TEI Simple: state of the nation

TEI Simple Workteam (SR, BPZ, MM, MT)

March 2015

What we do want to do today?

1. Report on progress against TEI Simple objectives
2. Discuss useability of Processing Model
3. Demonstrate implementation of Processing Engine (Wolfgang Meier)
4. Consider communications plan
5. Understand what DTA is doing

Reminder of objectives

1. The highly constrained and prescriptive element subset of TEI Simple
2. The processing model (Simple Processing Model: SPM)
3. Formal mapping of the TEI elements used by Simple to the CIDOC CRM/FRBRoo
4. TEI-Performance Indicators
5. Integration of TEI Simple into the TEI infrastructure

Deliverables

1. A TEI ODD customization with the choices and constraints, PM notation, and RDF mapping
2. Multi-stage tutorial documentation based on examples using page images and XML
3. Implementation(s) of the Processing Engine (PE)

These outputs will be offered to the TEI Technical Council to decide on how to best to incorporate and maintain them

TEI Simple: original work plan

October	Complete customization with closed value lists for attributes	**DONE**
November/ December	Complete definition of the SPM	**DONE**
December/ January/ February	Implementation of the SPM	**DONE (proof of concept)**
January/ February	Documentation and examples	**NOT COMPLETED**
March	TEI Performance indicators	**STARTED**
April	Mapping to RDF	**TO BE DONE**

Progress on TEI Simple subset

- 114 TEI Simple elements located in the <body> of a text. (A lesson from the MONK Project about unecessary variation.)
- Decided to leave <teiHeader> alone (rightly?)
- Customization maintains simplicity with closed value lists for selected attribute classes (e.g., <name>, <cell>, <row>).
- Completed conversion routine to TEI Simple from normal TEI files.
- Schematron <constraint>s make it possible to check any special constraints specified in the TEI-simple ODD file that are not already required in the TEI Simple schema.
- Besides Schematron, validation against a TEI Simple RelaxNG schema, using Jing.
- Schematron and Jing validation occur in a single Ant task (Ant runs in oXygen, on many platforms, and efficiently uses a single instance of the Java Virtual Machine).
- Tested on whole of EEBO etc. for all of TCP 60K+ files (EEBO, ECCO, Evans)

Reminder: how does processing of Simple work

The model assumes a workflow with three parts, and three distinct roles, to create a digital edition from a TEI text:

1. an *editor* manages the text integrity, makes the high-level output decisions. These fall into two parts
    1. *Structural* descriptions, eg 'should the original or corrected version be displayed by default', or 'is this a block level or inline component'
    2. Indications of *appearance* ('titles are in italics').
2. a *designer* creates the output envelope (for example, and book layout using InDesign, or a web site using Drupal), making decisions in conjunction with the editor ('use Garamond font throughout' or 'every page must show the departmental logo')
3. a *programmer* takes the *editor*'s specification, and the TEI text(s), and creates the input for the *designer* to make the output.

Simple processing model: the PM

- workflow (flowchart)
- extensions to TEI model etc
- example PM for simple elements
- implementation of PE to HTML in XSLT

Processing Model: how does that work?

1. TEI Simple schema ODD contains instructions for intended processing of TEI Simple elements
2. instructions are simple XML fragments stating which task from a limited TEI Simple function library should be executed when processing that element
3. multiple processing instructions may occur to define expected behaviour in various contexts or output formats
4. Processing Engine is able to transform TEI Simple source documents into output format(s) according to processing instructions from TEI Simple ODD

Necessary changes to TEI

TEI ODD language needs to be extended with following elements and attributes (proposal pending consideration by TEI Council):

- <model> *@behaviour* (specifies the actions to undertake), *@predicate* (circumstances where this rule applies), *@output* (output for which rule applies), *@useSourceRendition* (should processing take into account original rendition)
- <modelSequence> (groups sequences of tasks that are to be executed)
- <modelGrp> (groups <model>s and <modelSequence>s for readability and conciseness
- <rendition> (specifies the formatting of the output)

TEI Simple Processing model example

TEI Simple schema ODD has been enhanced with processing models for TEI Simple elements :

```
<elementSpec mode="change"
 ident="note">
 <model predicate="@place"
  behaviour="note(.,@place)"/>
 <model predicate="parent::div and not(@place)"
  behaviour="block(.)">
  <rendition>margin-left: 10px;margin-right: 10px;
font-size:smaller;</rendition>
 </model>
 <model predicate="not(@place)"
  behaviour="inline(.)">
  <rendition scope="before">content:" [";</rendition>
  <rendition scope="after">content:"] ";</rendition>
  <rendition>font-size:small;</rendition>
 </model>
</elementSpec>
```

TEI Simple Processing model example (2)

```
<elementSpec mode="change" ident="q">
 <model predicate="l"
  behaviour="block(.)" useSourceRendition="true">
  <rendition>margin-left: 10px; margin-right: 10px;
</rendition>
 </model>
 <model predicate="ancestor::p or ancestor::cell"
  behaviour="inline(.)" useSourceRendition="true">
  <rendition scope="before">content: '‘';</rendition>
  <rendition scope="after">content: '’';</rendition>
 </model>
 <model behaviour="block(.)"
  useSourceRendition="true">
  <rendition>margin-left: 10px; margin-right: 10px;
</rendition>
 </model>
</elementSpec>
```

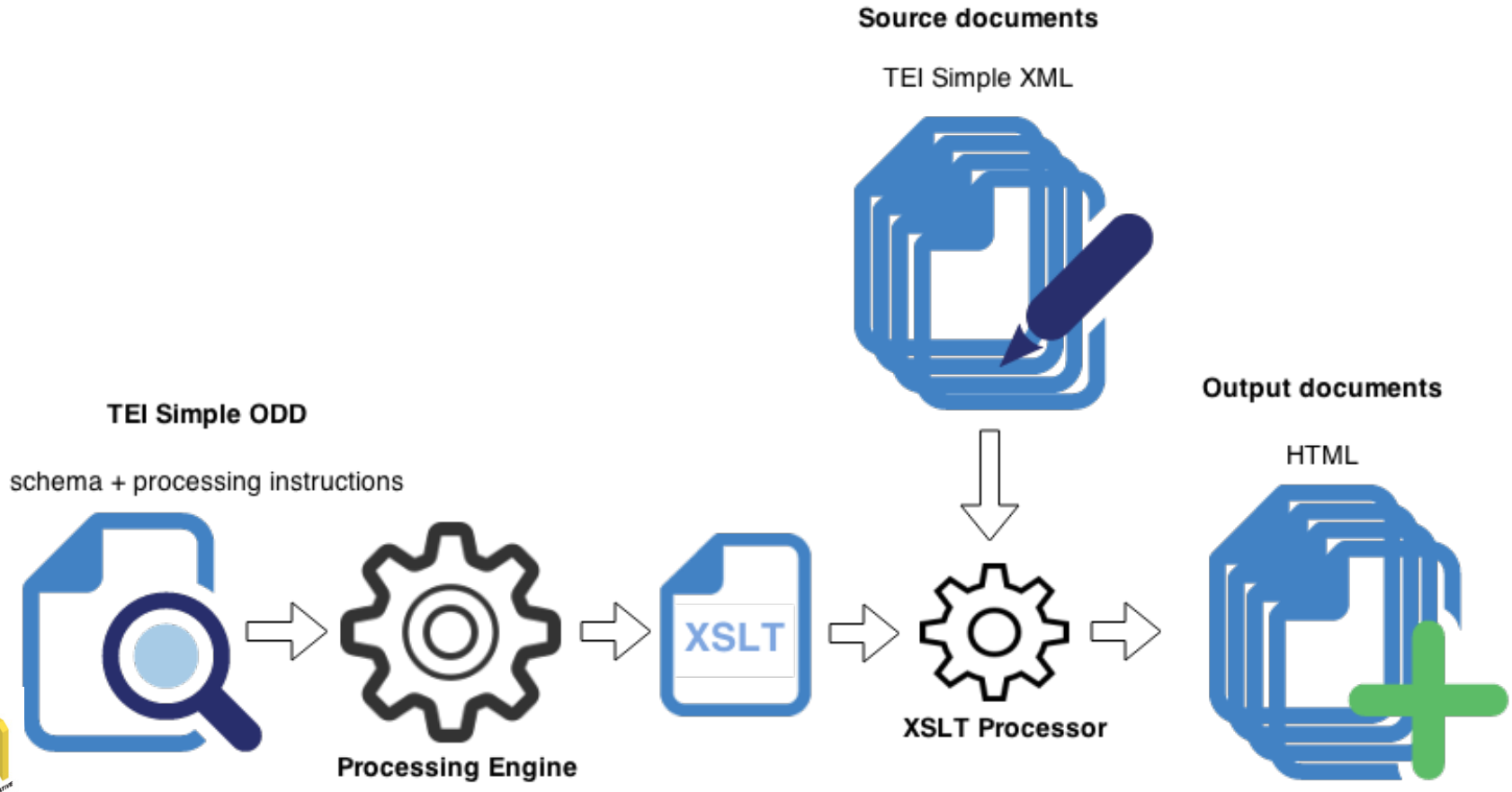
TEI Simple function library

Function name	Description	Probable HTML con-tainer
alternate	Creates a specialized display of alternating elements for displaying the preferred version, both at once or toggling between the two.	
anchor	create anchor with ID	
block	Creates a block out of the content parameter.	div
break	make a line, column, or page break according to *type*	
cell	Creates a table cell	td
cit	shows the content, with an attribute of the source	
document	start a new output document	
figure	make a figure with the *title* as caption	div
glyph	show a character by looking up reference	
graphic	If *@url* is present, uses it to display graphic, else display placeholder image.	img
heading	Creates a heading.	span
index	generate list according to *type*	
inline	Creates inline element out of `content` if there's something in <rendition>, use that formatting otherwise just show text of selected content.	span
link	create hyperlink	a
list	Create a list by following *content*	ol, ul, or

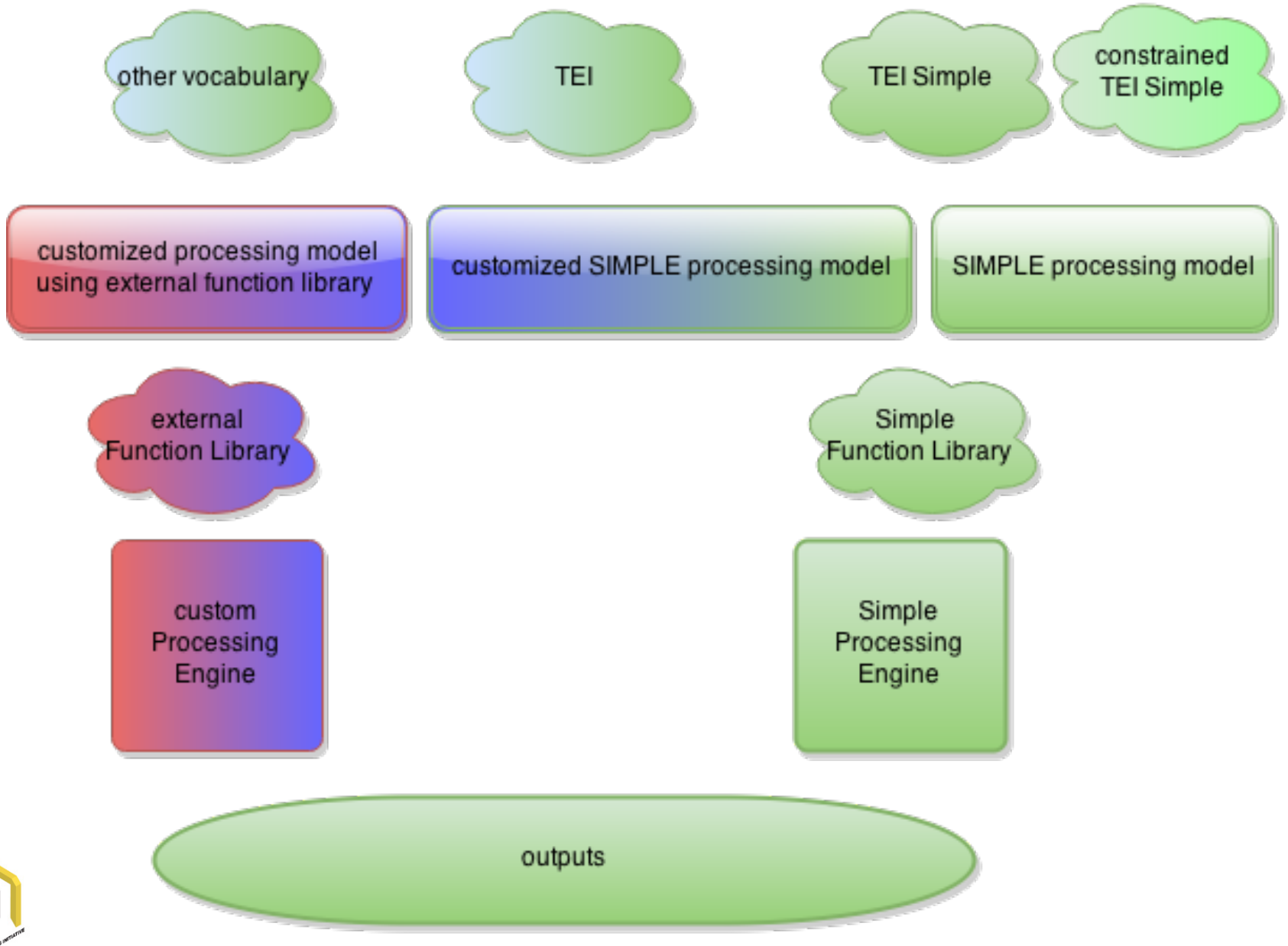
listItem	create list item	li
metadata	create metadata section	head
note	Creates a note block out of `content`, as per value of `place`	span or div
omit	Do nothing, do not process children	
paragraph	Creates a paragraph out of `content`.	p
row	make table row	tr
section		
table	make table	table
title	make document title	

The Processing Engine implementation: PE

As a proof of concept an implementation of PE and Simple function library has been created in XSLT. It consists of XSLT stylesheets suite that generate transform stylesheets based on processing models from TEI Simple ODD.



TEI Simple universe and beyond



The Performance Indicator

The aim is to be able to show for a given text

- Whether it claims to conform to the TEI Simple view of the world
- How it matches up against various encoding criteria
- Which directions it has gone in

The idea is that one can browse a corpus of texts and quickly filter on the basis of encoding (cf `http://www.ota.ox.ac.uk/tcp/`)

Formal editorial decisions

Each of these elements has attributes (as well as prose) for providing quantifiable information

Within <encodingDesc>/<editorialDecl>:

- <correction >
- <hyphenation >
- <normalization >
- <punctuation >
- <quotation >

Extensions we propose on <editorialDecl>

- add "sampled" to values for @status on <correction>
- add "standoff" to values for @method on <correction>
- add @quality to <correction>, with a datatype of *data.probability* (cf Martin's analysis of EEBO)
- add @methodology to <editorialDecl> (eg `<editorialDecl methodology="teisimple">`)

Typical categorizations

Named entities, from ‘Edward was born in London. From an early age, little Teddie loved Buckingham Palace gardens’:

- named entities recognized (London, Buckingham)
- phrases identified (Buckingham Palace)
- identifying unique entities (correlating Edward and Teddie)
- linked to external authority (King Edward VIII)

Date from ‘dated Mons, 3rd August’:

- marked up as date
- normalized date

Citation identification (Biblical texts)

POS tagging:

- word recognition
- sentence recognition
- tagging with POS
- lemmatization

Marking up categorizations

In the header:

```
<category xml:id="named">
 <catDesc>personal name recognition</catDesc>
 <category>
  <category xml:id="named-1">
   <catDesc>identified names</catDesc>
  </category>
  <category xml:id="named-2"/>
  <catDesc>joined together names to form unique
individuals</catDesc>
  <category xml:id="named-3">
   <catDesc>linked people to external sources</catDesc>
   <category xml:id="named-3-1">
    <catDesc>mostly done but didn't bother with
servants</catDesc>
   </category>
  </category>
 </category>
</category>
```

In <editorialDecl>:

```
interpretation ana="#named-3-1"/>
```

Another example of PI, from DTA

```
<extent>
 <measure type="images">537</measure>
 <measure type="tokens">88183</measure>
 <measure type="types">5053</measure>
 <measure type="characters">516483</measure>
</extent>
```

Mapping to RDF (CRM and FRBRoo)

Planned to write X3ML mapping file and test transformation on EEBO texts

(cf `https://github.com/delving/x3ml` and `http://139.91.183.3/3M/`)

Integration with TEI Council

Scheduled for May? meeting of Council, at which James Cummings will present the PM and PI markup additions, and the Simple customization

Practical problems to be solved still

- Finalize syntax of PM
- Decide on a reliable route to PDF
- Decide how to demonstrate or implement the PI
- Work out the RDF mapping

Upcoming activities

- SR creating oXygen framework which references schemas and HTML transforms using XSLT PE
- WM delivers eXist-based system with PE
- MB delivers command-line Java-based PE
- Northwestern demo edition testing design/programmer differentiation
- MT organizes a Simple Hackathon in May/June to test PM 1.0
- (if needed, create PM 2.0, and ask WM, MT and MB to redo PE implementations)
- MM writes documentation/tutorial of the Simple Weltanschauung based on markup of real texts (referencing facsimiles)
- James Cummings presents TEI Simple at DH2015
- Final presentation of Simple to TEI members meeting at Lyon