charter archive of Monasterium.net works with CEI
- <website> & an example. Also: the example below.

```
<text type="charter">
  <idno>600202b</idno>
  <chDesc id="a">
    <head>Prag, 1360 Febr. 2.</head>
      <issued>
         <placeName>Prag</placeName>
         <date>1360-02-02</date>
      </issued>
      <abstract>
         <p> Karl verspricht Ludwig ...
         </p>
      </abstract>
      <witList>
         <witness sigil="B">
            Brandenburgisches LHA Potsdam
            “Rep. 37 Hohennauen Nr. 683,
            fol. 225” (18. Jh.)
         </witness>
      </witList>
      <diplomaticAnalysis>
         <bibl type="D">Fidicin, 42...</bibl>
      </diplomaticAnalysis>
   </chDesc> <tenor> </tenor>
</text>
```

Resource Description Framework (RDF)

RDF

- framework for describing resources in the World Wide Web
- can contain metadata
- language of the ‘Semantic Web’ (web 3.0) – makes things machine-processable
- RDF Schema (RDFS) offers Classes and Properties

RDF/ turtle notation (`.ttl`) example from before

```
@prefix ex: <http://example.com/#> .

ex:Graz a ex:city;
ex:name "Graz" ;
ex:inhabitants 288806 ;
ex:location [ ex:lat 47.4; ex:long 5.26 ] .
```

Simple Knowledge Organization System (SKOS)

SKOS

RDF vocabulary for representing semi-formal *knowledge organization systems* (KOSs), such as thesauri, taxonomies, classification schemes and subject heading lists.
→ less rigorous than the logical formalism of ontology languages such as OWL

(SKOS primer)

```
@prefix skos:
  <http://www.w3.org/2004/02/skos/core#> .
@prefix rdf:
  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:
  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix dct: <http://purl.org/dc/terms/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix ex: <http://www.example.com/> .
@prefix ex1: <http://www.example.com/1/> .
@prefix ex2: <http://www.example.com/2/> .

ex:animals rdf:type skos:Concept;
  skos:prefLabel "animals"@en;
  skos:narrower ex:mammals.

ex:mammals rdf:type skos:Concept;
  skos:prefLabel "mammals"@en;
  skos:broader ex:animals.
```

Europeana Data Model (EDM) i

EDM

Model to integrate data sources from different providers and thus improve interoperability:

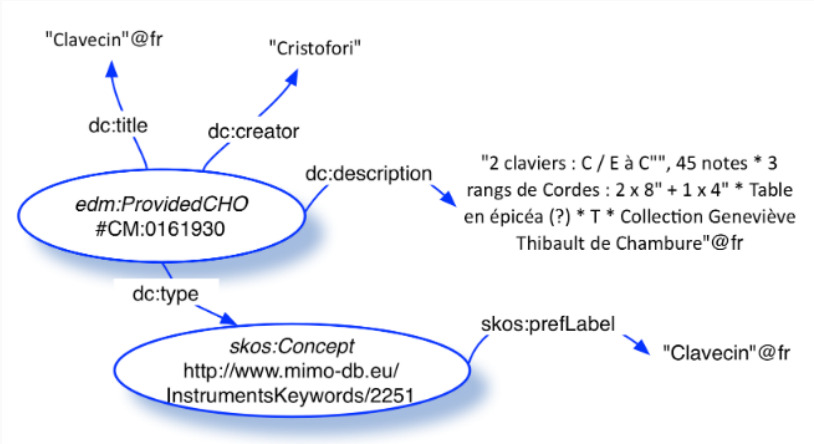
> “EDM transcends domain-specific metadata standards, yet accommodates the range and richness of community standards such as LIDO for museums, EAD for archives or METS for digital libraries.” (EDM Factsheet)

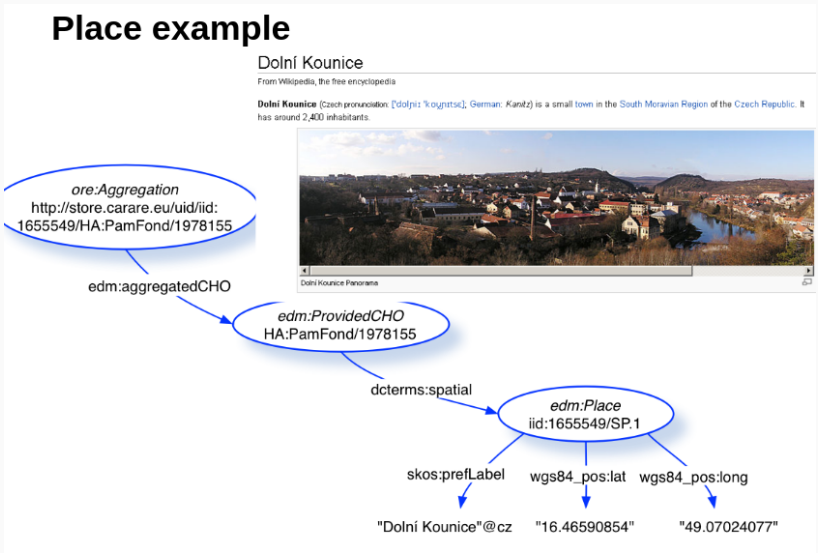
...integrates the following standards:

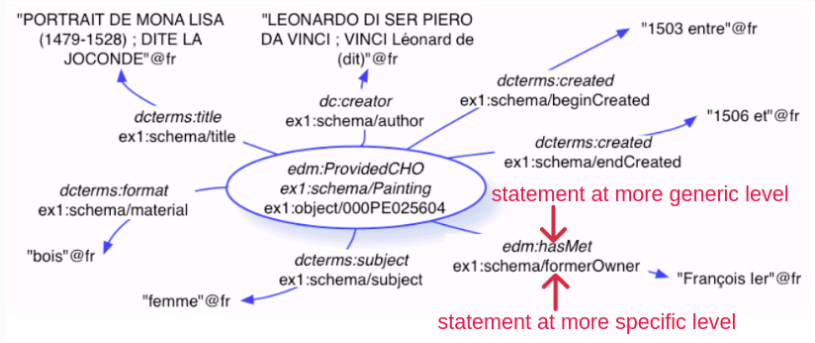
1. **OAI ORE** (Open Archives Initiative Object Reuse & Exchange) for organizing an object’s metadata and digital representation(s)
2. **Dublin Core** for descriptive metadata
3. **SKOS** (Simple Knowledge Organization System) for conceptual vocabulary representation
4. **CIDOC-CRM** for event and relationships between objects

This is achieved in RDF (can be written as XML) → RDF uses the Semantic Web principles to integrate those data sources → Metadata standards aren’t exclusive, they can be combined!

Europeana Data Model (EDM) ii







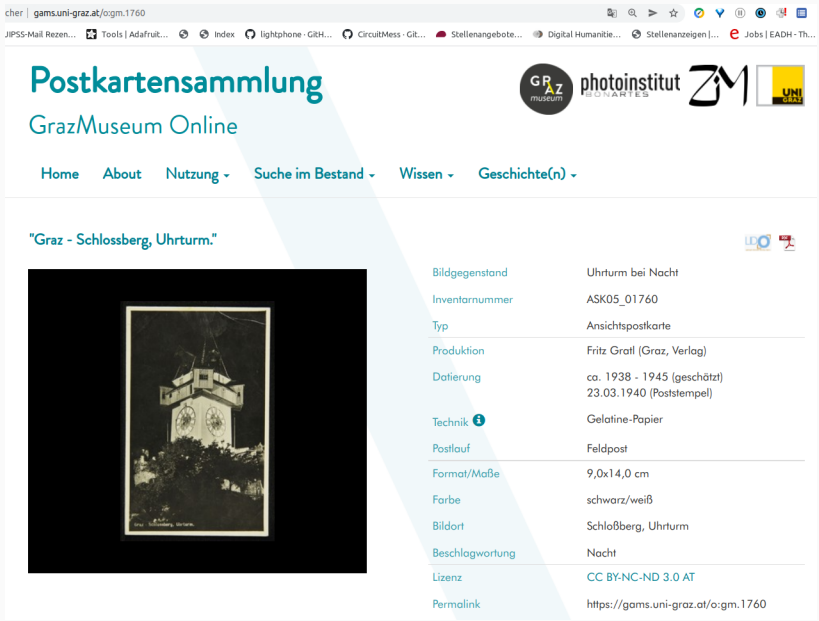
(More info.)

Lightweight Information Describing Objects (LIDO)

LIDO

> Lightweight Information Describing Objects (LIDO) is an **XML schema for describing museum or collection objects.** Memory institutions use LIDO for "exposing, sharing and connecting data on the web". It can be applied to all kind of disciplines in cultural heritage, e.g. art, natural history, technology, etc.
> LIDO is a specific application of CIDOC CRM. (Wikipedia)

A postcard (text-bearing object)




```
<lido:titleWrap>
    <lido:titleSet>
        <lido:appellationValue>
            Chickens and Ducks
        </lido:appellationValue>
    </lido:titleSet>
</lido:titleWrap>
```

LIDO example 1

Notice the two namespaces (*lido:* and *t:*)!

```
<lido:lido
  xmlns:lido="http://www.lido-schema.org"
  xmlns:t="http://www.tei-c.org/ns/1.0">
  <lido:lidoRecID lido:type="PID">o:gm.1760</lido:lidoRecID>
  <lido:category>
    <lido:conceptID lido:source="CIDOC"
                    lido:type="ID">E22</lido:conceptID>
    <lido:term xml:lang="eng">Man-Made Object</lido:term>
    <lido:term lido:label="info:fedora/context:gm">
      Postkartensammlung Online</lido:term>
  </lido:category>
  <lido:descriptiveMetadata xml:lang="deu">
    <lido:objectClassificationWrap>
      <lido:objectWorkTypeWrap>
        <lido:objectWorkType>
          <lido:conceptID lido:source="http://vocab.getty.edu/aat"
            lido:type="ID">300026819</lido:conceptID>
          <lido:term lido:label="info:fedora/context:gm-ansicht">
            Ansichtspostkarte</lido:term>
        </lido:objectWorkType>
      </lido:objectWorkTypeWrap>
    </lido:objectClassificationWrap>
    <!-- to be continued... LIDO is very verbose --->
```

LIDO example 2

```
<!-- continued... --->

    <lido:objectIdentificationWrap>
      <lido:titleWrap>
        <lido:titleSet>
          <lido:appellationValue>Graz - Schlossberg, Uhrturm.
             </lido:appellationValue>
        </lido:titleSet>
      </lido:titleWrap>
      <lido:repositoryWrap>
        <lido:repositorySet lido:type="orgname">
          <lido:repositoryName>
            <lido:legalBodyID lido:source="http://d-nb.info/gnd"
              lido:type="ID">2022740-1</lido:legalBodyID>
            <lido:legalBodyName>
              <lido:appellationValue>GrazMuseum</lido:appellationValue>
            </lido:legalBodyName>
          </lido:repositoryName>

    <!-- to be continued... LIDO is very verbose --->
```