TEI BY EXAMPLE

MODULE 4: POETRY

Edward Vanhoutte

Ron Van den Branden

Melissa Terras

Centre for Scholarly Editing and Document Studies (CTB) , Royal Academy of Dutch Language and Literature, Belgium, Gent, 9 July 2010

Last updated September 2020

TABLE OF CONTENTS

1. Introduction

According to the Oxford English Dictionary, poetry is:

> Composition in verse or some comparable patterned arrangement of language in which the expression of feelings and ideas is given intensity by the use of distinctive style and rhythm; the art of such a composition. (*Oxford English Dictionary*, "poetry," 2a)

or

> The product of this art as a form of literature; the writings of a poet or poets; poems collectively or generally. (*Oxford English Dictionary*, "poetry," 2b)

A poem then can have different manifestations:

- A piece of writing or an oral composition, often characterised by a metrical structure, in which the expression of feelings, ideas, etc., is typically given intensity or flavour by distinctive diction, rhythm, imagery
  (*Oxford English Dictionary*, "poem," 1)
- A composition in prose having elements in common with a poem.
  (*Oxford English Dictionary*, "poem," 2a)
- An artwork or piece of music having elements in common with a poem.
  (*Oxford English Dictionary*, "poem," 2b)

The length of a poem can vary from a couple of words to a multi-volume book.

If length is not a decisive feature of the poetry genre, verse or some comparable patterned arrangement of language, metrical structure, and metaphorical language certainly are.

The use of metaphorical language is for some critics an argument in favour of the treatment of prose poetry as a form of poetry, while for other critics the absence of a patterned arrangement of language and metrical structure is an argument in favour of the treatment of prose poetry as a form of prose. Likewise, Elizabethan and Jacobean drama, which was often written in verse by authors such as William Shakespeare, Christopher Marlowe, Thomas Middleton, and Ben Jonson, can be considered drama or poetry just as the chorus in a Greek play that was originally composed to be sung.

The TEI Guidelines don't interfere with this poetry/prose debate by treating texts by their document types rather than by their genres. That's why the TEI Guidelines present a chapter on verse, not on poetry. The document type is—contrary to what it may seem—not an innate property of the document, but a formalised analysis of a document by a subjective agent such as a scholar, editor, or encoder who identifies a text as prose, verse, or drama through their own subjective reasoning. For the detailed documentation of the possible encodings of the thus identified text, the encoder or scholar may then turn to the appropriate sections of the TEI Guidelines.

In this module on poetry, we take a step back and consider poetry by example. We will analyse different instances of poetry structurally and encode them for their defining features such as:

- patterned arrangement of language
- rhyme
- metrical structure
- metaphorical language

2. Patterned Arrangement of Language

2.1 Document Type

A shopping list is commonly not considered a form of literature, not even when it features in a literary work, or when it shows some incidental metrical structure or rhyme, unless someone identifies the text as such. The following text can thus be considered as non-literary prose:

```
 Poppadom Oatmeal
Bubble gum
Cut of veal
Mince for pie
Frozen peas
Video for Guy
Selection of teas
Paper towels/garbage bags
Pasta sauce and Parmesan
Pumpkin seed and olive oil
Cheesy crisps and favourite mags
Kidney beans (1 large can)
Cling film and kitchen foil
```

Example 1. A verse-like text.

What we see here is a functional block of fourteen lines of prose, typographically separated from each other by line breaks. The fundamental organisational unit for prose is the paragraph which is encoded in TEI as <p>. The start of a new typographical line is encoded with the milestone tag <lb>, which appears as <lb/> because it is a self-closing empty element. Applied to the shopping list, this produces the following encoding:

```
<p xmlns="http://www.tei-c.org/ns/1.0">Poppadom<lb/>Oatmeal<lb/>Bubble gum<lb/>Cut of
 veal<lb/>Mince for pie<lb/>Frozen peas<lb/>Video for Guy<lb/>Selection of teas<lb/
>Paper towels/garbage bags<lb/>Pasta sauce and Parmesan<lb/>Pumpkin seed and olive
 oil<lb/>Cheesy crisps and favourite mags<lb/>Kidney beans (1 large can)<lb/>Cling film
 and kitchen foil<lb/></p>
```

Example 2. A verse-like text, encoded as prose with line breaks (<lb>).

But our shopping list is a specific type of prose: it is a list, for which we could use <list> containing twelve items for which we use <item>. This could result in the following encoding:

```
<list xmlns="http://www.tei-c.org/ns/1.0">
  <item>Poppadom</item>
  <item>Oatmeal</item>
  <item>Bubble gum</item>
  <item>Cut of veal</item>
  <item>Mince for pie</item>
  <item>Frozen peas</item>
  <item>Video for Guy</item>
  <item>Selection of teas</item>
  <item>Paper towels/garbage bags</item>
  <item>Pasta sauce and Parmesan</item>
  <item>Pumpkin seed and olive oil</item>
  <item>Cheesy crisps and favourite mags</item>
  <item>Kidney beans (1 large can)</item>
  <item>Cling film and kitchen foil</item>
</list>
```

Example 3. A verse-like text, encoded as a list.

On the most basic structural level, this shopping list is thus a named grouping of lines of text. When we use the element <lg> for line group, and the element <l> for line, we arrive at the following encoding:

```
<lg xmlns="http://www.tei-c.org/ns/1.0">
  <l>Poppadom</l>
  <l>Oatmeal</l>
  <l>Bubble gum</l>
  <l>Cut of veal</l>
  <l>Mince for pie</l>
  <l>Frozen peas</l>
  <l>Video for Guy</l>
  <l>Selection of teas</l>
  <l>Paper towels/garbage bags</l>
  <l>Pasta sauce and Parmesan</l>
  <l>Pumpkin seed and olive oil</l>
  <l>Cheesy crisps and favourite mags</l>
  <l>Kidney beans (1 large can)</l>
  <l>Cling film and kitchen foil</l>
</lg>
```

Example 4. A verse-like text encoded with <l>.

SUMMARY

Up to now we have looked at three different possible encodings for the same document which we identified as a shopping list. The first encoding considers the document as a block of prose in which different lines of text are separated by line breaks. The second one structures the shopping list with the appropriate labels (list and item). The third one makes abstraction of the meaning of these typical labels and considers the shopping list as a group of lines of text. The latter is less specific than the second, but more structurally descriptive than the first.

2.2 Structural Divisions

But is this shopping list just a block of prose, organised as a group of lines? The metrical composition, form, and structure of this shopping list, together with the use of rhyme may suggest that this is rather a piece of poetry which can be encoded as such. The rhyme present in the poem gives away a possible structure which consists not of one but of four line groups or stanzas. We use <lg> to encode them:

```
<lg xmlns="http://www.tei-c.org/ns/1.0">
  <lg>
    <l>Poppadom</l>
    <l>Oatmeal</l>
    <l>Bubble gum</l>
    <l>Cut of veal</l>
  </lg>
  <lg>
    <l>Mince for pie</l>
    <l>Frozen peas</l>
    <l>Video for Guy</l>
    <l>Selection of teas</l>
  </lg>
  <lg>
    <l>Paper towels/garbage bags</l>
    <l>Pasta sauce and Parmesan</l>
    <l>Pumpkin seed and olive oil</l>
  </lg>
  <lg>
    <l>Cheesy crisps and favourite mags</l>
    <l>Kidney beans (1 large can)</l>
    <l>Cling film and kitchen foil</l>
  </lg>
</lg>
```

Example 5. Distinction of line groups in a verse-like text, with `<lg>`.

There is nothing in this encoding, however, which documents whether these four line groups belong together or not. Therefore we can wrap another <lg> element around them inside which they nest comfortably.[2]

In order to distinguish among the nesting line groups and the parental one, we can add some semantic information in a @type attribute which specifies a name conventionally used for this level of division and label the line groups respectively as "stanza" and "poem" as in the following example:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <lg type="stanza">
    <l>Poppadom</l>
    <l>Oatmeal</l>
    <l>Bubble gum</l>
    <l>Cut of veal</l>
  </lg>
  <lg type="stanza">
    <l>Mince for pie</l>
    <l>Frozen peas</l>
    <l>Video for Guy</l>
    <l>Selection of teas</l>
  </lg>
  <lg type="stanza">
    <l>Paper towels/garbage bags</l>
    <l>Pasta sauce and Parmesan</l>
    <l>Pumpkin seed and olive oil</l>
  </lg>
  <lg type="stanza">
    <l>Cheesy crisps and favourite mags</l>
    <l>Kidney beans (1 large can)</l>
    <l>Cling film and kitchen foil</l>
  </lg>
</lg>
```

Example 6. Distinguishing nesting line groups with nesting <lg> elements.

.....

2 Like other text-division elements, <lg> elements can nest hierarchically.

Notice that, while the @type attribute can have any value defined by the encoder (as long as it does not contain white space), it is intended solely for conventional names of different classes of text blocks. When <lg> is used to encode paragraphs in prose poetry, the @type attribute value could be "para" or anything else. If the <lg> represents an arbitrary organisation of lines, the @type attribute value could be "free" or anything else. If, for instance, this poem would have been organised differently, say in two quatrains and one sestet, we could have the following encoding:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <lg type="quatrain">
    <l>Poppadom</l>
    <l>Oatmeal</l>
    <l>Bubble gum</l>
    <l>Cut of veal</l>
  </lg>
  <lg type="quatrain">
    <l>Mince for pie</l>
    <l>Frozen peas</l>
    <l>Video for Guy</l>
    <l>Selection of teas</l>
  </lg>
  <lg type="sestet">
    <l>Paper towels/garbage bags</l>
    <l>Pasta sauce and Parmesan</l>
    <l>Pumpkin seed and olive oil</l>
    <l>Cheesy crisps and favourite mags</l>
    <l>Kidney beans (1 large can)</l>
    <l>Cling film and kitchen foil</l>
  </lg>
</lg>
```

Example 7. An alternative grouping of lines, distinguished with the `@type` attribute on `<lg>`.

Next, we can number the stanzas and lines in our poem inside an @n attribute and document that this poem has four stanzas and fourteen lines of verse:[3]

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <lg type="stanza" n="1">
    <l n="1">Poppadom</l>
    <l n="2">Oatmeal</l>
    <l n="3">Bubble gum</l>
    <l n="4">Cut of veal</l>
  </lg>
  <lg type="stanza" n="2">
    <l n="5">Mince for pie</l>
    <l n="6">Frozen peas</l>
    <l n="7">Video for Guy</l>
    <l n="8">Selection of teas</l>
  </lg>
  <lg type="stanza" n="3">
    <l n="9">Paper towels/garbage bags</l>
    <l n="10">Pasta sauce and Parmesan</l>
    <l n="11">Pumpkin seed and olive oil</l>
  </lg>
  <lg type="stanza" n="4">
    <l n="12">Cheesy crisps and favourite mags</l>
    <l n="13">Kidney beans (1 large can)</l>
    <l n="14">Cling film and kitchen foil</l>
  </lg>
</lg>
```

Example 8. Numbering of structural units in a verse text, with @n.

Poems often carry a title which can be encoded using <head>:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <head>Shopping</head>
```

.....

3 For large corpora of verse texts these numberings are commonly added automatically by some sort of programmed routine.

```
  <lg type="stanza" n="1">
    <l n="1">Poppadom</l>
    <l n="2">Oatmeal</l>
    <l n="3">Bubble gum</l>
    <l n="4">Cut of veal</l>
  </lg>
  <lg type="stanza" n="2">
    <l n="5">Mince for pie</l>
    <l n="6">Frozen peas</l>
    <l n="7">Video for Guy</l>
    <l n="8">Selection of teas</l>
  </lg>
  <lg type="stanza" n="3">
    <l n="9">Paper towels/garbage bags</l>
    <l n="10">Pasta sauce and Parmesan</l>
    <l n="11">Pumpkin seed and olive oil</l>
  </lg>
  <lg type="stanza" n="4">
    <l n="12">Cheesy crisps and favourite mags</l>
    <l n="13">Kidney beans (1 large can)</l>
    <l n="14">Cling film and kitchen foil</l>
  </lg>
</lg>
```

Example 9. Encoding the title of a poem in <head>.

Preceding quotations introducing the poem as a motto can be encoded using <epigraph>:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <head>Shopping</head>
  <note type="attribution">To my sweetest son</note>
  <epigraph>
    <cit>
      <quote>Thou shalt not steal</quote>
      <bibl>Ex. 20:15</bibl>
    </cit>
  </epigraph>
```

```
  <lg type="stanza">
    <l>Poppadom</l>
    <l>Oatmeal</l>
    <l>Bubble gum</l>
    <l>Cut of veal</l>
  </lg>
  <lg type="stanza">
    <l>Mince for pie</l>
    <l>Frozen peas</l>
    <l>Video for Guy</l>
    <l>Selection of teas</l>
  </lg>
  <lg type="stanza">
    <l>Paper towels/garbage bags</l>
    <l>Pasta sauce and Parmesan</l>
    <l>Pumpkin seed and olive oil</l>
  </lg>
  <lg type="stanza">
    <l>Cheesy crisps and favourite mags</l>
    <l>Kidney beans (1 large can)</l>
    <l>Cling film and kitchen foil</l>
  </lg>
</lg>
```

Example 10. Encoding introductory quotations with `<epigraph>`.

Poems are often signed, which can be encoded using <signed> outside the `<lg type="poem">` element:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <head>Shopping</head>
  <lg type="stanza">
    <l>Poppadom</l>
    <l>Oatmeal</l>
    <l>Bubble gum</l>
    <l>Cut of veal</l>
  </lg>
  <lg type="stanza">
```

```
    <l>Mince for pie</l>
    <l>Frozen peas</l>
    <l>Video for Guy</l>
    <l>Selection of teas</l>
  </lg>
  <lg type="stanza">
    <l>Paper towels/garbage bags</l>
    <l>Pasta sauce and Parmesan</l>
    <l>Pumpkin seed and olive oil</l>
  </lg>
  <lg type="stanza">
    <l>Cheesy crisps and favourite mags</l>
    <l>Kidney beans (1 large can)</l>
    <l>Cling film and kitchen foil</l>
  </lg>
</lg>
<signed xmlns="http://www.tei-c.org/ns/1.0">M. Ystery-Shopper</signed>
```

Example 11. Encoding the signature for a poem with `<signed>`.

SUMMARY

Different document types such as prose, verse, or drama can be considered poetry. Poetry of the document type "verse" consists typically of minimally one group of one or more lines, which may be encoded using <lg> and <l>, respectively. Line groups can nest inside each other. To all instances of line groups and lines, semantic and analytical information can be added in attributes. The title of a poem can be encoded using <head>, motto's can be encoded using <epigraph>, the signature of the poet can be encoded using <signed>.

2.3 Grouping Structures

Single poems may appear isolated as independent texts, as part of some other document types such as prose and drama, or in combination with other poems as part of composite texts. Typical examples of such composite texts are anthologies, cycles of poems, and composite poems, i.e., poems consisting of other poems. The line

between cycles of poems and composite poems, however, is thin and assigning either interpretation to the texts is the encoder's decision, who can usually depend on how the author or publisher represented the texts in the original publication.

Many encoding strategies can be used to encode either of the composite text types mentioned. In the following paragraphs we suggest only a couple of them.

2.3.1 Composite Poems

Since <lg> elements can nest, it is possible to encode nesting poems using <lg> with a value "poem" for the @type attribute. If we consider the stanza's of the example poem as poems in their own right, which were numbered by the author, this can result in the following encoding:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <head>Shopping</head>
  <lg type="poem" n="1">
    <head>I.</head>
    <l>Poppadom</l>
    <l>Oatmeal</l>
    <l>Bubble gum</l>
    <l>Cut of veal</l>
  </lg>
  <lg type="poem" n="2">
    <head>II.</head>
    <l>Mince for pie</l>
    <l>Frozen peas</l>
    <l>Video for Guy</l>
    <l>Selection of teas</l>
  </lg>
  <lg type="poem" n="3">
    <head>III.</head>
    <l>Paper towels/garbage bags</l>
    <l>Pasta sauce and Parmesan</l>
    <l>Pumpkin seed and olive oil</l>
  </lg>
  <lg type="poem" n="4">
    <head>IV.</head>
```

```
    <l>Cheesy crisps and favourite mags</l>
    <l>Kidney beans (1 large can)</l>
    <l>Cling film and kitchen foil</l>
  </lg>
</lg>
```

Example 12. Encoding a composite poem.

2.3.2 Cycles of Poems

Cycles of poems are structurally akin to composite poems but the top level element is not a `<lg type="poem">` but a with a suggested value `"cycle"` for the @type attribute. If we reconsider the example poem, this generates the next encoding:

```
<div xmlns="http://www.tei-c.org/ns/1.0" type="cycle">
  <head>Shopping</head>
  <lg type="poem" n="1">
    <head>I.</head>
    <l>Poppadom</l>
    <l>Oatmeal</l>
    <l>Bubble gum</l>
    <l>Cut of veal</l>
  </lg>
  <lg type="poem" n="2">
    <head>II.</head>
    <l>Mince for pie</l>
    <l>Frozen peas</l>
    <l>Video for Guy</l>
    <l>Selection of teas</l>
  </lg>
  <lg type="poem" n="3">
    <head>III.</head>
    <l>Paper towels/garbage bags</l>
    <l>Pasta sauce and Parmesan</l>
    <l>Pumpkin seed and olive oil</l>
  </lg>
  <lg type="poem" n="4">
```

```
    <head>IV.</head>
    <l>Cheesy crisps and favourite mags</l>
    <l>Kidney beans (1 large can)</l>
    <l>Cling film and kitchen foil</l>
  </lg>
</div>
```

Example 13. Encoding a cycle of poems.

2.3.3 Anthologies

An anthology can as well be represented using a element but it is good practice to use the more powerful <group> element. The <group> element groups together a sequence of distinct texts (or groups of such texts) which are regarded as a unit for some purpose. The <group> element consists of <text> elements which may have optional <front> and <back> elements and mandatory <body> elements. If we reconsider the example poem that way, the following encoding may apply:[5]

```
<text xmlns="http://www.tei-c.org/ns/1.0">
  <front>
    <docTitle>
      <titlePart>Shopping</titlePart>
    </docTitle>
  </front>
  <group>
    <text>
      <body>
        <lg type="poem" n="1">
          <head>I.</head>
          <l>Poppadom</l>
          <l>Oatmeal</l>
          <l>Bubble gum</l>
          <l>Cut of veal</l>
```

.....

5 Each <text> element can have its own <front> and <back> elements so that it becomes possible, for instance, to encode complete works of poets, maintaining the original front and back materials of the separately published poetry volumes. See Module 1: Common Structure, Elements, and Attributes, section 2.2.

```
      </lg>
    </body>
  </text>
  <text>
    <body>
      <lg type="poem" n="2">
        <head>II.</head>
        <l>Mince for pie</l>
        <l>Frozen peas</l>
        <l>Video for Guy</l>
        <l>Selection of teas</l>
      </lg>
    </body>
  </text>
  <text>
    <body>
      <lg type="poem" n="3">
        <head>III.</head>
        <l>Paper towels/garbage bags</l>
        <l>Pasta sauce and Parmesan</l>
        <l>Pumpkin seed and olive oil</l>
      </lg>
    </body>
  </text>
  <text>
    <body>
      <lg type="poem" n="4">
        <head>IV.</head>
        <l>Cheesy crisps and favourite mags</l>
        <l>Kidney beans (1 large can)</l>
        <l>Cling film and kitchen foil</l>
      </lg>
    </body>
  </text>
 </group>
</text>
```

Example 14. Encoding an anthology with `<group>`.

SUMMARY

Composite poems, cycles of poems, and anthologies are composite texts which can be encoded using different encoding strategies. Composite poems may be encoded using nesting <lg> elements. Cycles of poems may be encoded using sibling <lg type="poem"> elements. Anthologies may be encoded using the <group> element which may have multiple <text> elements.

3. Rhyme

So far, all of the elements and attributes for structuring verse texts were members of the "common" elements and attributes defined in the **core** TEI module. Yet, TEI provides a specific **verse** module as well, which defines a number of elements and attributes specific for the encoding of verse texts. Some of these will be discussed in the following sections. In order to use them, a TEI schema must include all (or just the required) components of that **verse** module; see Module 8: Customising TEI, ODD, Roma for a tutorial on how to customise TEI.

3.1 Rhyming Words

The rhyming words of a line of verse can be encoded using the appropriate <rhyme> element:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <lg type="stanza">
    <l>
      <rhyme>Poppadom</rhyme>
    </l>
    <l>
      <rhyme>Oatmeal</rhyme>
    </l>
    <l>Bubble <rhyme>gum</rhyme></l>
    <l>Cut of <rhyme>veal</rhyme></l>
  </lg>
  <lg type="stanza">
    <l>Mince for <rhyme>pie</rhyme></l>
```

```
    <l>Frozen <rhyme>peas</rhyme></l>
    <l>Video for <rhyme>Guy</rhyme></l>
    <l>Selection of <rhyme>teas</rhyme></l>
  </lg>
  <lg type="stanza">
    <l>Paper towels/garbage <rhyme>bags</rhyme></l>
    <l>Pasta sauce and <rhyme>Parmesan</rhyme></l>
    <l>Pumpkin seed and olive <rhyme>oil</rhyme></l>
  </lg>
  <lg type="stanza">
    <l>Cheesy crisps and favourite <rhyme>mags</rhyme></l>
    <l>Kidney beans (1 large <rhyme>can</rhyme>)</l>
    <l>Cling film and kitchen <rhyme>foil</rhyme></l>
  </lg>
</lg>
```

Example 15. Encoding rhyme words with `<rhyme>`.

<rhyme> can appear anywhere in the line. This way, not only end-of-line rhymes can be tagged but also internal rhyme, even inside prose(-like) paragraphs like in the following fragment:

```
<p xmlns="http://www.tei-c.org/ns/1.0"><rhyme>This</rhyme> <rhyme>course</rhyme>
 on verse <rhyme>is</rhyme> <rhyme>terse</rhyme> and provides a <rhyme>fine</rhyme>
 de<rhyme>sign</rhyme> for the study of poetry like <rhyme>yours</rhyme>
 and <rhyme>mine</rhyme></p>
```

Example 16. `<rhyme>` can be used inside prose paragraphs, too.

3.2 Rhyme Patterns

Rhyme patterns can be documented with a @rhyme attribute which has a default notation in which distinct letters stand for rhyming lines. This attribute can be added to <lg> and/or to <l>, and also to any element that is used for the encoding of poetry. Of course, the

The rhyme scheme in the shopping list poem is *ababcdcdefgefg*. This can be documented inside the @rhyme attribute of <lg type="poem">. The rhyme scheme of the separate stanzas can be encoded inside the @rhyme attribute of <lg type="stanza">, and even the rhyme scheme of the separate lines can technically be encoded inside the @rhyme attribute of the <l> element. The complexity of the use of all these options depends on the encoder. A maximally complex encoding could be the following:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem" rhyme="ababcdcdefgefg">
  <lg type="stanza">
    <l rhyme="a">
      <rhyme>Poppadom</rhyme>
    </l>
    <l rhyme="b">
      <rhyme>Oatmeal</rhyme>
    </l>
    <l rhyme="a">Bubble <rhyme>gum</rhyme></l>
    <l rhyme="b">Cut of <rhyme>veal</rhyme></l>
  </lg>
  <lg type="stanza">
    <l rhyme="c">Mince for <rhyme>pie</rhyme></l>
    <l rhyme="d">Frozen <rhyme>peas</rhyme></l>
    <l rhyme="c">Video for <rhyme>Guy</rhyme></l>
    <l rhyme="d">Selection of <rhyme>teas</rhyme></l>
  </lg>
  <lg type="stanza">
    <l rhyme="e">Paper towels/garbage <rhyme>bags</rhyme></l>
    <l rhyme="f">Pasta sauce and <rhyme>Parmesan</rhyme></l>
    <l rhyme="g">Pumpkin seed and olive <rhyme>oil</rhyme></l>
  </lg>
  <lg type="stanza">
    <l rhyme="e">Cheesy crisps and favourite <rhyme>mags</rhyme></l>
    <l rhyme="f">Kidney beans (1 large <rhyme>can</rhyme>)</l>
    <l rhyme="g">Cling film and kitchen <rhyme>foil</rhyme></l>
  </lg>
</lg>
```

Example 17. Encoding the rhyme scheme of different structural units, with `@rhyme`.

3.3 Rhyming Words and Patterns

The correspondence between the rhyming pattern documented in the @rhyme attribute and the rhyming words encoded with the <rhyme> element can be specified in a @label attribute on the <rhyme> element. The value of this attribute is usually one of the letters of the rhyme pattern. Applied to the shopping list poem, this results in the following encoding:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem" rhyme="ababcdcdefgefg">
  <lg type="stanza">
    <l>
      <rhyme label="a">Poppadom</rhyme>
    </l>
    <l>
      <rhyme label="b">Oatmeal</rhyme>
    </l>
    <l>Bubble <rhyme label="a">gum</rhyme></l>
    <l>Cut of <rhyme label="b">veal</rhyme></l>
  </lg>
  <lg type="stanza">
    <l>Mince for <rhyme label="c">pie</rhyme></l>
    <l>Frozen <rhyme label="d">peas</rhyme></l>
    <l>Video for <rhyme label="c">Guy</rhyme></l>
    <l>Selection of <rhyme label="d">teas</rhyme></l>
  </lg>
  <lg type="stanza">
    <l>Paper towels/garbage <rhyme label="e">bags</rhyme></l>
    <l>Pasta sauce and <rhyme label="f">Parmesan</rhyme></l>
    <l>Pumpkin seed and olive <rhyme label="g">oil</rhyme></l>
  </lg>
  <lg type="stanza">
    <l>Cheesy crisps and favourite <rhyme label="e">mags</rhyme></l>
    <l>Kidney beans (1 large <rhyme label="f">can</rhyme>)</l>
    <l>Cling film and kitchen <rhyme label="g">foil</rhyme></l>
  </lg>
</lg>
```

Example 18. Identifying a <rhyme> in a rhyme scheme with @label.

All <rhyme> elements with the same value for their @label attribute are assumed to rhyme with each other within a given scope. That scope is defined by the nearest ancestor element for which the @rhyme attribute has been supplied.

In the following encoding of the same poem, the scope is defined by the nearest ancestor element with a @rhyme attribute, i.e., the `<lg type="stanza">` element. This means that the rhyming words labelled `a`, `b`, or `c` are only assumed to rhyme inside that stanza and not across stanzas:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <lg type="stanza" rhyme="abab">
    <l>
      <rhyme label="a">Poppadom</rhyme>
    </l>
    <l>
      <rhyme label="b">Oatmeal</rhyme>
    </l>
    <l>Bubble <rhyme label="a">gum</rhyme></l>
    <l>Cut of <rhyme label="b">veal</rhyme></l>
  </lg>
  <lg type="stanza" rhyme="abab">
    <l>Mince for <rhyme label="a">pie</rhyme></l>
    <l>Frozen <rhyme label="b">peas</rhyme></l>
    <l>Video for <rhyme label="a">Guy</rhyme></l>
    <l>Selection of <rhyme label="b">teas</rhyme></l>
  </lg>
  <lg type="stanza" rhyme="abc">
    <l>Paper towels/garbage <rhyme label="a">bags</rhyme></l>
    <l>Pasta sauce and <rhyme label="b">Parmesan</rhyme></l>
    <l>Pumpkin seed and olive <rhyme label="c">oil</rhyme></l>
  </lg>
  <lg type="stanza" rhyme="abc">
    <l>Cheesy crisps and favourite <rhyme label="a">mags</rhyme></l>
    <l>Kidney beans (1 large <rhyme label="b">can</rhyme>)</l>
    <l>Cling film and kitchen <rhyme label="c">foil</rhyme></l>
  </lg>
</lg>
```

Example 19. The nearest ancestor element with a `@rhyme` attribute determines the scope for `@label`.

SUMMARY

The occurrence of rhyming words and rhyming patterns and their correspondence can be encoded by a combination of tags and attribute values. Depending on the encoder's preferences, they can be applied to different structural levels of the text.

4. Metrical Structure

There is a difference between the conventional metrical structure of a poem and the actual realisation of that conventional metrical structure. Both can be documented with the use of attributes.

4.1 Conventional Metrical Structure

The conventional metrical structure in which a poet is working can be documented with the `@met` attribute, whose value specifies the metrical form of a single verse line. This attribute can be added to any text-division element in use in the encoding of poetry: , <lg>, and <l>. The metre documented inside the `@met` attribute is inherited by any metrical unit contained within the element for which the `@met` attribute has been supplied. This means that the `@met` value of a or an <lg>, for instance, is inherited by nesting s or <lg>s. The metrical structure of that <lg> is understood to contain as many repetitions of the pattern as there are lines in the line group. The same attribute value, when inherited in turn by the <l> element, must be understood not to repeat. Consider the third stanza of our shopping list poem, which exists of three lines of alternating long and short syllables:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="stanza" met="-u|-u|-u|-/">
  <l>Paper towels/garbage bags</l>
  <l>Pasta sauce and Parmesan</l>
  <l>Pumpkin seed and olive oil</l>
</lg>
```

Example 20. Encoding the metrical structure of a stanza.

4.2 Private Metrical Notation Scheme

The encoder is free to design their own metrical notation scheme. In the example above use has been made of the classical scansion system which marks quantitative metre originally by a macron (here a dash `-`) for long syllables and a breve (here a `u`) for short syllables. A bar `|` is used to mark the foot boundary and a slash `/` marks the line boundary. Other systems like the ictus (`/`) and `x (x)` system could be used to denote metrically stressed and unstressed syllables. The use of this latter system results in the following `@met` attribute value for the stanza above: `"/+/+/+/"`.

The metrical notation scheme used may be documented in the <metDecl> element within the <encodingDecl> element inside the TEI header. For the above example this could be documented as follows:

```
<metDecl xmlns="http://www.tei-c.org/ns/1.0">
  <metSym value="-">long syllable</metSym>
  <metSym value="u">short syllable</metSym>
  <metSym value="|">foot boundary</metSym>
  <metSym value="/">metrical line boundary</metSym>
</metDecl>
```

Example 21. Formal documentation of the metrical notation scheme with `<metSym>` elements inside `<metDecl>`.

This may also be expressed less formally, for instance:

```
<metDecl xmlns="http://www.tei-c.org/ns/1.0">
  <p>use has been made of the classical scansion system which marks quantitative metre
   originally by a macron (here a dash '-') for long syllables and a breve (here a 'u')
   for short syllables. A bar '|' is used to mark the foot boundary and a slash '/'
   marks the line boundary.</p>
</metDecl>
```

Example 22. Informal documentation of the metrical notation scheme as a loose description inside `<metDecl>`.

REFERENCE

See section 6.5 Metrical Notation Declaration of the TEI Guidelines for more examples and complex cases.

4.3 Realisation of Metrical Structure

When we consider the third and the fourth stanzas of the shopping list poem, we see that some lines of the fourth stanza divert from the metrical scheme documented in the `@met` attribute. This deviation, or otherwise put, the "realisation," may be documented inside a `@real` attribute:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="stanza" met="-u|-u|-u|-/">
  <l>Paper towels/garbage bags</l>
  <l>Pasta sauce and Parmesan</l>
  <l>Pumpkin seed and olive oil</l>
</lg>
<lg xmlns="http://www.tei-c.org/ns/1.0" type="stanza" met="-u|-u|-u|-/">
  <l>Cheesy crisps and favourite mags</l>
  <l real="-u|-'|-u|-">Kidney beans (1 large can)</l>
  <l real="- -|u-|u-">Cling film and kitchen foil</l>
</lg>
```

Example 23. Encoding the actual realisation of a metrical scheme with a `@real` attribute.

4.4 Caesura and Enjambements

Whereas a caesura expresses a metrical pause inside one line of verse, an enjambement marks the breaking of a syntactic unit between two lines of verse.

The third line of the last stanza of the shopping list poem consists of two equal metrical parts (`-u-`) with a rest in between. This rest is called a caesura, and is here represented with an apostrophe. We can encode this information also in the verse line, with the <caesura> element which marks the point at which a metrical line may be divided:

```
<l xmlns="http://www.tei-c.org/ns/1.0">Kidney beans <caesura/> (1 large can)</l>
```

Example 24. Encoding caesura with `<caesura>`.

The presence, absence, or degrees of discrepancy between lines of verse and syntactic units running over them can be documented as values of an optional `@enjamb` attribute on the `<l>` element. When used, this attribute can, for instance, have the value `"yes"` or `"no"` to signal its occurrence, or `"weak"` or `"strong"` to express some sort of evaluation. In the following stanza of Edgar Allen Poe's "The Raven" the presence or absence of enjambement is encoded:

```
<lg xmlns="http://www.tei-c.org/ns/1.0">
  <l enjamb="yes">But the Raven, sitting lonely on the placid bust, spoke only</l>
  <l enjamb="no">That one word, as if his soul in that one word he did outpour.</l>
  <l enjamb="no">Nothing further then he uttered, not a feather then he fluttered,</l>
  <l enjamb="no">Till I scarcely more than muttered,—"Other friends have flown
   before;</l>
  <l enjamb="no">On the morrow he will leave me, as my Hopes have flown before."</l>
  <l enjamb="no">Then the bird said, "Nevermore."</l>
</lg>
```

Example 25. Encoding enjambments with `@enjamb`.

SUMMARY

The expressions of conventional metrical structures in `@met` attributes are inherited by the children of the attributed elements, except where deviations are expressed in a `@real` attribute. Each encoder is free to design their own encoding scheme and document it inside `<metDecl>`. It is also possible to encode the occurrence and location of a caesura (with `<caesura>`) or an enjambement (with `@enjamb`).

5. Metaphorical Language

Just like any other aspect of the transcription and encoding process, analysing metaphorical language involves interpretation. Unlike the expression of this interpretation in structural markup, the analysis of metaphorical language addresses logical structures that probably will cross structural boundaries. Due to the primary focus of the TEI encoding scheme on representing structural characteristics of texts, and the syntactic requirement that all XML structures should nest properly, this kind of logical analysis will need a kind of workaround. One such workaround defines different analytic categories separately and links them to the relevant passages in the poems. A first requirement is that the smallest addressable structures be identified with an @xml:id attribute:[10]

```
<lg xmlns="http://www.tei-c.org/ns/1.0" xml:id="p001" type="poem">
  <lg xml:id="s001" type="stanza">
    <l xml:id="l001">Poppadom</l>
    <l xml:id="l002">Oatmeal</l>
    <l xml:id="l003">Bubble gum</l>
    <l xml:id="l004">Cut of veal</l>
  </lg>
  <lg xml:id="s002" type="stanza">
    <l xml:id="l005">Mince for pie</l>
    <l xml:id="l006">Frozen peas</l>
    <l xml:id="l007">Video for Guy</l>
    <l xml:id="l008">Selection of teas</l>
  </lg>
  <lg xml:id="s003" type="stanza">
    <l xml:id="l009">Paper towels/garbage bags</l>
    <l xml:id="l010">Pasta sauce and Parmesan</l>
    <l xml:id="l011">Pumpkin seed and olive oil</l>
  </lg>
  <lg xml:id="s004" type="stanza">
    <l xml:id="l012">Cheesy crisps and favourite mags</l>
    <l xml:id="l013">Kidney beans (1 large can)</l>
    <l xml:id="l014">Cling film and kitchen foil</l>
```

.....

10 @xml:id provides a unique identifier for the element bearing the attribute. Its value must start with a letter, or _ and may be followed by one or more Named Characters (letter, digit, ., -, _, Unicode Combining Character, or Extender). See the XML Specification for more guidance on the use of @xml:id.

```
  </lg>
</lg>
```

Example 26. Making structural units addressable via `@xml:id`.

These @xml:id values can later be used to identify spans of interpretive categories. Such interpretive spans are encoded as , identifying the structural scope of this interpretation with the @from attribute marking its beginning and the @to attribute marking its end. Their values are values for an @xml:id attribute elsewhere in the document, prefixed with a hash character (#), in order to indicate it as the identifier part of a formal *URI* reference. Preferably, the @resp attribute is used to identify who is responsible for the interpretation.[11] Related elements can be grouped in a <spanGrp> element. A @type attribute can characterise the kind of analysis. In our poem, one way of identifying all images concerned with food and non-food could be:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" xml:id="p001" type="poem">
  <lg xml:id="s001" type="stanza">
    <l xml:id="l001">Poppadom</l>
    <l xml:id="l002">Oatmeal</l>
    <l xml:id="l003">Bubble gum</l>
    <l xml:id="l004">Cut of veal</l>
  </lg>
  <lg xml:id="s002" type="stanza">
    <l xml:id="l005">Mince for pie</l>
    <l xml:id="l006">Frozen peas</l>
    <l xml:id="l007">Video for Guy</l>
    <l xml:id="l008">Selection of teas</l>
  </lg>
  <lg xml:id="s003" type="stanza">
    <l xml:id="l009">Paper towels/garbage bags</l>
    <l xml:id="l010">Pasta sauce and Parmesan</l>
    <l xml:id="l011">Pumpkin seed and olive oil</l>
  </lg>
  <lg xml:id="s004" type="stanza">
```

.....

11 The value of the @resp attribute must be a pointer to an element in the document header that is associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing, or encoding. See chapter 14 Certainty and Responsibility of the TEI Guidelines.

```
    <l xml:id="l012">Cheesy crisps and favourite mags</l>
    <l xml:id="l013">Kidney beans (1 large can)</l>
    <l xml:id="l014">Cling film and kitchen foil</l>
  </lg>
</lg>
<spanGrp xmlns="http://www.tei-c.org/ns/1.0" resp="#RvdB" type="imagery">
  <span from="#l001" to="#l006">food</span>
  <span from="#l007">non-food</span>
  <span from="#l008">food</span>
  <span from="#l009">non-food</span>
  <span from="#l010" to="#l013">food</span>
  <span from="#l014">non-food</span>
</spanGrp>
```

Example 27. Identifying interpretive spans in a text with `<span>`.

Notice that the @from attribute is mandatory on . The @to attribute is optional; when it is missing, the entire structure identified by the @from attribute will be taken as the context for the interpretation.

As will be clear from this example, this system of linking analyses to textual structures is not very economic when discontinuous structures share the same analysis. Another approach for logical analysis works from the opposite angle, by hooking text structures to specific interpretations. In order to do so, interpretive categories should be formally defined somewhere else, either in the same document or externally. This is done inside the <interp> element, bearing a unique @xml:id attribute. Related interpretive categories can be grouped inside an <interpGrp> element. Following example could identify the same semantic categories from the previous example:

```
<interpGrp xmlns="http://www.tei-c.org/ns/1.0" resp="#RvdB" type="imagery">
  <interp xml:id="food">food</interp>
  <interp xml:id="non-food">non-food</interp>
</interpGrp>
```

Example 28. Defining interpretive categories in `<interp>` elements.

Inside the transcription of the poem, reference can be made to these interpretations with an @ana attribute. Its value should always point to an identifier, either locally or externally. Suppose these interpretive categories are stored in a separate document named **analysis.xml**. Then the poem could be analysed as follows: [12]

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <lg ana="analysis.xml#food" type="stanza">
    <l>Poppadom</l>
    <l>Oatmeal</l>
    <l>Bubble gum</l>
    <l>Cut of veal</l>
  </lg>
  <lg xml:id="s002" type="stanza">
    <l ana="analysis.xml#food">Mince for pie</l>
    <l ana="analysis.xml#food">Frozen peas</l>
    <l ana="analysis.xml#non-food">Video for Guy</l>
    <l ana="analysis.xml#food">Selection of teas</l>
  </lg>
  <lg xml:id="s003" type="stanza">
    <l ana="analysis.xml#non-food">Paper towels/garbage bags</l>
    <l ana="analysis.xml#food">Pasta sauce and Parmesan</l>
    <l ana="analysis.xml#food">Pumpkin seed and olive oil</l>
  </lg>
  <lg xml:id="s004" type="stanza">
    <l ana="analysis.xml#food">Cheesy crisps and favourite mags</l>
    <l ana="analysis.xml#food">Kidney beans (1 large can)</l>
    <l ana="analysis.xml#non-food">Cling film and kitchen foil</l>
  </lg>
</lg>
```

.....

12 Of course, analyses can be much more complex. The TEI provides a generic mechanism for expressing complex hierarchies of interpretive relationships, called "Feature Structures." See chapter 12 Feature Structures of the TEI Guidelines for a detailed treatment.

Example 29. Linking text structures with interpretations with the `@ana` attribute.

SUMMARY

For the study of metaphorical language in poetry, logical structures should be identified and grouped into interpretive categories which can be related to other such categories. These interpretive categories with their logical relationships can be stored inside or outside the XML document containing the analysed text. In the poem, reference can be made to these internally or externally documented analyses.

6. Advanced Encoding

6.1 Components of the Verse Line

It is often convenient for various kinds of analysis to encode further subdivisions of verse lines. This can be done using the <seg> element which contains any arbitrary phrase-level unit of text (including other <seg> elements).

The third and the fourth stanzas of our poem, for instance, contain lines which include two items each, except for line 13. The appropriate encoding could be the following:

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="stanza" n="3">
  <l><seg type="item">Paper towels</seg> / <seg type="item">garbage bags</seg></l>
  <l><seg type="item">Pasta sauce</seg> and <seg type="item">Parmesan</seg></l>
  <l><seg type="item">Pumpkin seed</seg> and <seg type="item">olive oil</seg></l>
</lg>
<lg xmlns="http://www.tei-c.org/ns/1.0" type="stanza" n="4">
  <l><seg type="item">Cheesy crisps</seg> and <seg type="item">favourite mags</seg></l>
  <l>
    <seg type="item">Kidney beans (1 large can)</seg>
  </l>
  <l><seg type="item">Cling film</seg> and <seg type="item">kitchen foil</seg></l>
</lg>
```

Example 30. Identifying low-level structures with `<seg>`.

6.2 Dedications

If the poem is encoded as an <lg>, the TEI doesn't have an "out of the box" solution to deal with dedications which appear under the title of the poem. In the case a dedication needs to be encoded, there are two viable options. The first one is to consider the dedication as a <head> and use its @type attribute for specifying its function as a "dedication":

```
<lg xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <head>Shopping</head>
  <head type="dedication">To my sweetest son</head>
  <lg type="stanza" n="1">
    <l n="1">Poppadom</l>
    <l n="2">Oatmeal</l>
    <l n="3">Bubble gum</l>
    <l n="4">Cut of veal</l>
  </lg>
  <lg type="stanza" n="2">
    <l n="5">Mince for pie</l>
    <l n="6">Frozen peas</l>
    <l n="7">Video for Guy</l>
    <l n="8">Selection of teas</l>
  </lg>
  <lg type="stanza" n="3">
    <l n="9">Paper towels/garbage bags</l>
    <l n="10">Pasta sauce and Parmesan</l>
    <l n="11">Pumpkin seed and olive oil</l>
  </lg>
  <lg type="stanza" n="4">
    <l n="12">Cheesy crisps and favourite mags</l>
    <l n="13">Kidney beans (1 large can)</l>
    <l n="14">Cling film and kitchen foil</l>
  </lg>
</lg>
```

Example 31. Encoding a dedication following a heading as a special <head>.

The second option is to encode the poem as a `<div type="poem">` instead of an `<lg type="poem">`, and consider the title as an anonymous block <ab> with a value of `"dedication"` for its @type attribute.

```
<div xmlns="http://www.tei-c.org/ns/1.0" type="poem">
  <head>Shopping</head>
  <ab type="dedication">To my sweetest son</ab>
  <lg type="stanza" n="1">
    <l n="1">Poppadom</l>
    <l n="2">Oatmeal</l>
    <l n="3">Bubble gum</l>
    <l n="4">Cut of veal</l>
  </lg>
  <lg type="stanza" n="2">
    <l n="5">Mince for pie</l>
    <l n="6">Frozen peas</l>
    <l n="7">Video for Guy</l>
    <l n="8">Selection of teas</l>
  </lg>
  <lg type="stanza" n="3">
    <l n="9">Paper towels/garbage bags</l>
    <l n="10">Pasta sauce and Parmesan</l>
    <l n="11">Pumpkin seed and olive oil</l>
  </lg>
  <lg type="stanza" n="4">
    <l n="12">Cheesy crisps and favourite mags</l>
    <l n="13">Kidney beans (1 large can)</l>
    <l n="14">Cling film and kitchen foil</l>
  </lg>
</div>
```

Example 32. Encoding a dedication after a heading as `<ab>` inside `<div>`.

6.3 Acrostics

One example of the use of the <seg> element for further advanced analysis is the following proposal to signal the presence of an acrostic in a poem. In this poem by Edgar Allan Poe, entitled "An Acrostic," the first letters of each line of verse together form the first word of the first line: "Elizabeth."

```
<lg xmlns="http://www.tei-c.org/ns/1.0">
  <l><seg type="acros" rend="acros('ELIZABETH', 1)">E</seg>lizabeth it is in vain you
   say</l>
  <l>"<seg type="acros" rend="acros('ELIZABETH', 2)">L</seg>ove not" - thou sayest it
   in so sweet a way:</l>
  <l><seg type="acros" rend="acros('ELIZABETH', 3)">I</seg>n vain those words from thee
   or L.E.L.</l>
  <l><seg type="acros" rend="aros('ELIZABETH', 4)">Z</seg>antippe's talents had
   enforced so well:</l>
  <l><seg type="acros" rend="acros('ELIZABETH', 5)">A</seg>h! if that language from thy
   heart arise,</l>
  <l><seg type="acros" rend="acros('ELIZABETH', 6)">B</seg>reath it less gently forth -
   and veil thine eyes.</l>
  <l><seg type="acros" rend="acros('ELIZABETH', 7)">E</seg>ndymion, recollect, when
   Luna tried</l>
  <l><seg type="acros" rend="acros('ELIZABETH', 8)">T</seg>o cure his love - was cured
   of all beside -</l>
  <l><seg type="acros" rend="acros('ELIZABETH', 9)">H</seg>is follie - pride - and
   passion - for he died.</l>
</lg>
```

Example 33. A possible encoding of an acrostic.

7. What's Next?

You have reached the end of this tutorial module covering poetry markup with TEI. You can now either

- proceed with other TEI by Example modules
- have a look at the examples section for the poetry module.

- take an interactive test. This comes in the form of a set of multiple choice questions, each providing a number of possible answers. Throughout the quiz, your score is recorded and feedback is offered about right *and* wrong choices. Can you score 100%? Test it here!