TEI Simple

SELF-DOCUMENTING
ABSTRACTION LAYER
FOR
XML PROCESSING

TEI vs TEI Simple

TEI	TEI Simple
• concentrates on data modelling aspects and ideological agnosticism • avoids standardization constraining individual projects • avoids any recommendation for processing & publishing • TEI stylesheets often too complicated to customize • only concerned with TEI	• concentrates on eliminating the ambiguity for the encoder • aims at interoperability • provides out-of-the-box mechanism for default processing • allows for [relatively] easy customization and extension • TEI Simple processing model can be applied to other vocabularies

TEI Simple ODD

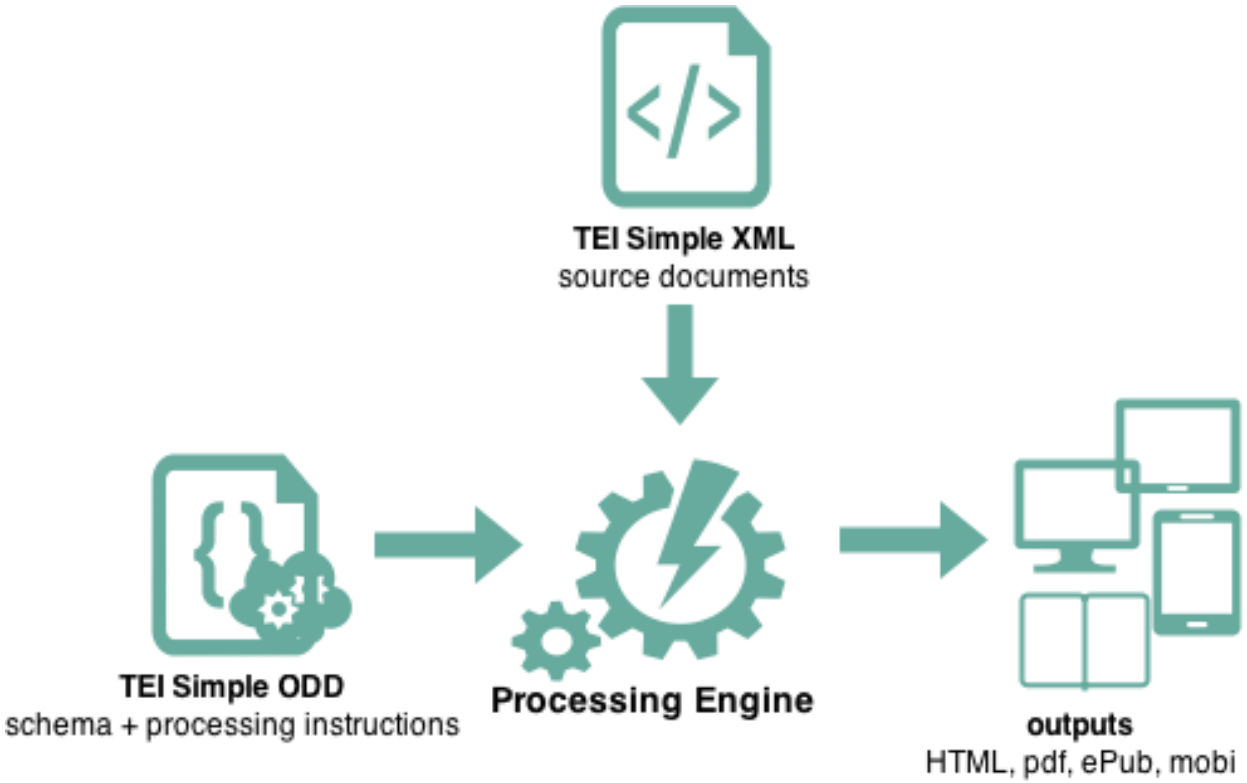
=

a schema

+

processing instructions

for all elements



TEI Simple processing model

Rahtz Rationale

workflow with three distinct roles

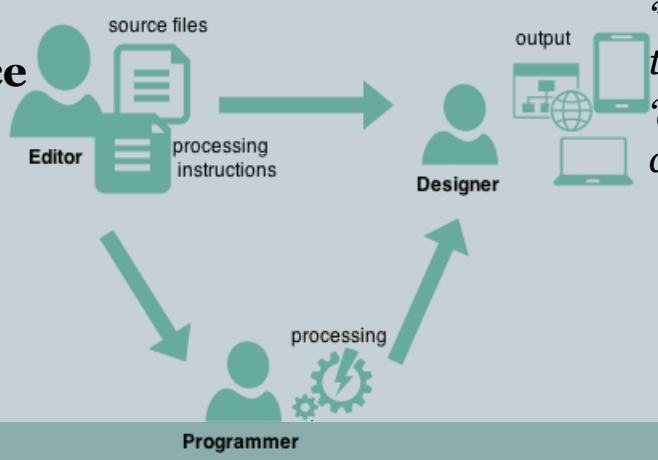
Editor

manages the text integrity, makes the high-level output decisions:

- **structural** descriptions
  '*should the original or corrected version be displayed by default*', or '*is this a block level or inline component*'
- indications of **appearance**
  '*titles are in italics*'

Programmer

takes the editor's specification, and the TEI text(s), and creates the input for the designer to make the output



Designer

creates the output envelope (for example book layout using InDesign, or a web site using Drupal), making decisions about the appearance in conjunction with the editor

'use Garamond font throughout'

'every page must show the departmental logo'

simple scenarios

default	customized
• editorial decisions recommended by teiSimple fit project's needs perfectly: just use teisimple.odd	• project requires customization: overwrite teisimple.odd with custom processing and rendition instructions

Turska Tenet

maximum expressivity to the editor

ODD stores as much information as possible

for each element there are potentially numerous <model> instructions that specify intended processing and rendering for different outputs and in various contexts

```
<elementSpec mode="change" ident="choice">
    <model output="plain" predicate="sic and corr" behaviour="inline">
        <param name="content">corr[1]</param>
    </model>
    <model output="plain" predicate="abbr and expan" behaviour="inline">
        <param name="content">expan[1]</param>
    </model>
    <model output="plain" predicate="orig and reg" behaviour="inline">
        <param name="content">reg[1]</param>
    </model>
    <model predicate="sic and corr" behaviour="alternate">
        <param name="default">corr[1]</param>
        <param name="alternate">sic[1]</param>
    </model>
    <model predicate="abbr and expan" behaviour="alternate">
        <param name="default">expan[1]</param>
        <param name="alternate">abbr[1]</param>
    </model>
    <model predicate="orig and reg" behaviour="alternate">
                                                </param>
                                               [1]</param>
```

```
<elementSpec mode="change" ident="div">
    <model predicate="@type='title_page'" behaviour="block">
        <rendition>border: 1px solid black; padding: 5px;</rendition>
    </model>
    <model behaviour="section" predicate="parent::body or parent::front or parent::back"/>
    <model behaviour="block"/>
</elementSpec>
```

editor's tasks (apart from editing)

- identify elements that require individual treatment
- if treatment differs depending on context, identify all possible situations via XPath expressions (eg. div type="act" headings treated differently than all other head elements)
- decide which behaviour is required under given circumstances, specify parameters (eg. use only lem child as visible by default in app entries)
- if treatment differs depending on output type create additional models with *@output* as necessary
- specify rendition as CSS where required

model syntax

ODD extensions for TEI Simple Processing Model (PM)

model

param
rendition
modelGrp
modelSeq

model

- *@output* specifies output in which model applies
- *@predicate* specifies context in which model applies
- *@behaviour* specifies function from *teiSimple function library* to apply
- *@useSourceRendition* indication to preserve *@rendition* value from the source
- <*param*> elements specify parameters for *behaviour*
- <*rendition*> elements specify CSS instructions to indicate outline appearance

TEI Simple *behaviours* library

handful of functions that achieve > 80% tasks

functions based on commonly used terms

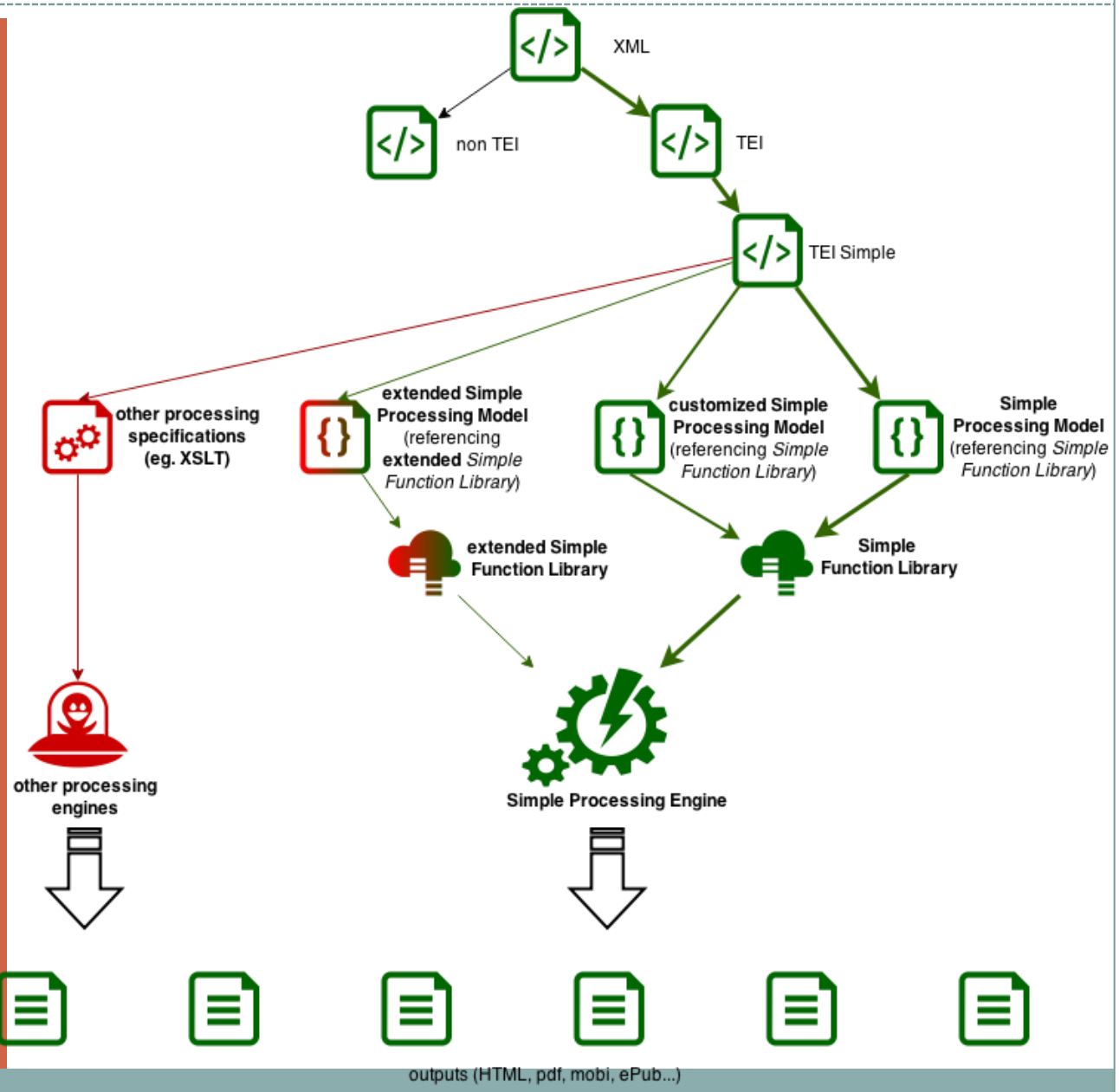
if no content parameter specified, all functions use current element as default content

alternate (default,alternate)	create a specialized display of alternating elements for displaying the preferred version, both at once or toggling between the two.
anchor (id)	create anchor with ID
block (content)	create a block out of the content parameter.
body (content)	create the body of a document.
break (type,label)	make a line, column, or page break according to type
cell (content)	create a table cell
cit (content,source)	show the content, with an indication of the source
document (content)	start a new output document
figure (title)	make a figure with the title as caption
glyph (content)	show a character by looking up reference
graphic (url)	if url is present, uses it to display graphic, else display a placeholder image.
heading (content)	creates a heading.
index (type)	generate list according to type
inline (content,label)	creates inline element out of content if there's something in rendition, use that formatting otherwise just show text of selected content.
link (content,target)	create hyperlink
list (content)	create a list by following content
listItem (content)	create list item
metadata (content)	create metadata section
note (content,place,marker)	create a note, according to value of place; could be margin, footnote, endnote, inline
omit	do nothing, do not process children
paragraph (content)	create a paragraph out of content.
row (content)	make table row
section (content)	create a new section of the output document
table (content)	make table
text (content)	literal text
title (content)	make document title

model

- there can be as many <*model*> statements as required, each of which can have multiple <*rendition*> children
- set of multiple <*model*> statements is regarded as an *alternation* and only the first <*model*> with *@predicate* matching current context is applied
- *@behaviour* specifies which one from *TEI Simple function library* should be applied and function parameters are specified as <*param*> children
- within each <*model*> there can be set of <*rendition*> elements
- <*desc*> allows for initial textual description of required processing

TEI Simple in a broader context



editor's necessary skills

- familiarity with source files and XML encoding used
- ability to identify different use scenarios
- relative XPath fluency to specify model parameters*
- relative CSS fluency*

* possibly assisted by project's IT support