Introduction to TEI schemata

Antidote workshop, Klosterneuburg & Lyon

Matthias Gille Levenson

Université Versailles Saint Quentin (UVSQ) & École Normale Supérieure de Lyon, France

matthias [dot] gille [-] levenson [at] ens-lyon [dot] fr

Where to find these slides

- https://github.com/matgille/Antidote_workshop_2025_ODD.git

Conformance and validity

- XML stands for **eXtensible Markup Language**.
- It is a format that allows to describe any kind of textual (or numeric) data
- It is the actual *format* the TEI uses, but it might change/evolve in the future.

Two important concepts: Conformance and validity

- A document **must** be XML conformant, that is, respect the rules of the XML format
- A document **may** be validated against a given **schema**, that is a document that checks given rules are respected
- Some examples of specifications: TEI, EAD, DublinCore, AltoXML, PageXML, RDF, etc...
- Along with validity comes another important concept/nightmare: *namespaces*. You'll understand soon enough.

Conformance

- XML is composed of elements, attributes, attribute values and text.

```
<?xml version="1.0" encoding="UTF-8"?>
<TEI xml:id="drp" xmlns="http://www.tei-c.org/ns/1.0">
   <teiHeader>
      <fileDesc>
         <titleStmt>
            <author cert="low">Matthias Gille Levenson</author>
         </titleStmt>
         <publicationStmt>
            <publisher>Unpublished</publisher>
            <availability>
               <licence>Creative Commons CC BY-NC-SA 4.0 FR</licence>
            </availability>
         </publicationStmt>
      </fileDesc>
   </teiHeader>
</TEI>
```

Figure: This is *correct* XML (but incorrect/incomplete TEI)

Recap: vocabulary

Four main objects (called *nodes*):

- **Element** nodes: the basic object of any XML document:
  `<elementName/>` or `<elementName>...</elementName>`
- **Attribute** nodes: part of an element that helps specifying it:
  `attributeName="value"`
- **Text nodes**: every textual object inside an element
- **Comment** nodes: content not parsed by the machine:
  `<!-- some useful comment -->`

The root node/element

- One and only one top element that contains everything else

```
<?xml version="1.0" encoding="UTF-8"?>
<TEI xml:id="drp" xmlns="http://www.tei-c.org/ns/1.0">
   <teiHeader>
      <fileDesc>
         <titleStmt>
            <author cert="low">Matthias Gille Levenson</author>
         </titleStmt>
         <publicationStmt>
            <publisher>Unpublished</publisher>
            <availability>
               <licence>Creative Commons CC BY-NC-SA 4.0 FR</licence>
            </availability>
         </publicationStmt>
      </fileDesc>
   </teiHeader>
</TEI>
```

Figure: This is *correct* XML (but incorrect/incomplete TEI)

Conformance

- *No* overlapping elements

```
<?xml version="1.0" encoding="UTF-8"?>
<TEXT xml:id="drp" xmlns="http://www.tei-c.org/ns/1.0">
   <NODE_A>Hello <NODE_B>World</NODE_A></NODE_B>
</TEXT>
```

Figure: This is *incorrect* XML

Conformance

- Some elements can contain other elements, but elements can also be empty

```
<?xml version="1.0" encoding="UTF-8"?>
<div type="chapitre" n="1">
   <lb/>ca . i. quomodo diuiduntur potentiae
   <lb/>anime et inquibus potenciis
   <lb/>habent esse
   <lb/>uirtutes
</div>
```

Figure: This is *correct* XML

Conformance

- Attribute values must be inside quotes

```
<?xml version="1.0" encoding="UTF-8"?>
<TEXT>
   <NODE_A type=sentence>Hello
      <NODE_B>World</NODE_B>
   </NODE_A>
</TEXT>
```

Figure: This is *incorrect* XML

Conformant or not ?

```
<sentence>Longtemps, je me suis couché de bonne heure.</sentence>

<sentence>Longtemps, je me suis couché de bonne heure.<sentence>

<sentence type=incipit>Longtemps, je me suis couché<linebreak></linebreak> de bonne heure.</sentence>

<sentence type=incipit>Longtemps, je me suis couché<linebreak/> de bonne heure.</sentence>

<sentence>Longtemps, je me suis couché de bonne heure.</sent>

<sentence type="incipit">Longtemps, je me suis couché de bonne heure.</sentence>

<paragraph>
        <sentence type="incipit">Longtemps, je me suis couché <strikethrough>de bonne heure.</sentence>
        <sentence>Parfois, à peine ma bougie éteinte</strikethrough>,
            mes yeux se fermaient si vite que je n'avais pas le temps de me dire</sentence>
</paragraph>

<paragraph>
        <sentence type="incipit">Longtemps, je me suis couché <strikethrough>de bonne heure.</strikethrough></sentence>
        <sentence><strikethrough>Parfois, à peine ma bougie éteinte</strikethrough>,
            mes yeux se fermaient si vite que je n'avais pas le temps de me dire</sentence>
</paragraph>

```

What is the TEI?

- A community
- A "standard" and a set of rules, the Guidelines: `https://tei-c.org/release/doc/tei-p5-doc/en/html/`
- A way of seeing/modelling "the" text

- Separate the appearance and the "essence" of textual objects

del mal del otro o del buen zelo[3]. Si del mal, cuidando que non devía el omne sofrir tal mal, esta es misericordia, ca misericordia non es otra cosa sinon tristeza del mal que omne sufre sin meresçimiento, segund que dize el philósofo en el IIº de la *Retórica*; mas si toma tristeza del bien, esto es en dos maneras, ca o le pesa del bien que otri ha, maguer que lo él meresca de aver, e esta es enbidia, ca enbidia non es otra cosa sinon // [Fol. 176r] dolor o tristeza del bien que otro ha[4]; [mas si le pesa del bien que otro ha] porque lo non meresçe aver, (e) así es némesis o

[3] del buen zelo] del bien *Glosa*.

[4] En este punto ha producido un salto de texto de igual a igual respecto a la *Glosa*, donde se lee seguidamente el texto que transcribo entre corchetes, que es necesario para que el sentido del pasaje no quede truncado.

[5] [a los mayores]] *om. Castigos* respecto a la *Glosa*.

Figure: Fragment of an edition of the *Castigos de Sancho IV* (Marín Sánchez 2003)

- Separate the appearance and the “essence” of textual objects

```
<p>del mal del otro o
  <app>
    <lem>del buen zelo</lem>
    <rdg wit="#Glosa">del bien</rdg>
  </app>
   . Si del mal, cuidando que non devía el omne sofrir tal mal, esta es misericordia, ca
  misericordia non es otra cosa sinon tristeza del mal que omne sufre sin meresçimiento,
  segund que dize el philósofo en el IIo de la Retórica; mas si toma tristeza del bien,
  esto es en dos maneras, ca o le pesa del bien que otri ha, maguer que lo él meresca de
  aver, e esta es enbidia, ca enbidia non es otra cosa sinon<pb n="176r"/> dolor o
  tristeza del bien que otro ha;
  <choice>
    <sic>mas si le pesa del bien que otro ha</sic>
    <corr/>
  </choice>
  <note>En este punto ha producido un salto de texto de igual a igual respecto a la Glosa,
    donde se lee seguidamente el texto que transcribo, que es necesario
    para que el sentido del pasaje no quede truncado.</note> porque lo non meresçe aver,
    <supplied>e</supplied> así es némesis o desdén</p>
```

Figure: Its possible representation in XML-TEI

The TEI, what for ?

- Describing a text using the experience of a **large community**
- Producing semantic data that can be **read** by the human and **processed** by the computer
- Make sharing and reusability of these representations easier

What is the use of a schema?

- A schema checks if the data is correctly formatted.
- A schema makes assumptions about what is legal and what is illegal in an XML data structure
- Applied to some text encoding, a schema is a model of the “text”, that is, a **formal representation** of the text.
- From a pragmatic point of view, it helps harmonizing your encoding and is an **important step towards interoperability**

Integrating the schema in your encoding sessions

Some XML editors use the schema as a tool for easing the encoding:

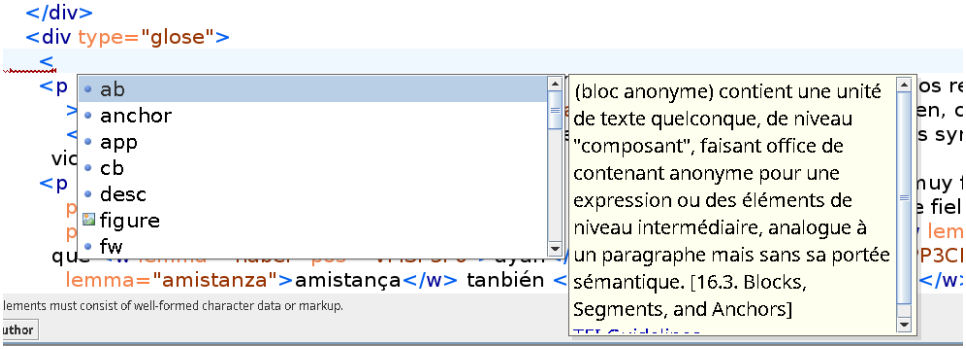


Figure: Oxygen XML Editor allows you to choose between the allowed elements

Schemas

- A schema is a document that is used to control the quality of some encoding.
- The TEI provides **rules** (the *Guidelines*, human readable), and their **formalization** (*schemas*, machine readable) to check the validity of a given document
- From an abstract point of view, the schema represents the formalization of *your* modelling of a given text or genre.

Schemas

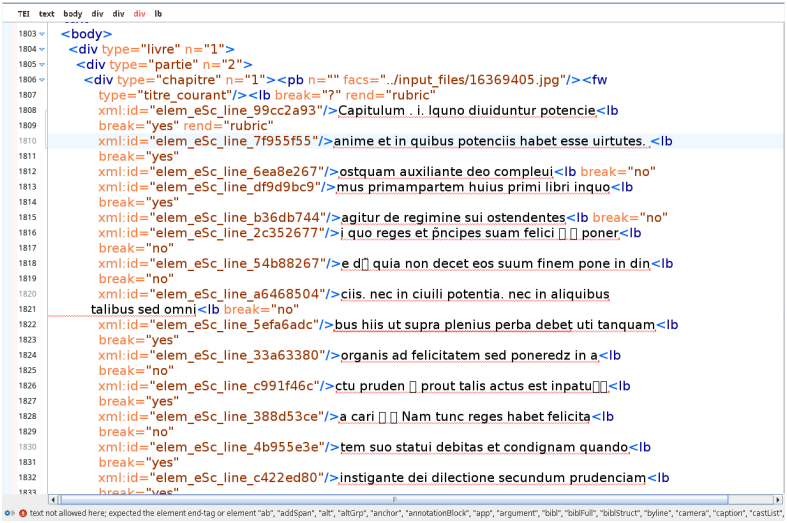


Figure: This fragment is *well-formed*, but it is not *valid* according to the XML-TEI schema. A paragraph `p` or an anonymous block `ab` should wrap the lines.

Schema formats

There are multiple schema formats. The most important are:

- Document Type Definition (DTD), already mentionned before
- XML schema (XSD format)
- Relax NG (RNG) schemas and its compact version (RNC)

The former is less used nowadays but can prove usefull for creating shortcut elements thanks to **XML entities** (e.g. for abbreviation encoding)

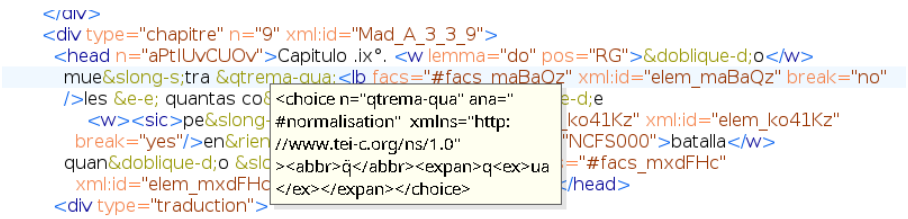
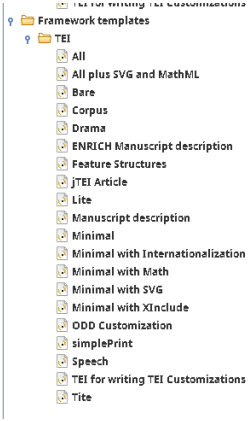


Figure: A possible way to encode abbreviations using XML entities / DTD definitions

TEI subsets



- There are multiple TEI schemas available.
- As a scientific standard, the TEI needs to be adapted to the users needs.

Why should you create your own TEI customization ?

- Consider the TEI as a tool to model your textual object
- You probably won't need all the intellectual tools provided by the TEI Consortium.
- As medievalists, you might want to discard the elements belonging to the "Transcriptions of Speech" section of the TEI (alas!)
- Your schema will be the formalization of your way of seeing the text, and you need it to be as specific as possible.

Two main methods

- The inclusion method: start from scratch, add the elements you need
- The exclusion method: start from a predefined schema, remove the elements you won't need

Two main methods

- The inclusion method: start from scratch, add the elements you need
- The exclusion method: start from a predefined schema, remove the elements you won't need
- I would recommended to adopt the first approach...

Two main methods

- The inclusion method: start from scratch, add the elements you need
- The exclusion method: start from a predefined schema, remove the elements you won't need
- I would recommended to adopt the first approach...
- or at least starting from a preset specification that meets some of your interests (we'll see this later).

Two main methods

- The inclusion method: start from scratch, add the elements you need
- The exclusion method: start from a predefined schema, remove the elements you won't need
- I would recommended to adopt the first approach...
- or at least starting from a preset specification that meets some of your interests (we'll see this later).
- Mastering the Guidelines is essential to know what is possible for your project.

Don't reinvent the wheel !

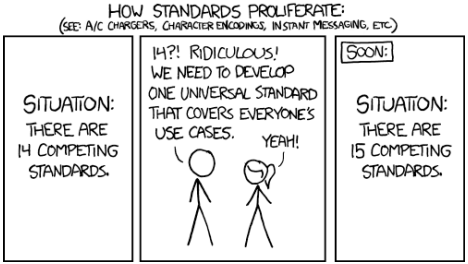


Figure: `https://xkcd.com/927/`

- You have to specify, and adapt the TEI rules to your own project
- However, the modelling problems you will face are probably *not* new
- Please don't reinvent the wheel !
- Search for what's been done for similar problems and try the provided solutions !
- And (for a start), avoid inventing/creating new elements.

Is that all, folks ?

- Create a specific schema is a good start.

Is that all, folks ?

- Create a specific schema is a good start.
- What's missing then ?

Is that all, folks ?

- Create a specific schema is a good start.
- What's missing then ? *Documentation*.

Is that all, folks ?

- Create a specific schema is a good start.
- What's missing then ? *Documentation*.
- You want now to explain your rules to your supervisor, to your colleagues and to your children. You need to write this *in natural language*.

Is that all, folks ?

- Create a specific schema is a good start.
- What's missing then ? *Documentation*.
- You want now to explain your rules to your supervisor, to your colleagues and to your children. You need to write this *in natural language*.
- What could be better than a TEI document to describe another TEI document?

Here comes the ODD

- ODD stands for "One Document Does it All"
- An ODD is a TEI document that is used to create both the documentation and the schema in a single document.
- Important reading: Lou Burnard. "What Is TEI Conformance, and Why Should You Care?" In: *Journal of the Text Encoding Initiative* 12 (2019). ISSN: 2162-5603

One Document to Rule Them All

An ODD contains:

- the machine-readable elements used to *validate* an edition
- the human-readable elements used to *document* it

Text Body

- The TEI is organized by *thematic* modules

TEI as modules

- 22 modules
- Each module is binded to a list of elements. Deactivate a module and those elements won't be accessible anymore.
- 4 main modules: `tei`, `core`, `header`, and `textstructure`, almost mandatory
- The other module you might want to take a look at are: `msdescription`, (`transcr`) and `textcrit`

An Odd is NOT a schema

- It is a **TEI document** used to produce schemata.
- It needs to be **converted** to a schema (and a documentation when applicable)
- The resulting schema (in general, in RNG) is the document you will use to validate your XML documents.
- Don't try to validate your files against ODD's !

Producing ODD's and schemas with ROMA

- ROMA, maintained by Raffaele Viglianti (lead), Peter Stadler and Bryan Wang.
- ROMA is a simple interface for designing ODDs and creating schemas
- URL: `https://roma.tei-c.org/`

A quick look at our ODD

With ROMA, you can easily design your own ODD:

- Include or exclude specific modules
- Completely remove elements (they won't be allowed anymore)
- Constrain attribute types, or attribute values
- Make some attributes mandatory
- etc.

Producing ODD's and schemas with ROMA

- I'm interested in the encoding of a medieval manuscript. How can I create the most adapted ODD with ROMA?
- I want to make sur the user uses an attribute `@n` for each page beginning element (`pb`). How can I do ?
- I want to remove the `div1`, `div2`, ..., `div7` elements, so that divisions are marked up with typed and identified `div` elements only.

Producing ODD's and schemas with ROMA

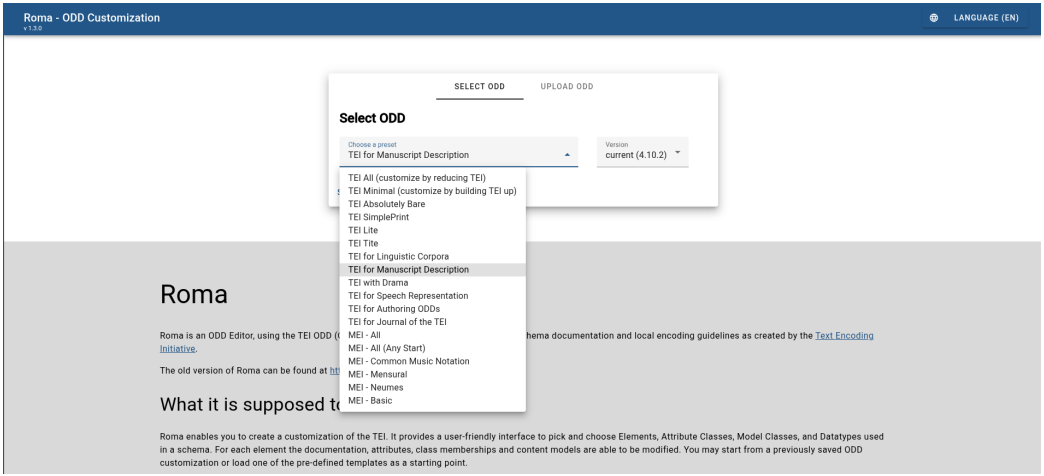


Figure: ROMA main page: presets

Producing ODD's and schemas with ROMA

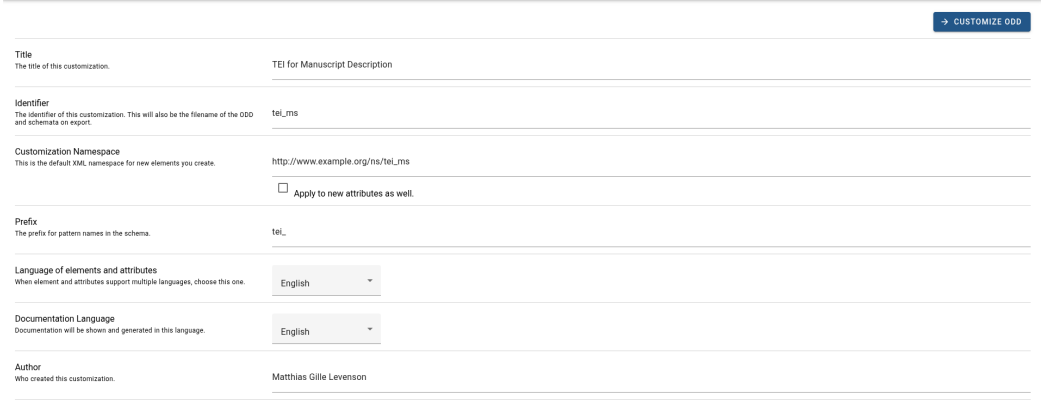


Figure: Configuring your ODD

Producing ODD's and schemas with ROMA

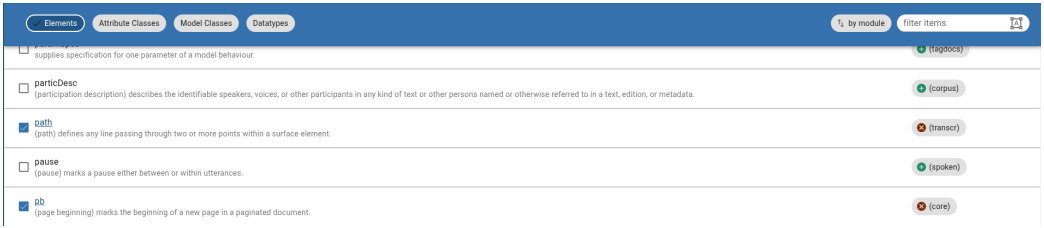


Figure: Find the element you want to modify (here, pb)

Producing ODD's and schemas with ROMA

Figure: Choose the feature you want to adapt (documentation or attribute)

Producing ODD's and schemas with ROMA

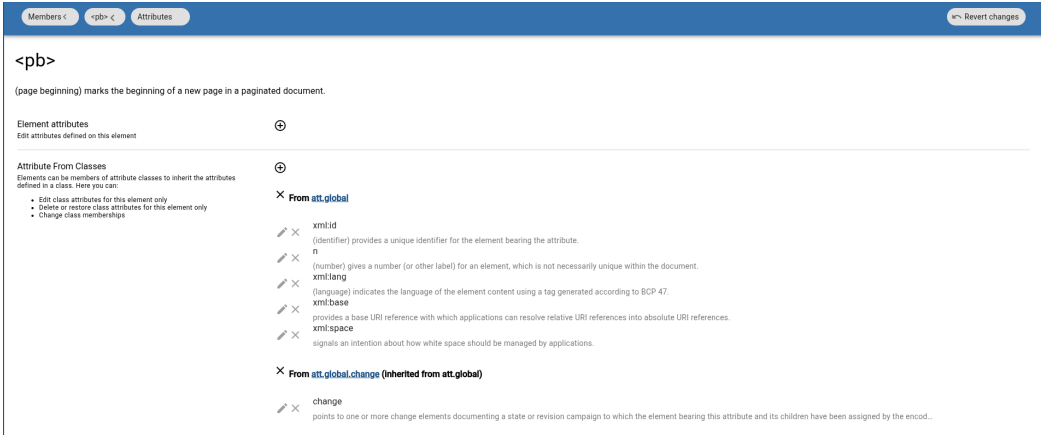


Figure: Find the attribute you want to modify

Producing ODD's and schemas with ROMA

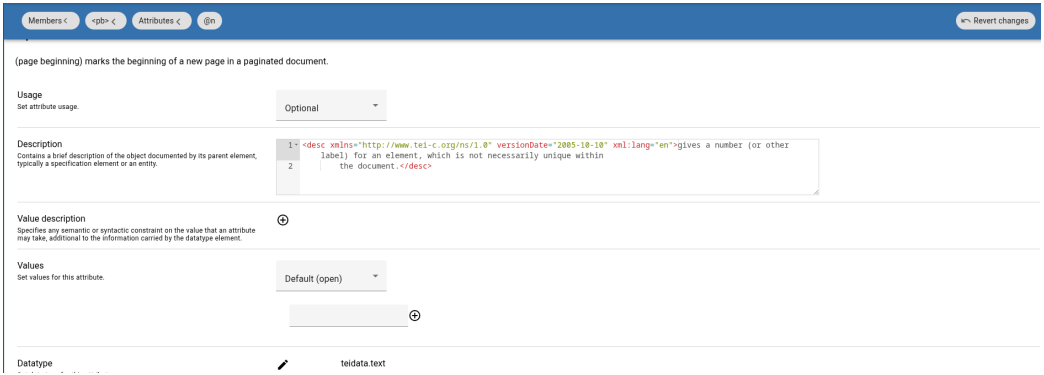


Figure: Change its usage ! It's (almost) automagical !

The produced ODD

```
      </encodingDesc>
   </teiHeader>
   <text>
      <body>
         <p>TEI for manuscript description</p>
         <schemaSpec ident="tei_ms" start="TEI teiCorpus" prefix="tei_" targetLang="en"
            docLang="en">
            <moduleRef key="header"/>
            <moduleRef key="linking"/>
            <moduleRef key="core"/>
            <moduleRef key="tei"/>
            <moduleRef key="textstructure" except="div1 div2 div3 div4 div5 div6 div7"/>
            <moduleRef key="namesdates"/>
            <moduleRef key="transcr"/>
            <moduleRef key="figures"/>
            <moduleRef key="msdescription"/>
            <elementSpec ident="pb" mode="change">
               <attList>
                  <attDef ident="n" mode="change" usage="req"/>
               </attList>
            </elementSpec>
         </schemaSpec>
      </body>
   </text>
</TEI>
```

I find this specification too broad. I would remove some modules. Do you want to describe figures? Are you interested in named entities?

Producing ODD's and schemas with ROMA: exporting the ODD

- Multiple output formats.
- The most important is ODD for scientific purposes.
- You'll need RNC/RNG for validation purposes.
- You won't be able to retrieve your ODD from an RNG or any other schema file.
- Oxygen can also produce the schema from the ODD's.

Making a reference to the schema in your TEI file

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE TEI SYSTEM "/home/mgl/Bureau/These/Edition/hyperregimiento-de-los-principes/Dedans/XML/schemas/Esc_Q.dtd">
<?xml-model href="../../schemas/out/ODD_edition.rng" type="application/xml" schematypens="http://relaxng.org/ns/structure/1.0"?>
<TEI xmlns="http://www.tei-c.org/ns/1.0" xml:id="Esc_Q" type="transcription">
 <teiHeader>
```

Figure: Before the root node, the reference to a DTD (in blue, second line) and to my ODD derived schema (in violet, third line).

A quick look at our ODD

Let's sum up. You can:

- Completely remove elements (they won't be allowed anymore)
- Partially remove elements (they will be allowed in specific contexts only)
- Constrain attribute types, or attribute values; make some attributes mandatory

How does it work ?

Activate a module and include selected elements

```
<moduleRef key="textcrit" include="listWit witness witEnd witStart app rdg"/>
```

- The `include` attribute means a rebuilding of the module from scratch: absent elements won't be allowed in your encoding
- Note here the choice not to include the `lemma` element, derived from a specific editorial methodology ("variance", etc).

Make attributes mandatory

```
<elementSpec ident="pb" mode="change">
  <attList>
     <attDef ident="n" mode="change" usage="req"/>
  </attList>
</elementSpec>
```

Remove attributes at element level

```
<elementSpec ident="material" mode="change" module="msdecription">
    <attList>
        <attDef ident="function" mode="delete"/>
        <attDef ident="target" mode="delete"/>
        <attDef ident="resp" mode="delete"/>
        <attDef ident="ref" mode="delete"/>
    </attList>
</elementSpec>
```

Constrain attribute values

```
<elementSpec ident="material" mode="change" module="msdecription">
    <attList>
        <attDef ident="type" mode="replace" usage="req">
            <valList mode="add" type="closed">
                <valItem ident="parchment">
                    <desc>The copy medium is parchment.</desc>
                </valItem>
                <valItem ident="paper">
                    <desc>The copy medium is paper.</desc>
                </valItem>
                <valItem ident="mixt">
                    <desc>The copy medium alternates between paper and
                    ↪  parchment.</desc>
                </valItem>
            </valList>
        </attDef>
    </attList>
</elementSpec>
```

The other side of ODD's: natural language documentation

- Roma is just the first step to create the machine-readable version of your ODD
- As said before, you'll need to create and write the documentation *in natural language*, using the TEI. Example: here.
- You can use any XML editor to do so.

```
<p>Enfin, il est important pour une bonne intelligence du texte, d'aller marquer des divisions structurelles et
  logiques qui se trouvent au sein des paragraphes. C'est pourquoi je propose d'utiliser des balises vides
    <gi>milestone</gi> qui vont marquer, dans un même paragraphe, des énumérations par exemple (méthode moins
  lourde qu'en utilisant <gi>list</gi>): <egXML xmlns="http://www.tei-c.org/ns/Examples">E <w lemma="este"
      pos="PD0MS000">este</w> es el primero capitulo <w lemma="de+este">desta</w>
    <w pos="NCFS000">parte</w> postrimera que <w lemma="estar" pos="VMIP3S0">esta</w> en tres conclusiones.
      <milestone type="structuration_discours" unit="item_rang_1" n="1"/>La primera es que cosa es <w
      lemma="caballería">caualleria</w>. <milestone type="structuration_discours" unit="item_rang_1" n="2"
    />La <w lemma="2" pos="AO0FS0">segunda</w> por que fue fecha e establecida.</egXML></p>
```

Figure: Documenting the use of `milestone` in the edition of the *Regimiento de los prínçipes*.

An example of complete ODD

- An ODD is an XML-TEI document divided in two parts: the documentation and the schema.
- For structuring, use `div1`...`div9` elements
- You can encode your documentation as you would encode any document
- And put you schema specification (the `schemaSpec`) in a `div1` for instance.

An example of complete ODD

```
      <head>Marques de lecture</head>
      <p>J'indique les marques de lecture de deux façons distinctes: <list>
          <item>La façon la plus simple est d'utiliser l'élément <gi>hi</gi> avec plusieurs valeurs d'attribut
            <att>rend</att> (Ajouter un exemple ici)</item>
          <item>Pour éviter les cas d'overlapping, en marquant des marques de lecture de différent niveau,
            j'utilise les balises auto-fermantes <gi>anchor</gi>, qui me permettent de marquer le début et la
            fin d'un passage. Ces balises sont typées (<att>type</att>) et contiennent un sous-type permettant
            de distinguer leur style (sous la ligne, en marge), ainsi qu'un attribut (<att>next</att> ou
              <att>previous</att>) pour indiquer si elles indiquent le début ou la fin de la marque de
            lecture.</item>
        </list><egXML xmlns="http://www.tei-c.org/ns/Examples">sinon <anchor next="#a2SAcLX9bbVCkXl"
            xml:id="a3dutf7XDj8d" type="marque_lecture" subtype="souligne"/>agua salada, aquella <w lemma="agua"
            pos="NCFS000">agua</w> pueden fazer dulçe colándola por la çera, faziendo baçines de çera, et<lb
            break="yes"/><add type="commentaire" place="margin" hand="distincte">agua de
            <unclear>mar</unclear></add> foraçándolos con agujas sotiles de muchos <w lemma="forado">forados</w>
          e espessos.<anchor prev="#a3dutf7XDj8d" xml:id="a2SAcLX9bbVCkXl" type="marque_lecture"
            subtype="souligne"/> Ca segunt <w lemma="el" pos="DA0MS0">el</w>
          <w lemma="filósofo" pos="NCMS000">filósofo</w></egXML></p>
      </div4>
    </div3>
    <div3>
      <head>Éléments utilisés dans le cadre de la citation du texte</head>
      <p>La citation de fragments du texte se fait par copie et traitement de nœuds XML entre éléments
        <gi>anchor</gi>. Ces éléments sont de type <val>citation</val>. Des <gi>anchor</gi> de type
          <val>reference</val> sont utilisés pour citer une ligne précise du texte, et des <gi>anchor</gi> de type
          <val>reference</val> sont utilisés pour le texte de façon plus générale. Ces éléments doivent
        impérativement être identifiés par un <att>xml:id</att>.</p>
    </div3>
    <div3> [5 lines]
  </div2>
</div1>
<div1 type="Encodage">
  <head>Manuel d'encodage</head>
  <schemaSpec ident="oddbyexample" start="TEI teiCorpus" targetLang="fr" docLang="fr">
    <moduleRef key="tei"/>
    <!--Checking module textcrit-->
    <classRef key="att.textCritical"/>
    <classRef key="model.rdgLike"/>
    <classRef key="model.rdgPart"/>
    <moduleRef key="textcrit" include="listWit witness witEnd witStart app rdg"/>
```

Figure: Documenting the use of `milestone` in the edition of the *Regimiento de los prínçipes*.

Fine-tune the validation with Schematron

```
<!--The following element can be a child of any elementSpec node-->
<constraintSpec xmlns:sch="http://purl.oclc.org/dsdl/schematron"
↪  scheme="isoschematron">
    <constraint>
      <sch:rule context="pb">
         <sch:assert
            test="matches(@n, '\d+[vr]')"
            >Something is wrong with the foliation of this page. Have you
            ↪  forgotten the page side indication ?</sch:assert>
      </sch:rule>
    </constraint>
</constraintSpec>
```

Conclusions

- Schemas even minimal, are essential for any scientific publishing project based on TEI.
- TEI-based editions without the publication of XML sources is quite unuseful and won't last much
- Please provide access to your data (and its documentation) !

Exercises

- The exercise instructions are in the xml file `bentham.xml`

ODDbyexample

- ODDByExample is a tool to produce an ODD using an already existing XML-TEI document
- It will create the ODD that matches this document
- Very useful tool ...
- ... that can be tricky if the XML-TEI example shows inconsistency: they will be embedded into your specification
- Make sure to use this tool with small, carefully checked fragments of your document.
- Link: https://github.com/TEIC/Stylesheets/blob/dev/tools/oddbyexample.xsl

References

Burnard, Lou. “What Is TEI Conformance, and Why Should You Care?” In: *Journal of the Text Encoding Initiative* 12 (2019). ISSN: 2162-5603.

Marín Sánchez, Ana María. “La Versión Interpolada de Los Castigos de Sancho IV: Edición y Estudio”. PhD thesis. Universidad de Zaragoza, 2003.