Talk 13: Customising and Documenting Your TEI project

James Cummings

28 January 2014

Customising the TEI

- How the TEI is constructed
- Making a TEI schema
- Specifying your profile of the TEI
- Generating your own documentation

Every use of the TEI involves making use of a customisation of the TEI.

Some terminology

- The TEI encoding scheme consists of a number of *modules*
- Each module contains a number of *element specifications*
- Each element specification contains:
  - a canonical name (<gi>) for the element, and optionally other names in other languages
  - a canonical description (also possibly translated) of its function
  - a declaration of the *classes* to which it belongs
  - a definition for each of its *attributes*
  - a definition of its *content model*
  - usage examples and notes
- a TEI *schema* specification (<schemaSpec>) is made by selecting modules or elements and (optionally) modifying their contents
- a TEI document containing a schema specification is called an *ODD* (One Document Does it all)

What is a module?

- A convenient way of grouping together a number of element declarations
- These are usually on a related topic or specific application
- Most chapters of P5 focus on elements drawn from a single module, which that chapter then defines
- A TEI Schema is created by selecting modules and adding or removing elements from them as needed

Which modules exist?

Module name	Chapter
analysis	Simple Analytic Mechanisms
certainty	Certainty and Responsibility
core	Elements Available in All TEI Documents
corpus	Language Corpora
dictionaries	Dictionaries
drama	Performance Texts
figures	Tables, Formulae, and Graphics
gaiji	Representation of Non-standard Characters and Glyphs
header	The TEI Header
iso-fs	Feature Structures
linking	Linking, Segmentation, and Alignment
msdescription	Manuscript Description
namesdates	Names, Dates, People, and Places
nets	Graphs, Networks, and Trees
spoken	Transcriptions of Speech
tagdocs	Documentation Elements
tei	The TEI Infrastructure
textcrit	Critical Apparatus
textstructure	Default Text Structure
transcr	Representation of Primary Sources
verse	Verse

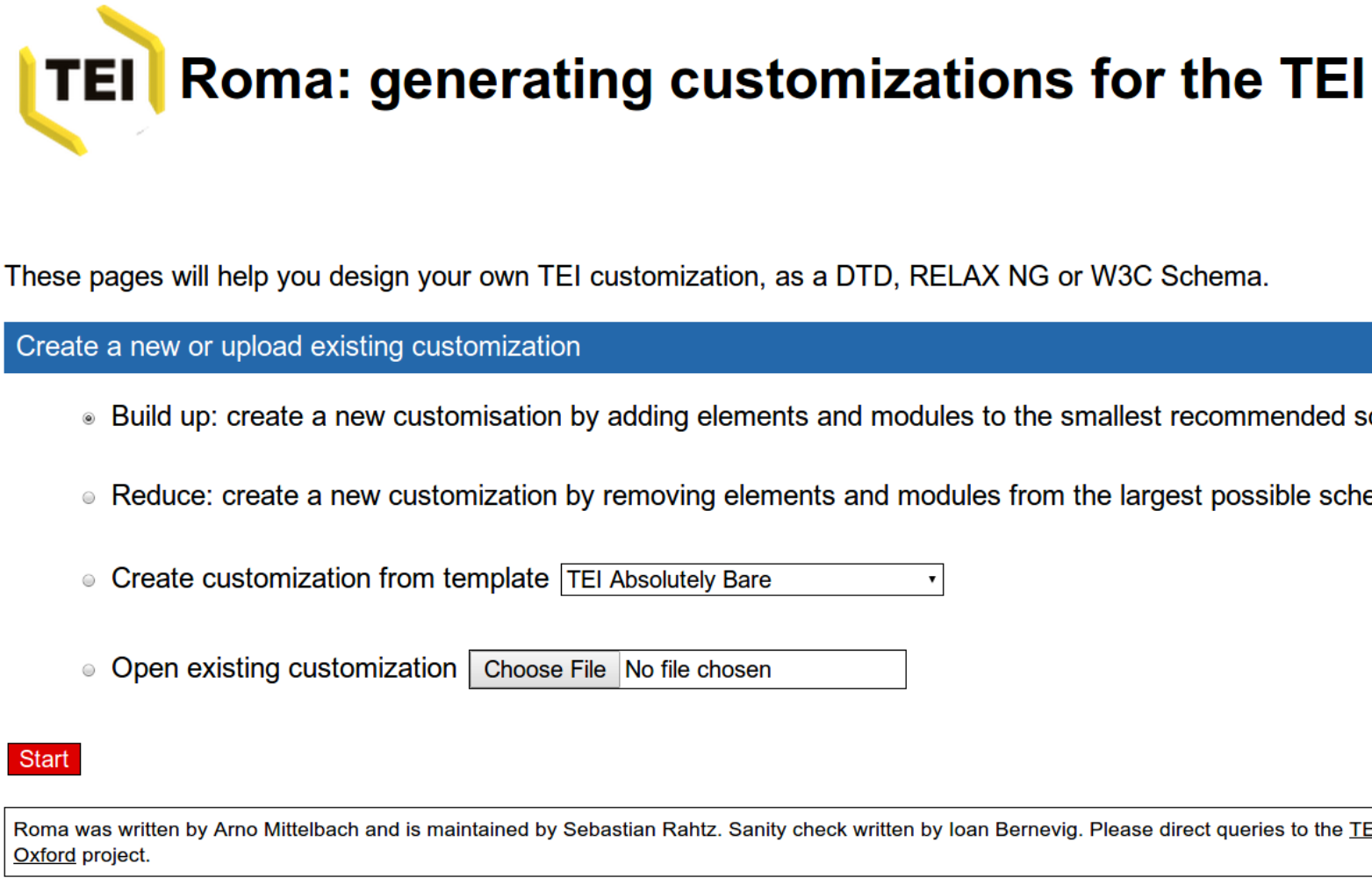
How do you choose?

- Just choose everything (not really a good idea)
- The TEI provides a small set of predefined combinations (TEI Lite, TEI Bare...)
- Or you could roll your own (but then you need to know what you're choosing)

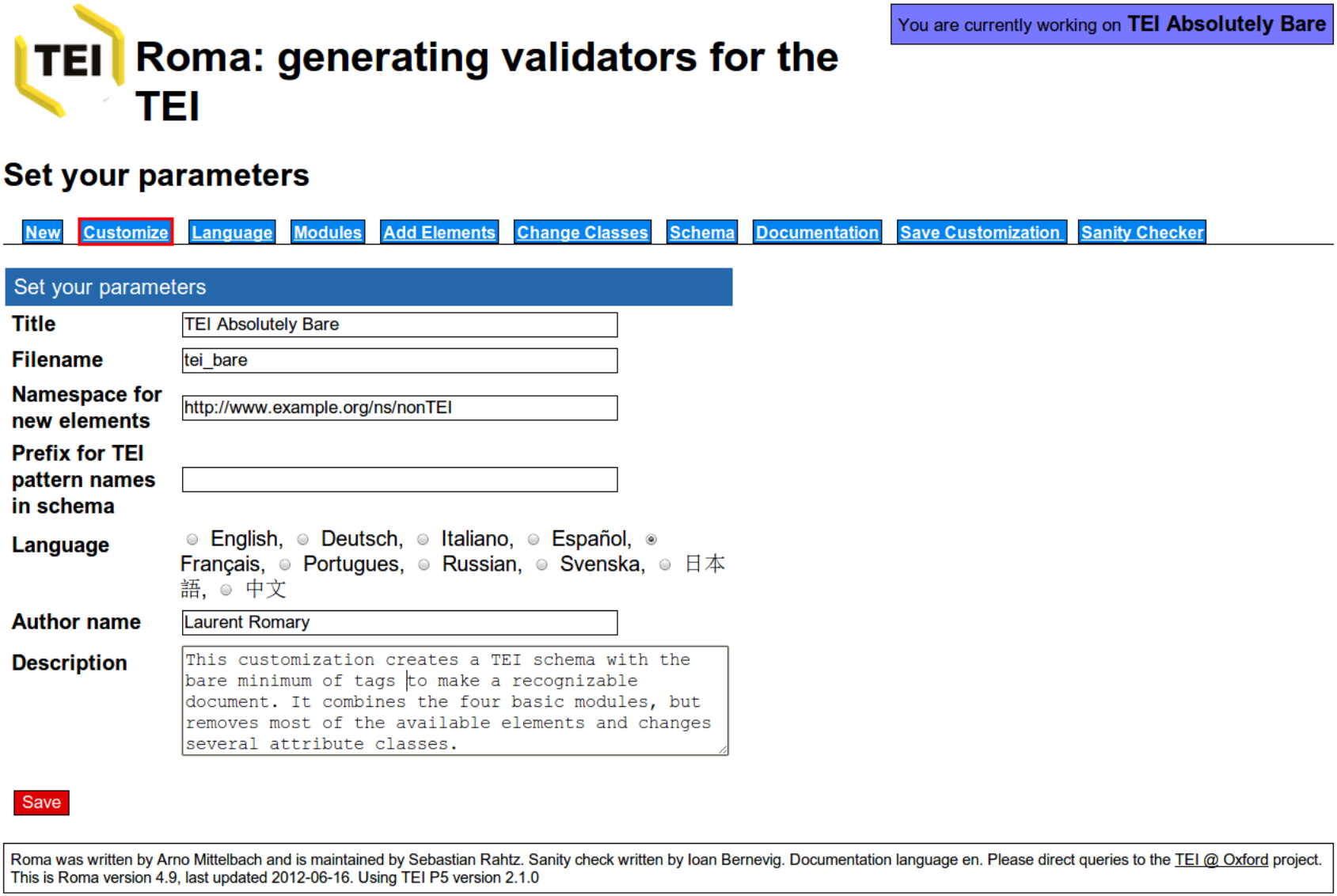
Roma a command line script, with a web front end, designed to make this process much easier

http://www.tei-c.org/Roma/

Roma: New


TEI
Roma: generating customizations for the TEI
These pages will help you design your own TEI customization, as a DTD, RELAX NG or W3C Schema.
Create a new or upload existing customization
Build up: create a new customisation by adding elements and modules to the smallest recommended schema
Reduce: create a new customization by removing elements and modules from the largest possible schema
Create customization from template
TEI Absolutely Bare
Open existing customization
Choose File
No file chosen
Start
Roma was written by Arno Mittelbach and is maintained by Sebastian Rahtz. Sanity check written by Ioan Bernevig. Please direct queries to the TEI @ Oxford project.

Roma: Customize



Roma: Schema

Vous travaillez actuellement sur **TEI Absolutely Bare**

Roma: Production des validateurs TEI

Enfin je vous rends votre schéma...

Nouvelle | Personnaliser | Langage | Modules | Ajouter des éléments | Modifier les classes | Schéma
Documentation | Enregistrer | Contrôleur de validité

Creation du schéma en cours

Quel format préfériez-vous?

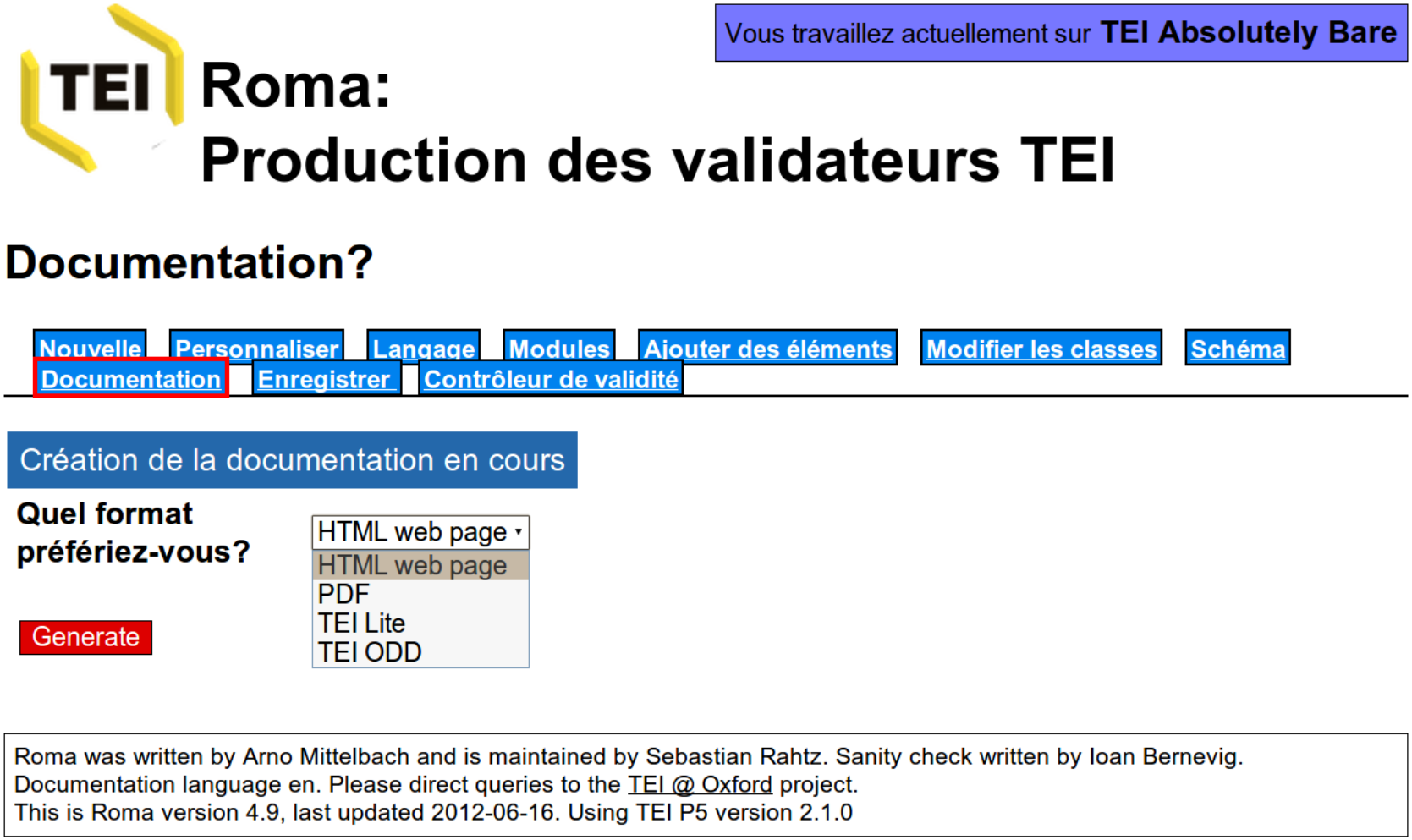
RELAX NG schema (compact syntax)
RELAX NG schema (compact syntax)
RELAX NG schema (XML syntax)
ISO Schematron
Schematron
W3C schema (in ZIP archive)
DTD

Generate

We use RELAX NG
http://en.wikipedia.org/wiki/XML_Schema_Language_comparison.

Roma was written by Arno Mittelbach and is maintained by Sebastian Rahtz. Sanity check written by Ioan Bernevig.
Documentation language en. Please direct queries to the TEI @ Oxford project.
This is Roma version 4.9, last updated 2012-06-16. Using TEI P5 version 2.1.0

Roma: Documentation



What did we just do?

We processed a pre-existing ODD file which contained (as well as some discursive prose) the following schema specification:

```
<schemaSpec ident="tei_bare" start="TEI">
  <moduleRef key="core"/>
  <moduleRef key="tei"/>
  <moduleRef key="header"/>
  <moduleRef key="textstructure"/>
  <elementSpec ident="abbr" mode="delete" module="core"/>
  <elementSpec ident="add" mode="delete" module="core"/>
<!-- ... -->
  <elementSpec ident="trailer" mode="delete" module="textstructure"/>
  <elementSpec ident="title" mode="change" module="core">
    <attList>
      <attDef ident="level" mode="delete"/>
    </attList>
  </elementSpec>
<!-- ... -->
</schemaSpec>
```

We selected four modules, deleted loads of elements, and also deleted an attribute.

Roma provides an interface to the detail

- The [Modules] tab shows the modules available
- Selecting a module from it shows the elements within that module, and gives you the choice to
  - include all of them (and then remove some)
  - exclude all of them (and then put back the ones you want)
- You can also change an element's attribute list, and the values they permit

Roma: Modules

Liste des modules TEI

	Titre du module		Description brève	Modification
ajouter	analysis	?	Mécanismes analytiques simples	
ajouter	certainty	?	Degré de certitude et responsabilité	
ajouter	core	?	Éléments disponibles pour tous les documents TEI	modifié
ajouter	corpus	?	Corpus linguistiques	
ajouter	dictionaries	?	Dictionnaires	
ajouter	drama	?	Théâtre	
ajouter	figures	?	Tableaux, formules et graphiques	
ajouter	gaiji	?	Représentation des caractères et des glyphes non standard	
ajouter	header	?	En-tête TEI	modifié
ajouter	iso-fs	?	Structures de traits	
ajouter	linking	?	Liens, segmentation et alignement	
ajouter	msdescription	?	Description de manuscrits	
ajouter	namesdates	?	Noms, dates, personnes et lieux	
ajouter	nets	?	Graphes, réseaux et arbres	
ajouter	spoken	?	Transcriptions de la parole	
ajouter	tagdocs	?	Éléments de déclaration d'un modèle	
ajouter	textcrit	?	Apparat critique	
ajouter	textstructure	?	Structure textuelle par défaut	modifié
ajouter	transcr	?	Représentation de sources primaires	
ajouter	verse	?	Poésie	

Liste des Modules selectionnés

supprimer	core
	tei
supprimer	header
supprimer	textstructure

Roma: Change Module

Liste des éléments compris dans ce modulecore

	Inclure	Exclure	Nom		Description	Attributs
abbr	○	◉	abbr	?	(abréviation) contient une abréviation quelconque.	Changer les attributs
add	○	◉	add	?	(ajout) contient des lettres, des mots ou des phrases insérées dans le texte par un auteur, un copiste, un annotateur ou un correcteur.	Changer les attributs
addrLine	○	◉	addrLine	?	(ligne d'adresse) contient une ligne d'adresse postale.	Changer les attributs
address	○	◉	address	?	contient une adresse postale ou d'un autre type, par exemple l'adresse d'un éditeur, d'un organisme ou d'une personne.	Changer les attributs
analytic	○	◉	analytic	?	(niveau analytique) contient des éléments descriptifs qui décrivent la bibliographie d'une ressource (par exemple un poème ou un article de revue) publiée à l'intérieur d'une monographie ou d'une ressource et non publiée de façon indépendante.	Changer les attributs
author	◉	○	author	?	(auteur) dans une référence bibliographique contient le nom de la (des) personne(s) physique(s) ou du collectif, auteur(s) d'une oeuvre ; la première mention de responsabilité comme seul élément bibliographique.	Changer les attributs
bibl	○	◉	bibl	?	(référence bibliographique.) contient une référence bibliographique faiblement structurée dans laquelle les	Changer les attributs

What does our REED corpus need?

A simple selection of elements, but also

- we want to allow only certain values for *@type* on
- we may want to create a new element (can you think of something?)

Other constraints are possible — we might want to insist that a `<div type="record">` contains a date, for example.

The ODD advantage

We can express these constraints in our ODD meta-schema, and then generate a formal schema to enforce them using whichever schema language we like.

- TEI schemas can be generated in
  - ISO RELAX NG language
  - W3C Schema Language
  - XML DTD language
- ODD itself defines an element's content models using a subset of RELAX NG syntax
- Datatypes are defined in terms of W3C datatypes
- Some facilities (e.g. alternation, namespaces) cannot be expressed in DTDs — RELAX NG schema is recommended
- Additional constraints can be expressed in Schematron

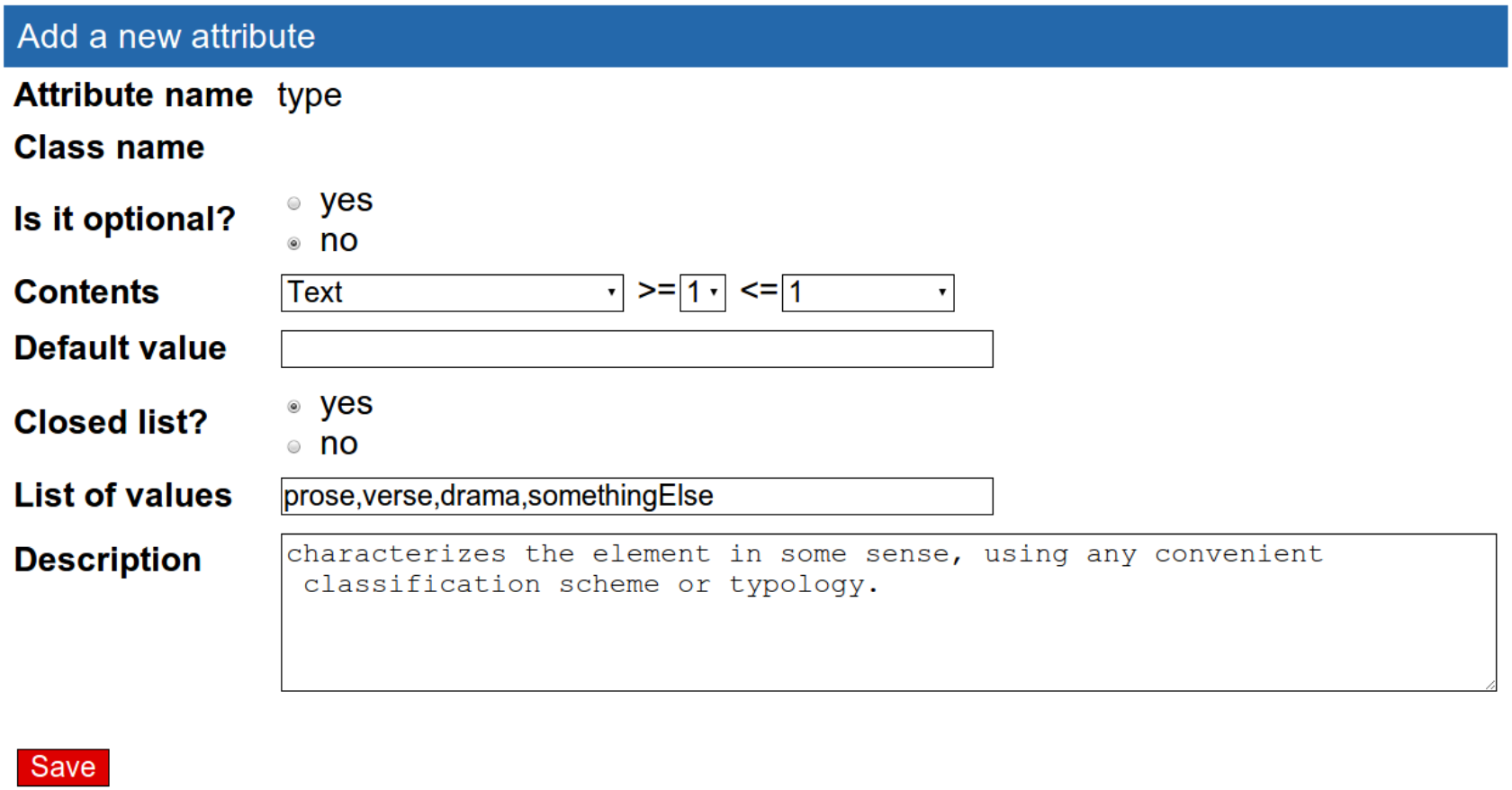
Roma: selecting attributes

List of attributes: div

Add new attributes

Change attribute	Include	Exclude	Name	Description	Delete
ana	◉	○	ana	indicates one or more elements containing interpretations of the element on which the ana attribute appears.	
change	◉	○	change	points to one or more change elements documenting a state or revision campaign to which the element bearing this attribute and its children have been assigned by the encoder.	
copyOf	◉	○	copyOf	points to an element of which the current element is a copy.	
corresp	◉	○	corresp	points to elements that correspond to the current element in some way.	
decls	◉	○	decls	identifies one or more declarable elements within the header, which are understood to apply to the element bearing this attribute and its content.	
exclude	◉	○	exclude	points to elements that are in exclusive alternation with the current element.	
facs	◉	○	facs	points to all or part of an image which corresponds with the content of the element.	
met	◉	○	met	contains a user-specified encoding for the conventional metrical structure of the element.	
n	◉	○	n	gives a number (or other label) for an element, which is not necessarily unique within the document.	

Roma: constraining attribute values



What did we just do?

Our ODD now includes something like this:

```
<elementSpec ident="div" mod-
ule="textstructure" mode="change">
 <attList>
  <attDef ident="type" mode="change" usage="req">
   <valList type="closed" mode="replace">
    <valItem ident="prose"/>
    <valItem ident="verse"/>
    <valItem ident="drama"/>
<!-- ... -->
   </valList>
  </attDef>
 </attList>
</elementSpec>
```

Note that we can also add documentation to the ODD:

```
<valItem ident="verse">
 <gloss>contains (parts of ) a poem</gloss>
</valItem>
```

Defining a new element

When defining a new element, we need to consider

- its name and description
- what attributes it can carry
- what it can contain
- where it can appear in a document

The TEI class system helps us answer all these questions (except the first).

The TEI Class System

- The TEI distinguishes over 500 elements,
- Having these organised into classes aids comprehension, modularity, and modification.
- *Attribute class*: the members share common attributes
- *Model class*: they can appear in the same locations (and are often semantically related)
- Classes may contain other classes
- An element can be a member of any number of classes, irrespective of the module it belongs to.

Attribute Classes

- Attribute classes are given (usually adjectival) names beginning with att.; e.g. *att.naming*, *att.typed*
- all members of `att.naming` inherit from it attributes *@key* and *@ref*; all members of `att.typed` inherit from it *@type* and *@subtype*
- If we want an element to carry the *@type* attribute, therefore, we add the element to the `att.typed` class, rather than define those attributes explicitly.

A very important attribute class: att.global

All elements are usually members of att.global; this class provides, among others:

@xml:id	a unique identifier
@xml:lang	the language of the element content
@n	a number or name for an element
@rend	how the element in question was rendered or presented in the source text.

Model Classes

- Model classes contain groups of elements which are allowed in the same place. e.g. if you are adding an element which is wanted wherever the <bibl> is allowed, add it to the model.biblLike class
- Model classes are usually named with a Like or Part suffix:
  - members of model.pLike are all things that 'behave like' paragraphs, and are permitted in the same places as paragraphs
  - members of model.pPart are all things which can appear *within* paragraphs. This class is subdivided into
    - model.pPart.edit elements for simple editorial intervention such as <corr>, <del> etc.
    - model.pPart.data'data-like' elements such as <name>, <num>, <date> etc.
    - model.pPart.msdesc extra elements for manuscript description such as <seal> or <origPlace>

Basic Model Class Structure

Simplifying wildly, one may say that the TEI recognises three kinds of element:

divisions	high level major divisions of texts
chunks	elements such as paragraphs appearing within texts or divisions, but not other chunks
phrase-level elements	elements such as highlighted phrases which can occur only within chunks

There are 'base model classes' corresponding with each of these, and also with the following groupings:

inter-level elements	elements such as lists which can appear either in or between chunks
components	elements which can appear directly within texts or text divisions

And yes, there is a class model.global for elements that can appear *anywhere* inside a text — at any hierarchic level.

Defining a new element

What other elements is it like?

What other elements can contain it?

What can it contain?

Conclusions:

- What model do we select?
- What content model do we select?

Roma: Defining a new element

Defining a new element:

Name

Namespace http://www.example.org/ns/nonTEI

Description

Model classes

- model.addrPart
- model.addressLike
- model.applicationLike
- model.availabilityPart
- model.biblLike
- model.biblPart
- model.castItemPart
- model.catDescPart
- model.certLike
- model.choicePart
- model.common
- model.dateLike
- model.dimLike
- model.div1Like
- model.div2Like
- model.div3Like
- model.div4Like
- model.div5Like
- model.div6Like
- model.div7Like
- model.divBottom
- model.divBottomPart
- model.divGenLike
- model.divLike
- model.divPart
- model.divPart.spoken
- model.divTop
- model.divTopPart
- model.divWrapper
- model.editorialDeclPart
- model.egLike
- model.emphLike
- model.encodingDescPart
- model.entryLike

Defining a content model

- A typical TEI element defines its content by referencing *classes* of element which it can contain, rather than using specific elements.
- Content models are defined using the RELAXNG vocabulary
- Here are some very common predefined content models:

  macro.paraContent content of paragraphs and similar elements

  macro.limitedContent content of prose elements that are not used for transcription of extant materials

  macro.phraseSeq a sequence of character data and phrase-level elements

  macro.phraseSeq.limited a sequence of character data and those phrase-level elements that are not typically used for transcribing extant documents

  macro.specialPara the content model of elements which either contain a series of component-level elements or else contain a series of phrase-level and

Roma: Defining a new element 2

Attribute classes

- att.ascribed
- att.breaking
- att.canonical
- att.combinable
- att.coordinated
- att.damaged
- att.datable
- att.datable.custom
- att.datable.iso
- att.datable.w3c
- att.datcat
- att.declarable
- att.declaring
- att.dimensions
- att.divLike
- att.docStatus
- att.duration
- att.duration.iso
- att.duration.w3c
- att.editLike
- att.enjamb
- att.entryLike
- att.global
- att.global.analytic
- att.global.change
- att.global.facs
- att.global.linking
- att.handFeatures
- att.identified
- att.internetMedia
- att.interpLike
- att.lexicographic
- att.measurement
- att.metrical
- att.msExcerpt
- att.naming
- att.personal
- att.placement
- att.pointing
- att.pointing.group
- att.ranging
- att.rdgPart
- att.readFrom
- att.responsibility
- att.scoping
- att.segLike
- att.sortable
- att.sourced
- att.spanning
- att.tableDecoration
- att.textCritical
- att.timed
- att.transcriptional
- att.translatable
- att.typed
- att.witnessed

Contents

User content

```
<content xmlns:rng="http://relaxng.org/ns/structure/1.0">
</content>
```

Save

What did we just do?

We added a new element specification to our ODD, like this:

```
<elementSpec
  ident="something"
  ns="http://www.example.org/ns/nonTEI"
  mode="add">
 <desc> contains something division like.</desc>
 <classes>
  <memberOf key="model.divPart"/>
  <memberOf key="att.typed"/>
 </classes>
 <content>
  <rng:ref name="something"/>
  <rng:oneOrMore>
   <rng:ref name="model.pLike"/>
  </rng:oneOrMore>
 </content>
</elementSpec>
```

Note that this new element is *not* in the TEI namespace. It belongs to this specific project only!

Other kinds of constraints

- You can also constrain the content of an element or the value of an attribute to be of a particular *datatype* (for example, to insist that the *@when* attribute of the element <date> contains only a date)
- This can be done by using one of a set of predefined *macros* to define the content. Examples include

  data.word a single word or token
  data.name an XML Name
  data.enumerated a single XML name taken from a documented list
  data.temporal.w3c a W3C date
  data.truthValue a truth value (true/false)
  data.language a human language
  data.sex human or animal sex

- Or you can define a more complex constraint, e.g. using Schematron

Schematron constraints

- An element specification can also contain a <constraintSpec> element which contains rules about its content expressed as ISO Schematron *constraints*

```
<elementSpec ident="div" mod-
ule="teistructure" mode="change"
   xmlns:s="http://purl.oclc.org/dsdl/schematron">
 <constraintSpec ident="div" scheme="isoschematron">
  <constraint>
   <s:assert test="@type='record' and .//tei:date">prose
must include a paragraph
   </s:assert>
  </constraint>
 </constraintSpec>
</elementSpec>
```

However...

- You can only add such rules by editing your ODD file: Roma doesn't know about them.
- Not all schema languages can implement these constraints.

TEI customization for fun and profit

With the Stationers' Register project the keying company would:

- do double-keying data entry matching any XML schema
- and charged per kilobyte of output!

So we:

- used TEI ODD to create a highly compressed / byte-reduced renaming schema
- and up-converted the result back to pure TEI!

tei_corset example

```
<nm r="rm">iijli vjs viijd</nm> from

<n r="b">Thomas barthelett</n>
```

was up-converted to

```
<seg type="notFee" rend="roman-numerals">
  <num type="totalPence" value="800">
    <num type="poundsAsPence" value="720"> iij<hi rend="superscript">li</hi>
    </num>
    <num type="shillingsAsPence" value="72"> vj<hi rend="superscript">s</hi>
    </num>
    <num type="pence" value="8"> viij<hi rend="superscript">d</hi>
    </num>
  </num>
</seg> from
<persName role="stationer" rend="bold">
  <forename>Thomas</forename>
  <surname>barthelett</surname>
</persName>
```

Next

Any Questions? That is a quick look at some of the basic things one can do with the TEI ODD language, and the Roma web tool.

Now let's do an exercise where we try out customising the TEI!