情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2009-DD-72 No.8
2009/7/30

# 多言語による共同執筆・出力環境に関する一考察

佐々木フェリックス[†]　阿南康宏[††]　山口琢[†††]
小林龍生[††††]

本稿では、W3C のテクニカルノート執筆のために用いられた多言語による共同執筆・出力環境を紹介する。この環境の目的は、新たな技能習得を要することなく執筆過程も含めて、多様な出力形態を保証することにある。本環境は固有の目的のために開発されたものではあるが、類似の目的に資するための一般的考察も行う。

# Design considerations and their implementation for a multilingual, collaborative document editing and publication environment

Felix Sasaki[†]　Yasuhiro Anan[††]　Taku Yamaguchi[†††]
Tatsuo Kobayashi[††††]

This paper introduces an implementation of a multilingual, collaborative document editing and publication environment. The purpose is to provide a low bar for learning new editing techniques, but keeping flexibility for e.g. creating various outputs and using various data sources in the processing chain. The implementation described here is project specific, but we will draw general conclusions hopefully useful for similar efforts.

## 1. Introduction

This paper introduces an implementation of a multilingual, collaborative document editing and publication environment. The main components of this environment are:

- The general system, see section 2.
- Mechanisms used to validate the edited documents, see section 3.
- A schema used for validation of characteristics specific to the multilingual format, see section 4.
- A processing chain for generating various versions to be published, see section 5.

The main feature of this approach towards multilingual editing is that it relies on XHTML as the underlying document format. This makes it easy for editors to use their favorite editing tools. Steps of validation and generation of a version to be published are subsequent to the editing, and the editors do not need to bother with these. This approach can be compared to microformats, which are a means to hide application-specific semantics, in this case the properties of a multilingual document format, within ordinary (X)HTML. A difference between our approach and microformats is that our approach is more sustainable, since it offers not only a format but also a processing chain, and a key part in this chain is the validation of the application-specific semantics. The main focus of the paper is the description of the schema used for validation.

## 2. Overview: The system in general

The system consists of three steps, executed in the following order: a) editing, b) validation, c) postprocessing.

a) The editing step is done by several editors. Each of them modifies XHTML files. No further constraints are applied to these files than the constraints of usual (X)HTML editing environments, that is the constraints of the (X)HTML DTD.

b) The validation step is executed after editing. The schema described in the subsequent section plays a key role in this step. In theory an editor can evoke this step on his own. In practice the system works also well if the validation and subsequent processing is left to other

[†] ポツダム応用科学大学（ドイツ）
The University of Applied Sciences Potsdam, Germany
[††]マイクロソフトディベロップメント
Microsoft Development, Japan
[†††]ジャストシステム
Justsystems Corp.
[††††]ジャストシステムデジタル文化研究所
Justsystems Digital Culture Research Center

project participants. These people then give feedbacks to the editors about errors, which lead to a back-and-forth between editing and validation.

c) The postprocessing step is in a sense a misnomer, since from a document processing perspective it involves most of the effort: merging of (due to sheer size) physically distributed document parts, generation of indexes like table of contents, and integration of external data source. However, from the perspective of the "manual work", that is mainly the editing work, this step is following after everything is finished. Hence we use the term "postprocessing".

## 3. The validation mechanisms: DTDs, RELAX NG and XPath

The system uses three validation mechanisms: XML DTD, RELAX NG and XPath. The XML DTD validation is validation against the XHTML DTD and part of the editing environment used by the individual editors. The RELAX NG validation is part of the step b). It is the main means to verify the application specific semantics, that is the properties of the multilingual document format. The XPath validation is part of the post-processing step c) and checks constraints which are beyond both the multilingual format and the XHTML DTD, like certain co-occurrence constraints between glossary entries and term definitions.
A separation of validation steps in the processing chain has the benefit that different people deal with different problems and separate technologies to solve them. There are the three roles of the editors, the schema designer and the HTML or e.g. PDF "output generator". Aligned with these roles the three validation technologies DTD, RELAX NG and XPath are used within the steps a), b) and c). Readers who are familiar with XML validation might wonder why the co-occurrence constraints are not verified via Schematron. This would have been possible; however, co-occurrence errors are closely bound to processes to be executed during the post-processing step, like generation of indexes or cross-references. Using Schematron for checking co-occurrences would have meant to double the implementation effort: one implementation for constraint checking, another for the processes mentioned above. Hence, "pure" XPath was a good choice to simplify the effort.

## 4. Details of the RELAX NG schema

As mentioned before the RELAX NG schema used in step b) is the main means of the document format described in this paper. In this section we will introduce this schema, separated in two parts: general structure and aspects related to multilinguality.

### (1) General structure of the schema

The schema constrains the element which may appear in an XHTML 1.0 document. It constitutes a super-set of the constraints which are imposed by the XHTML "strict" DTD. Or to put it differently, each document which is valid against the RELAX NG schema is valid against the XHTML "strict" DTD, but not the other way round.
The general structure imposed by the RELAX NG schema is very similar to other document-oriented XML formats like DocBook or the TEI: a document consists of a header with meta information, the document body and an appendix. However, the content of the header and the body is very restricted. The header contains only meta information related to character encoding and language, the title, optionally links to other documents (e.g. the "next" or "previous" section), and style information. An example of the <head> element is given below.

```
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <meta http-equiv="Content-Style-Type" content="text/css" />
  <title>Requirements of Japanese Text Layout (English version)</title>
  <link rel="stylesheet" type="text/css" href="./StyleSheets/base.css" />
  <link rel="contents" href="#en-contents" />
</head>
```

The textual part of the document is even more restricted, see below:

```
<body>
  <div class="body">
    <div class="div1" id="introduction">...</div>
  </div>
  <div class="back">
    <div class="div1" id="app1">...</div>
  </div>
</body>
```

It consists of a "body" which contains "div1" divisions, and a back part. "div1" divisions are structured as follows:

```
<div class="div1">div1heading, div-mix?, div2*</div>
```

The <div> element is identified as a "div1" division via the "class" attribute with the value "div1". "div1heading" is a heading for "div1" division, div-mix is a mixture of block-level elements for paragraphs, figures, lists or tables. "div2" is structured like "div1", the difference being that "div2" contains "div2" headings and optionally "div3".

(2) **Aspects of the schema related to multilinguality**

The main means to express multilinguality are <div> elements with a “class” attribute of the value “Taiyaku”. Taiyaku "対訳" means „aligned translation“, and the purpose of these elements is just this: to align the Japanese and the English part of the document. Below is an example of the alignment for paragraphs.

```
<div class="Taiyaku">
 <p lang="ja" xml:lang="ja">これはインラインのタグも使用できる例文です。</p>
 <p lang="en" xml:lang="en">This is an example sentence which may also contain
inline markup.</p>
</div>
```

The <div class="Taiyaku"> element contains two <p> elements. Each of the element has a “lang” and an “xml:lang” attribute, which have prescribed values: “ja” and “en”. If we would make the semantics hidden in this structure explicit, the following XML fragment might be used:

```
<p-taiyaku>
 <p-ja>これはインラインのタグも使用できる例文です。</p-ja>
 <p-en>This is an example sentence which may also contain inline markup.</p-en>
</p-taiyaku>
```

The following structure is used for figures. It is again an alignment of Japanese and English parts.

```
<div class="Taiyaku">
 <p class="figure" xml:lang="ja" lang="ja" ...>
 <img .../>
 <span class="figureCaption">漢字・平仮名・片仮名</span>
</p>
<p class="figure" xml:lang="en" lang="en" ...>
 <img .../>
 <span class="figureCaption">Kanji, hiragana and katakana.</span>
</p>
</div>
```

Making the semantics hidden in this structure explicit might lead to the following XML fragment:

```
<fig-taiyaku>
 <fig-japanese ...>
  <img .../>
  <caption>漢字・平仮名・片仮名</caption>
 </fig-japanese>
 <fig-english ...>
  <img .../>
  <caption>Kanji, hiragana and katakana.</caption>
 </fig-english>
</fig-taiyaku>
```

The RELAX NG schema contains definitions for several other kinds of aligned units: headings, list items of various types (e.g. ordered versus unordered), or tables. These are the main means to achieve a multilingual, aligned document structure.

(3) **Further aspects of the RELAX NG schema**

The aspects of the RELAX NG schema described above were related to general structures available in many document-oriented XML schemes and the specific aspect of multilingual alignment, realized via <div> elements with the class “taiyaku”. In this section we will introduce – as an example - a feature of the schema which is specific to our project, but not necessarily to general aspects of multilinguality.

The feature concerns the ability to assure the representation of Unicode character names. These names are standardized in the Unicode character data base. For example, the following text, created via the editors in step a) (editing),

```
In JIS X 4051, <span class="character">[、]</span> and <span
class="character">[,]</span>...
```

should be given in the output as follows.

```
In JIS X 4051, <span class="character">IDEOGRAPHIC COMMA "、"</span> and
<span class="character">COMMA ","</span>...
```

This is easy to achieve via XSLT. The <span> element with the attribute class='character' is used to identify the <span> elements properly. The XSLT-stylesheet uses the character

provided between the brackets "[、]" as an input for a query against the Unicode character data base. As a result the name of the character, e.g. "IDEOGRAPHIC COMMA", is retrieved.
Of course such a process is possible without validation against a RELAX NG schema. Nevertheless the schema helps in assuring consistency of the use of class attributes. Without this consistency the XSLT-transformation would be error-prone and finding the errors would be a tedious task.

## 5. Details of the processing chain

The processing chain has been implemented with the ANT build tool. Choosing ANT was arbitrary: Other pipeline mechanisms like MAKE could have been used as well. In the future, however, we might switch to XProc, a pipeline language currently being developed by W3C. XProc is designed for processing of XML data and is itself expressed in an XML syntax. This has the benefit that we can design the information flow, which is after all a flow of XML information, more naturally.
In ANT various so-called tasks have been defined. Some of them are interdependent, that is one task cannot be executed before another one is finished.
- "validateSlices" is the task used to validate the XHTML documents, due to size available in physically separated files, against the RELAX NG schema described above.
- "merge" is used to merge the physically separated files into one file, using XSLT. This task depends on the validation of the files, that is on the execution of "validateSlices".
- "validateMergedDocument" is used to again validate the merged file against the RELAX NG schema. This is necessary since in some cases the result of the "merge" task needs to be corrected.
- "create-en-doc-edcopy" is used to create an editor's copy of the English version of the document. This means that all Japanese content is filtered out of the merged document, indexes are produced, external information e.g. from the Unicode data base is integrated etc. The result of this task is a document for an English audience. This task depends on "validateMergedDocument".
- "create-en-doc-public-draft" is nearly identical to "create-en-doc-edcopy". The difference is only that the result of "create-en-doc-public-draft" is styled as a public draft and not as an editor's copy.
- "create-ja-doc-edcopy" is complementary to "create-en-doc-edcopy". All English content is filtered out of the merged document, indexes are produced with e.g. Japanese numbering of tableOfContents entries, etc. The dependency is also identical to "create-en-doc-edcopy".
- "create-ja-doc-public-draft" is the counterpart to "create-en-doc-public-draft".
- "updateCharNames" is used to integrate information from the Unicode character data base into the document. This task is executed against the merged document.
The modular architecture of this processing chain allows it easily to add more processing steps, or in the terminology of ANT tasks.

## 6. Summary and Conclusions

This paper introduced a multilingual, collaborative document editing and publication environment. The backbone of the environment is an architecture of validation and document transformation, bound together by a series of interdependent processing step, to be applied in a specific order. This architecture relies heavenly on three techniques of XML-validation: XML DTDs, RELAX NG and XPath. RELAX NG is the key means for hiding the application-specific semantics of multilingual document properties below the "common" XHTML DTD.
The conclusion to be drawn from this project is that multilingual editing of documents with a group of participants requires a careful consideration for designing an editing environment. For example: What is the knowledge of the participants, e.g. the editors, about potential editing formats and processing chains? What tools do they have available? What communication is necessary or possible to set up between editors and other contributors, e.g. the "output generator"?
In the area of localization, a common model is to use XLIFF as a means for storing and editing original and to be translated content. However, in the current project the design considerations mentioned above resulted in a different approach. In addition, the huge amount of project specific requirements, e.g. the usage of external data sources and validation of related markup structures, made it feasible to develop a different structure.
For similar efforts we can recommend that all these aspects should taken into account for setting up an architecture for multilingual document editing and publishing.