Proposal to add <ruby>, <rb>, and <rt> to encode ruby glosses for Japanese texts
ルビ符号化要素（<ruby>, <rb>, <rt>）の追加申請

East Asian / Japanese SIG, the TEI Consortium
(Kazuhiro Okada, Satoru Nakamura, and Kiyonori Nagasaki)
In cooperation with So Miyagawa, Yifan Wang, Nobutake Kamiya, Naoki Kokaze, and Martin Holmes

Drafted Feb 28, 2020
The first edition Aug, 20, 2020

Summary

The ruby gloss, or *rubi,* or *furigana,* is a device for giving guidance on phonation, as well as presenting another reading for the text, which sometimes can be an essential part of the text. The structure of ruby glosses has, until recently, been established between characters and glosses, rather than words and glosses, since over a millennium ago. Such a textual structure requires another layer of semantics besides the existing set of vocabulary, such as <seg>, <note> (or <add>) and . In its centuries of history, ruby glosses can also be double-sided. Reflecting on the current encoding schema in HTML5, this proposal discusses encoding examples attested in real texts, taking Taiwanese Bopomofo into consideration.

ルビ（振り仮名）は、発音の手がかりを与える手段である一方、あらたな読解をテクストに与えることも多く、むしろ、それが本体であることすらあった。ルビの構造は、ながらく本行の文字と振られる文字の関係を本位とし、単語本位ではなかった。このテクストの構造は、既存の語彙、たとえば<seg>、<note>（あるいは<add>）との組み合わせのような意味論では捉えられず、あらたな把握を要する。また長い歴史の中では、ルビはテクストの両側に付されることもあった。既存のHTMLでの符号化スキーマを参考にしつつ、この提案では、実例をもとに符号化の例を検討し、台湾の注音符号に及ぶ。

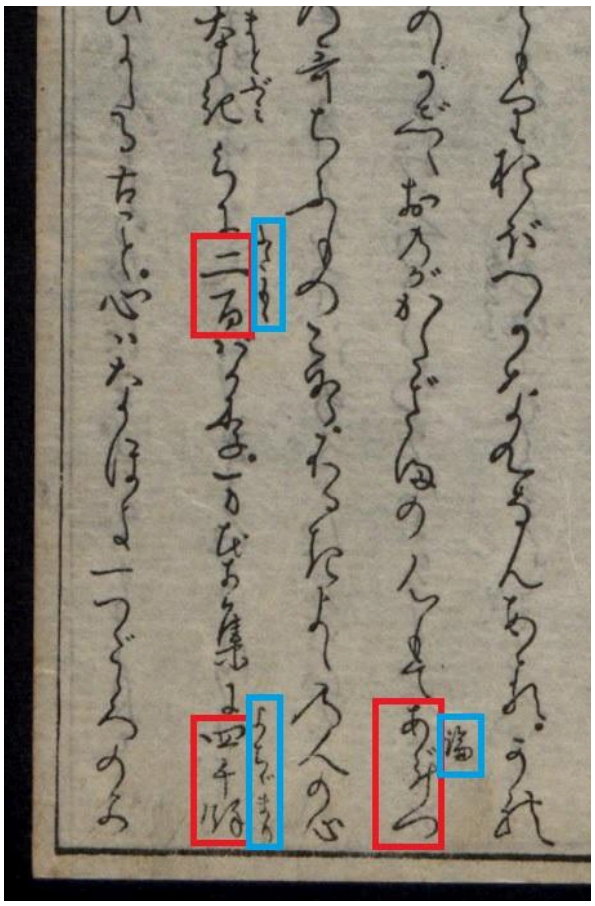
Introduction

Encoding of the ruby glossed text is a crucial requirement for the Japanese community, because the TEI Guidelines have not had effective encoding methods in any schemas despite its importance in texts. Since this problem pertains to most, if not all, Japanese texts, if this problem is not addressed in the TEI P5 Guidelines themselves, the Japanese community will be forced to address this on an individual basis, leading to possible multiple incompatible solutions. This is the main reason we here propose a solution below.

Japanese texts use both syllabaries and a morpho-syllabary, called kanji. Kanji can represent both native Japanese and borrowed Chinese vocabularies, without a clear indication of phonation. Generally, this will not be a problem because Japanese readers have been trained — or context can provide enough information for them to deduce phonation. However, there are many cases of ambiguity, unfamiliar characters, or layers of readings, and in those cases, ruby glosses are useful to the reader.

Ruby glosses are thus supposed to reconcile the native and Sino-Japanese vocabularies with their scripts. They were originally used as interlinear glosses to Classical and authoritative Chinese texts, and finally became a part of Japanese texts. This means that Ruby is used to provide not merely explanation but also readings both in phonetic and textual aspects.

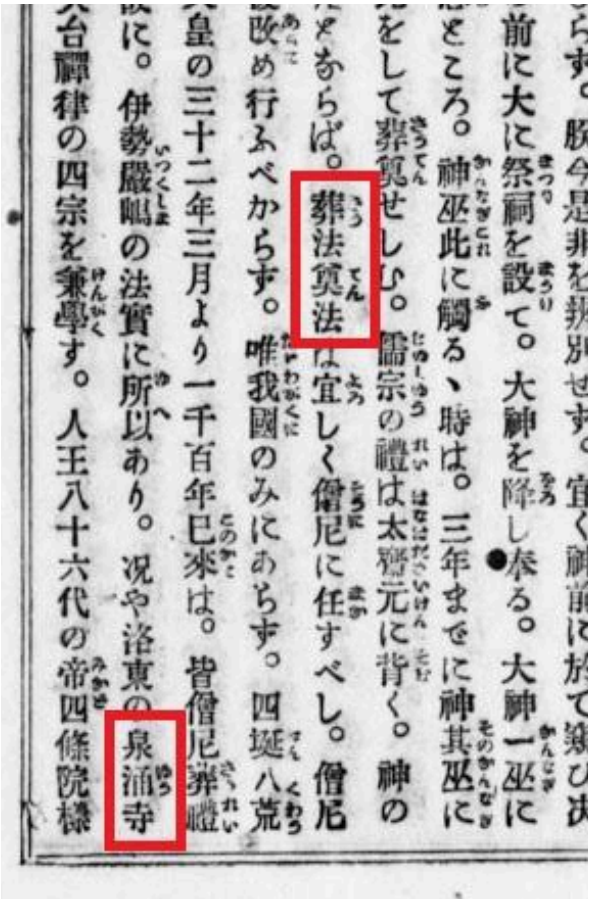
The ruby text (RT) is attached to a particular portion of the base text (RB), in order to provide the reading of that portion. The figure below shows an old woodblock printing text by Kamo-no Mabuchi, published in 1760. This example shows RTs in both syllabic and kanji.



- RB: あげつらふ (ageturō); RT: 論 (ron) (l. 3)
- RB: 二百 (ni-hyaku); RT: ふたもも (futamomo) (l. 5)
- RB: 四千余 (yon-sen-yo); RT: よちぢまり (yo-tiji-mari) (l. 5)

Figure 1: Kamo no Mabuchi, Man'yoshū Taikō, from Staatsbibliothek zu Berlin, Libri japon. 262/264

As ruby is a kind of interlinear gloss, why does it present an encoding issue? Since interlinear glosses are not peculiar to Japanese tradition, isn't it enough just to encode it with <add> or <note> elements?



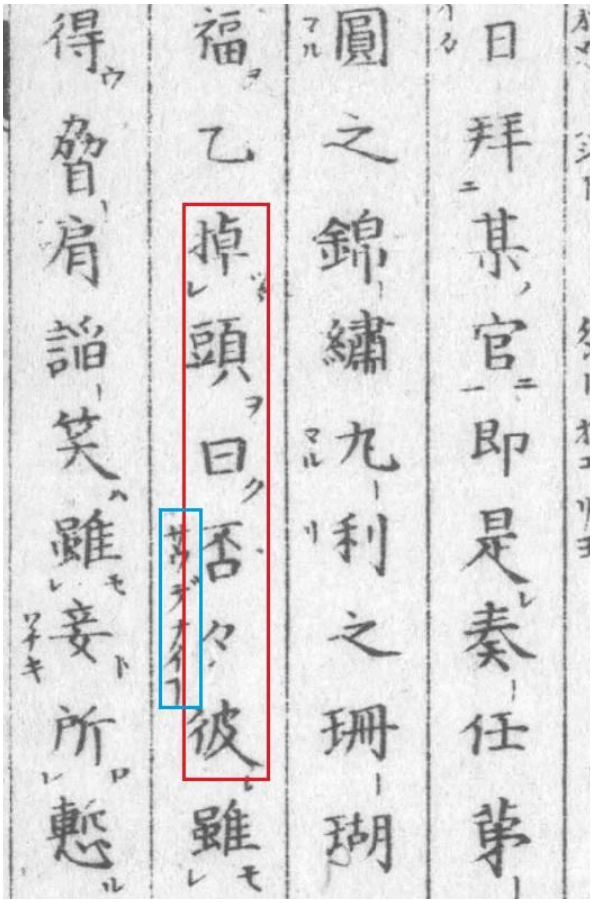
- RB: 葬[法]奠[法] ([ ] = no ruby portion); RT: さう, てん
- RB: [泉]涌[寺]; RT: ゆう

Figure 2: Rentai, *Tsūzoku Kōseki shū*, from National Diet Library, 188.54-R18t (http://dl.ndl.go.jp/info:ndljp/pid/993061/9)

The ruby here differs from interlinear glosses. Structurally, it is attached to the base word or even a particular character in the targeted word. It makes two parallel texts and an addendum is not sufficient for this use. The ruby can be attached to an arbitrary sequence of text, meaning that it can be used on a character basis. For example, in Figure 2, only part of each word has ruby texts. In this case, some existing TEI elements can represent the relationship between the base characters and the ruby texts by use of specific use of attributes or prose texts. However, it is naturally treated for researchers and practitioners using Japanese texts.

The parallelism of the ruby text and the main text can bring literary effects, as in Figure 3. Although the text is written in Classical Chinese, a Colloquial Japanese reading of the Classical Chinese phrase is echoed by the means of ruby as a phonetic indicator. Shimizu

(2019) classifies such emergent rubies in terms of “dual textuality.” As such, ruby can be not merely a later addition or a simple note, but an essential part of a text.



- RB: [乙頭ヲ掉ツテ曰ク]否々; RT: サウデナイ事
  (Transposed into the Japanese word order)

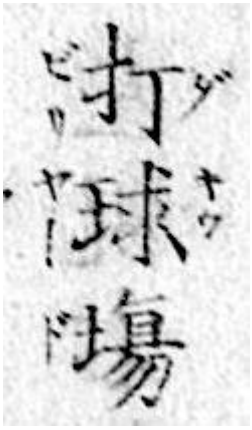
Figure 3: Narushima Ryūhoku, Ryūhoku shinshi, pt. 2, from NIJL, 87-80-2

It should be kept in mind that the ruby is not present solely in the Japanese writing system. As Figure 4 shows, the ruby is also used in Chinese texts. We thus need to be conscious that the ruby encoding schema should not be biased only towards Japanese usages when it is formulated.

Figure 4: Direction sign in a Taiwanese elementary school

Then how should we encode it?

In the following example, we will examine encoding strategies in existing schemas. The example to be considered is the following double-sided ruby. Its base reads *dakyūjo*, or "a billiard hall," which has ruby texts on both sides. The ruby on the right-hand side reads *dakyu*, ordinary Sino-Japanese reading of the first two characters of the base. The ruby on the left-hand side reads *biriyado*, a Japanese rendition of the English word "billiard."



The ruby encoding has been codified in numerous schemas, including W3C HTML in either XHTML or HTML5 (HTML Living Standard). In-house XML schemas have been devised by some institutions. Here, we will review W3C's latest attempt.

(see: https://developer.mozilla.org/en-US/docs/Web/HTML/Element/ruby)

The W3C's 2013 markup schema in the HTML5 context presents a deep consideration into the use of ruby in Japanese. The schema proposes two strategies for the example in question. One is nesting two ruby elements, and the other is using a container element, named <rtc>.

Of course, we can't simply port the HTML5's encoding schema to TEI, but we see it contains some problems in structure. First, their conception of ruby is heavily influenced by layout issues, which is no more than a typographic custom. This is understandable in the light of the W3C's concern with rendering in web browsers, but we should encode things in a more semantic way. As we have shown, the alignment problem is a big issue, but should it be solved just by repetition of <rt>s and <rb>s? Rather, shouldn't we utilize a breaking element? We see the essence of ruby is creating dual textuality. So, it should generally be a pair of RB and RT, as well as encompassing a word-level entities, rather than fragmentally encoded. However, we want to utilize W3C's html ruby, so we need to keep interoperability with them, especially when we have existing resources encoded using this convention.
Our proposal is demonstrated and discussed below according to ruby patterns. Here we use the <anchor> element to show segmentation of the ruby text, and rb that has no ruby text is not tagged by <rb>.

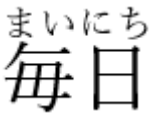
Examples

Pattern A: Simple ruby

Pattern A-1: In horizontal texts

Pattern A-1-1: A simple word
If a ruby of multiple hiragana characters is put on a word consisting of kanji characters in the main text, one should tag the hiragana characters in ruby with <rt> and the kanji characters in the main text with <rb>, and finally tag the entire ruby section with <ruby>. For example,

<ruby><rt>まいにち</rt><rb>毎日</rb></ruby>



In this case, the entire ruby of four hiragana characters corresponds to the word "毎日" which is made from two kanji characters.

Pattern A-1-2: Correspondence with characters or words?

However, if the ruby is added to each kanji character, one tends to tag a single kanji character with <rt> and write hiragana characters in the ruby corresponding to the kanji next to it with <rb> one by one. For example,


```
<ruby>
 <rb>常</rb><rt>じょう</rt>
 <rb>用</rb><rt>よう</rt>
 <rb>漢</rb><rt>かん</rt>
 <rb>字</rb><rt>じ</rt>
 <rb>表</rb><rt>ひょう</rt>
</ruby>
```

However, one should avoid tagging on character-base as above, as in HTML encoding, to prevent breaking textual sequences. Rather, we should use <anchor> to make correspondences between characters in the main text and the ruby text.

```
<ruby>
   <rb>常<anchor xml:id="rb1" />用<anchor xml:id="rb2" />漢<anchor xml:id="rb3" />字<anchor xml:id="rb4" />表<anchor xml:id="rb5" /></rb>
  <rt>じょう<anchor xml:id="rt1" corresp="#rb1" />よう<anchor xml:id="rt2" corresp="#rb2" />かん<anchor xml:id="rt3" corresp="#rb3" />じ<anchor xml:id="rt4" corresp="#rb4" />ひょう<anchor xml:id="rt5" corresp="#rb5" /></rt>
</ruby>
```

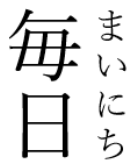
This encoding strategy is good not just for its beauty, but also for not making things harder: one can tag deeper levels of the structure only with this anchoring strategy. The example above can be divided into three morphs (there may be an argument on the morphological

status of these entities, though): 常用, 漢字, and 表, and thus be consistently encoded in terms of linguistics:

```
<ruby>
 <rb>
  <w>
   <m>常<anchor xml:id="rb1"/>用<anchor xml:id="rb2"/></m>
   <m>漢<anchor xml:id="rb3"/>字<anchor xml:id="rb4"/></m>
   <m>表<anchor xml:id="rb5"/></m>
  </w>
 </rb>
 <rt>
  <m>じょう<anchor xml:id="rt1" corresp="#rb1"/>
   よう<anchor xml:id="rt2" corresp="#rb2"/></m>
  <m>かん<anchor xml:id="rt3" corresp="#rb3"/>じ
   <anchor xml:id="rt4" corresp="#rb4"/></m>
  <m>ひょう<anchor xml:id="rt5" corresp="#rb5"/></m>
 </rt>
</ruby>
```

Pattern A-2: In vertical texts

If the characters are written vertically, one can specify it with ... . In this pattern, one can also add the ruby either to one whole word or to each character. We use the following example to demonstrate:



Correspondence to a word

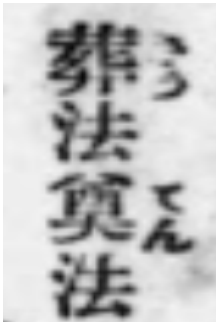
```
<p style="writing-mode: vertical-rl">
 <ruby>
  <rb>毎日</rb>
  <rt>まいにち</rt>
```

```
 </ruby>
</p>
```

Correspondence to each character

```
<p style="writing-mode: vertical-rl">
 <ruby>
  <rb>毎<anchor xml:id="rb1"/>日</rb>
  <rt>まい<anchor xml:id="rt1" corresp="#rb1"/>にち</rt>
 </ruby>
</p>
```

Ruby can also be assigned for kanji characters in the main text discontinuously. For example, in the example below, only the first and third kanji characters contain ruby.




```
<ruby><rb>葬</rb><rt>さう</rt>法</ruby>
<ruby><rb>奠</rb><rt>てん</rt>法</ruby>
```

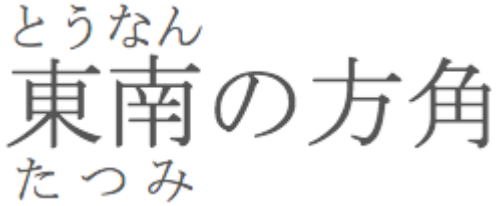
In the example above, the second and fourth kanji characters contain no ruby, but each constitutes a single word with the previous character.

Pattern B: Double-sided

Ruby is not only written on a single side of characters in the main text but also on two sides. Both are encoded by nested ruby elements like <app>. In the double-sided ruby, one should specify @rend value in each occurrence for <ruby>.

Pattern B-1: Simple

We will begin with a simple case. The example below shows that the ruby is written on the top and bottom of two characters in the main text written horizontally. The upper ruby text indicates a usual pronunciation. The bottom ruby does another pronunciation derived from *I Ching* which is a view of the world in ancient Chinese — so it cannot be divided into the character basis. The cluster aims to represent two different renditions of the word.



Here, one can nest the ruby written on the bottom as:

```
<ruby>
 <rb>
  <ruby>
   <rb>東南</rb>
   <rt type="primary" place="above">とうなん</rt>
  </ruby>
 </rb>
 <rt  type="secondary" place="below">たつみ</rt>
</ruby>
```

Pattern B-2: Correspondence with each character

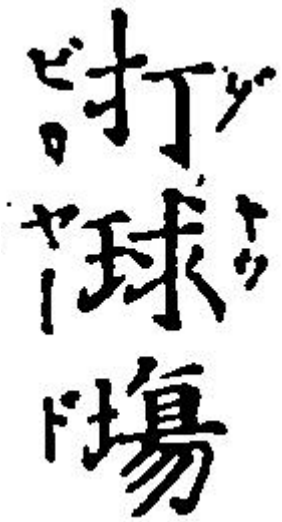
One can also assign each ruby to a specific character in the main text with <anchor/> as shown below:

```
<ruby>
 <rb>
  <ruby>
   <rb>東<anchor xml:id="rb1" />南<anchor xml:id="rb2" /></rb>
        <rt  type="primary"  place="above">とう<anchor  xml:id="rt1"  corresp="#rb1"  />なん
<anchor xml:id="rt2" corresp="#rb2" /></rt>
```

```
  </ruby>
 </rb>
 <rt type="secondary" place="below">たつみ</rt>
</ruby>
```

Pattern B-3: Complex aligning ruby characters with Kanji (base characters)

The last example is a combination of patterns we showed above. Here, the main text is written vertically. The three characters are one word 打球場 which means billiard hall. On the right side of the word, the ruby signifies the Sino-Japanese pronunciation of the first two characters 打球, whilst the ruby on the left side signifies the Japanese rendition of the English word “billiard.” The alignment of characters in ruby and the main text must be solved with <anchor/> tags as below.



5872

```
<ruby>
 <rb>
  <ruby><!-- Or declare the default rendering -->
   <rb><anchor xml:id="rb1"/>打<anchor xml:id="rb2"/>球</rb>
        <rt  place="right"><anchor xml:id="rt1" corresp="#rb1"/>ダ<anchor xml:id="rt2" corresp="#rb2"/>キウ</rt></ruby>
場</rb>
 <rt type="secondary" place="left">ビリヤード</rt>
</ruby>
```

There may be an argument that ruby tags are not needed and can be encoded like the following:

<seg xml:id="rb1">
<seg xml:id="rb2">打球</seg>
場</seg>
<add target="#rb2" place="right">ダキウ</add>
<add target="#rb1" place="left">ビリヤード</add>

or

<ab xml:id="rb1">打<anchor xml:id="rb2"/>球<anchor xml:id="rb3"/>場<anchor xml:id="rb4"/></ab>
ダキウ
ビリヤード

<add> is not enough because it cannot represent the texts are parallel without the aid of <alt> (that is too complex for the heavy use of ruby). is also too general to use for this specific structure which could be benefitted from if the designated elements were used. However, there may also be an argument that we should allow stand-off markup, or that we do not need <ruby> at all. In contrast with <app>, stand-off markup of ruby is structurally impossible in that <rb> must be used in the main text, whereas <app> does not need to be encoded inside the main text. Moreover, we believe that using the enclosing element is natural in parallel with <app> or <choice> schemas as following:

<ruby>
<rb xml:id="rb1">打<anchor xml:id="rb2"/>球<anchor xml:id="rb3"/>場<anchor xml:id="rb4"/></rb>
<rt from="#rb1" to="#rb3">ダキウ</rt>
<rt from="#rb1" to="#rb4">ビリヤード</rt>
</ruby>

cf. HTML5-like solution
<ruby style="ruby-position: over;">

```
 <rb>打</rb><rb>球</rb><rb>場</rb>
 <rt>ダ</rt><rt>キウ</rt>
 <rtc style="ruby-position: under;">ビリヤード</rtc>
</ruby>

<ruby>
<rb xml:id="rb1">打<anchor xml:id="rb2"/>球<anchor xml:id="rb3"/>場<anchor xml:id="rb4"/></rb>
<rt from="#rb1" to="#rb3">ダキウ</rt>
<rt from="#rb1" to="#rb4">ビリヤード</rt></ruby>
```

Pattern C: Ruby in other languages, ex: Bopomofo

Ruby is also utilized for writing systems other than Japanese. For example, Bopomofo / Zhuyin phonetic signs are used for Mandarin Chinese mainly in Taiwan. The assignment of Bopomofo is similar to ruby. Here is an example to render Bopomofo as ruby. Unlike Japanese ruby whose positioning is tied with writing direction, this Taiwanese Bopomofo can also be written on the right side of each character in the main text, even when the main text is written horizontally. Ruby encoding in Bopomofo is already adopted in HTML. Note that as a rule Chinese words consist of single syllables.


```
<ruby><rb>敦</rb><rt place="right">ㄉㄨㄣ</rt></ruby>
<ruby><rb>化</rb><rt place="right">ㄏㄨㄚˋ</rt></ruby>
<ruby><rb>國</rb><rt place="right">ㄍㄨㄛˊ</rt></ruby>
<ruby><rb>小</rb><rt place="right">ㄒㄧㄠˇ</rt></ruby>
```

Conclusion

Each of the examples above can be represented by some combinations of existing elements and attributes. However, users who swim in the tradition of Japanese texts know a method to easily treat them in one way under the concept of ruby. By implementing the ruby into the TEI Guidelines, we believe that the guidelines will embrace not only potential users in the tradition, but also have an opportunity to indicate an attitude about internationalization/globalization to the global researchers and practitioners.

Content model

Encompassing these encoding examples above, we propose the following content model for ruby related elements:

ruby: model.rubyLike (should be in macro.phraseSeq and macro.phraseSeq.limited?)
rb, rt: model.rubyPart

ruby: any: text node, model.rubyPart
rb: text, model.segLike, macro.phraseSeq, model.rubyLike
rt: same as rb except for model.rubyLike

attributes:
ruby, rb, rt: @att.global, @att.typed
macro.rt should allow @from and @to ?

```
<elementSpec ident="ruby" ns="http://www.example.org/ns/tei_ruby" mode="add">
 <desc versionDate="2020-02-28" xml:lang="en">contains a ruby gloss, with a
```

```
   glossed text and a ruby text.</desc>
  <classes>
   <memberOf key="att.global"/>
   <memberOf key="att.typed"/>
   <memberOf key="model.rubyLike"/>
  </classes>
  <content>
   <elementRef key="rb" minOccurs="1" maxOccurs="unbounded"/>
   <elementRef key="rt" minOccurs="1" maxOccurs="unbounded"/>
  </content>
</elementSpec>

<elementSpec ident="rt" ns="http://www.example.org/ns/tei_ruby" mode="add">
  <desc versionDate="2020-02-28" xml:lang="en">contains a ruby text.</desc>
  <classes>
   <memberOf key="att.global"/>
   <memberOf key="att.typed"/>
   <memberOf key="model.rubyPart"/>
  </classes>
  <content>
   <alternate>
    <textNode/>
    <classRef key="model.segLike"/>
    <classRef key="macro.phraseSeq"/>
    <classRef key="model.rubyLike"/>
   </alternate>
  </content>
  <attList>
   <attDef ident="from" usage="opt" mode="add">
    <desc>gives the identifier of the node which is the starting point of
     the span of text being glossed.</desc>
    <datatype>
     <dataRef key="teidata.pointer"/>
    </datatype>
    <constraintSpec scheme="schematron" ident="rt-from">
     <desc>Enforce the presence of the <att>to</att> attribute</desc>
     <constraint>
      <assert test="@to" xmlns="http://purl.oclc.org/dsdl/schematron"
       >The to= attribute of <name
        xmlns="http://purl.oclc.org/dsdl/schematron"/> is
       required.</assert>
     </constraint>
    </constraintSpec>
   </attDef>
   <attDef ident="to" usage="opt" mode="add">
    <desc>gives the identifier of the node which is the end-point of the
     span of text being glossed.</desc>
    <datatype>
     <dataRef key="teidata.pointer"/>
    </datatype>
   </attDef>
  </attList>
</elementSpec>
```

```
<elementSpec ident="rb" ns="http://www.example.org/ns/tei_ruby" mode="add">
 <desc versionDate="2020-02-28" xml:lang="en">contains a ruby glossed text,
  in case of multi-layered ruby gloss, nesting is possible.</desc>
 <classes>
  <memberOf key="att.global"/>
  <memberOf key="att.typed"/>
  <memberOf key="model.rubyPart"/>
 </classes>
</elementSpec>

<classSpec ident="model.rubyLike" type="model" mode="add">
 <classes>
  <memberOf key="macro.phraseSeq"/>
  <memberOf key="macro.phraseSeq.limited"/>
 </classes>
</classSpec>

<classSpec ident="model.rubyPart" type="model" mode="add">
 <classes>
  <memberOf key="att.placement"/>
 </classes>
</classSpec>
```