データベースシステム　100－19
（１９９４．１０．１４）

HyTimeを用いた多言語文書記述

今郷　詔　(株) リコー
土屋　哲　富士通 (株)
小町祐史　松下電送 (株)

電子化文書の最大の利点の一つは、複数の順序で情報をアクセスできることにある。日本語や中国語・韓国語などで記述されたテキストが相互に関連しているような多言語文書は、読者が様々な関連を辿れるようにするためにハイパー文書として記述するのが自然である。
そのような多言語文書を幅広く利用できるようにするには、アプリケーションから独立した記述が必要である。ISO 標準である SGML/HyTime はこの目的に適した特長を持っている。
多言語文書記述に必要な、様々な文字の表現と言語の特定を SGML によって、異言語部分の関連表現を HyTime によって行なう方法を提案する。

Multilingual document descriptions with HyTime

IMAGO Satosi　RICOH Co.,Ltd.
TSUCHIYA Satoshi　FUJITSU Ltd.
KOMACHI Yushi　Matsushita Graphic Communication Systems, Inc.

One of the largest benefit of electronic documents is their feature of information accessibility in more than one order. Multilingual documents contain different language portions, which have close relationship with each other. The traversal between them has to be realized by introducing the hypermedia structure.
To utilize widely such multilingual documents, they need be described independently of application programs. SGML and HyTime, which are ISO standards, have suitable features to this purpose.
The scheme is proposed of describing multilingual documents. Language identification and character representation can be achieved using SGML. Relationships between each language portions can be described using HyTime.

1 Introduction

One of the largest benefit of electronic documents is their feature of information accessibility in more than one order. Multilingual documents contain different language portions, which have close relationship with each other. The traversal between them has to be realized by introducing the hypermedia structure. Those multilingual hypermedia documents could be described in a number of schemes.

Considering the interchange of the hypermedia documents, model and description scheme should be specified by using international standards, SGML[1][2][3][4] and HyTime[5][6][7].

CJK DOCP [1] has been discussing about multilingual documents[8]. We think user requirements for such document are as follows;

- Multilingual characters can be described.
- Each language portions should be identified.
- Relationships between each language portions should be described.

SGML has the abilities which can identify arbitrary portions of a document and can describe multilingual characters whether they have character code or not.

HyTime, which is an application standard of SGML, has the capability of hyperlinking within a document and among documents.

Then language identification and character representation can be achieved using SGML. Relationships between each language portions can be described using HyTime.

We don't define specific DTD for multilingual documents, but define architectural forms. Users can consequently define their own DTD.

[1]China/Japan/Korea Document Processing Meeting (CJK DOCP) was established in July 1992 to discuss about the technology for East Asian document processing communication. Major topics are applications of ISO developed document standards to East Asian document environments. Actual works are carried out by the working projects (WPs) called SPREAD "Standardization Projects regarding East Asian Documents", which include WP for DTD of CJK multilingual documents, WP for CJK multilingual hypermedia documents, WP for Fonts of CJK multilingual documents, and WP for CJK multilingual linguistic-markup.

In section 2, we explain what are multilingual documents and introduce their possible applications. In section 3, Processing models of SGML and HyTime are summarized. In section 4, How to represent multilingual characters and language identification techniques are discussed. In section 5, Representation of hyperlinks between different language portions are discussed. In section 6, architectural forms for multilingual documents are proposed.

2 Multilingual documents

A multilingual document is a document which is closely related to more than two languages. One type of such documents has more than two portions of same content in different languages, such as multilingual manuals and diplomatic documents. Another type document contains foreign language portions in the body of main language.

A multilingual document can be classified into an interlaced type and a separated type (see figure 1). Both type of documents can be converted to another type since they are structured.

Interlaced type A document which is comprised with segments denoted by single language. Creating a document of this type, a multilingual text editor is required.

Separated type A set of monolingual documents, whose related portions are linked together. Creating a document of this type, a monolingual text editor is adequate but the mechanism of inter-document link is needed.

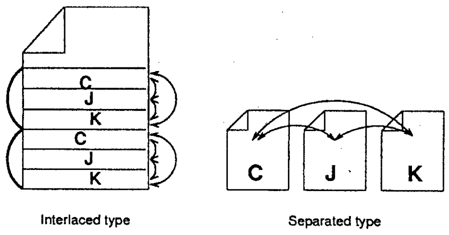

Figure 1: Types of multilingual documents

2.1 Applications of multilingual documents

There are two type of multilingual document applications. Applications of one type utilize relationship between different language portions, which relationship could be "this part is translated from other certain part".

Applications of another type do not utilize such relationship but only language identification information. Such applications can recognize language switching context and does specified operation. For example, we could print multilingual documents if the language contexts were given.

2.1.1 On-line viewing

Sample system is comprised with master database, online document viewer, and printer. The required document type is different for each stage. For the printed manual, the source document should be separated into monolingual documents to ease to print. For the on-line viewer program, separated document with hyperlink should be provided. For maintenance stage, translated phrase should be located in turn. (See figure2)

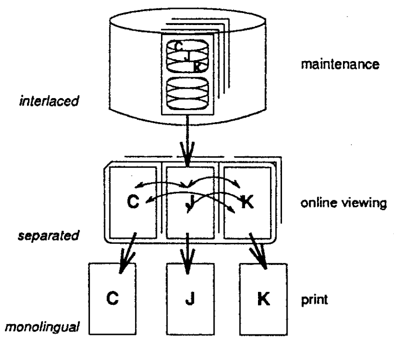

Figure 2: Online viewing application

2.1.2 Translation

There are three stages to make and print a translation material (See figure3).

1. An original monolingual document is gotten (or created). In this stage, the document is written using single language.
2. Original document is translated into other language, then reviewed. In this stage, a interlaced type description is suitable.
3. Interlaced type document is separated into each language part on printing.

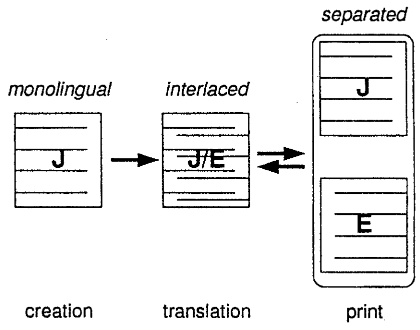

Figure 3: Translation application

3 Processing models of SGML and HyTime

We select SGML and HyTime to describe multilingual documents since they have adequate abilities and are international standards.

Their processing models are shown in this section.

3.1 SGML processing model

SGML is designed to be independent of coded character set. Any coded character sets can be used to describe an SGML document.

Characters in physical storages are processed as following(see figure 4):

1. Characters in physical storages such as file systems and databases could have a variety of coded representations. For example, size of codes could be 7-bit, 8-bit or several 8-bit bytes, and a character set could be ASCII or Japanese graphic character set.

2. Entity manager maps the characters in physical storages to the document character set which is defined in SGML declaration. SGML parser reads document character stream, not physical character stream. Code of the document character set does not have upper limit of the size, so it's size may be 8-bit or 16-bit or more.

3. SGML parser reads document character stream as if it was a single file.

4. SGML applications receive data from the SGML parser. Since the data format is not standardized, characters may be represented in any form.

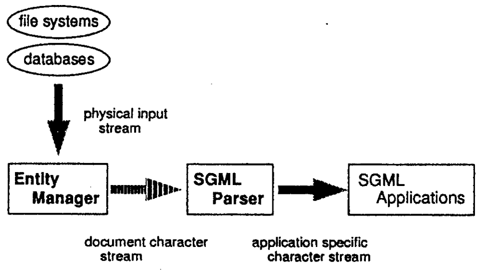

Figure 4: SGML processing model

3.2 Processing model of HyTime

HyTime documents is processed as following (see figure 5):

1. An application orders HyTime engine to parse HyTime documents.

2. HyTime engine orders SGML parser to parse the hub document.

3. The hub document is parsed by SGML parser.

4. HyTime engine analyses the hub document using the output of SGML parser.

5. Documents linked to the hub document are processed likewise.

6. HyTime engine builds internal database.

7. An application does its job with interaction to HyTime engine.

In principle HyTime engine does not care what character code set is used or whether characters are multibyte or single byte, because HyTime engine always accesses data through SGML parser (in detail, entity manager).

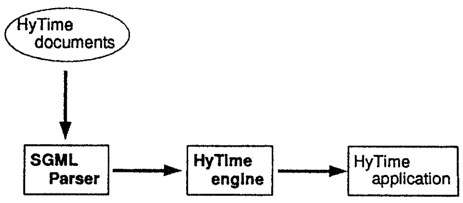

Figure 5: HyTime processing model

3.3 Architectural forms

DTDs are often defined and standardized in specific fields. But it is impossible to fulfill all users' requirements by only one DTD.

Defining architectural forms, which are meta rules to document instances, users could define their own DTDs based on the architectural forms.

3.3.1 Concepts of Architectural Forms

Architectural forms, which are concept proposed by HyTime, are rules on constructing a DTD and an instance. For example, "contextual link" form is defined as follows:

```
<!element clink -- Contextual link --
    - O (%HyBrid;)*>
<!attlist clink
    HyTime  NAME  clink
    id      ID    #IMPLIED
        -- Default: none --
    linkend -- Link end --
    IDREF #REQUIRED
>
```

This form looks like DTD itself, but not. A form is not identified by its element type name but the "HyTime" attribute value. Even if a DTD designer changes the element type name, a form

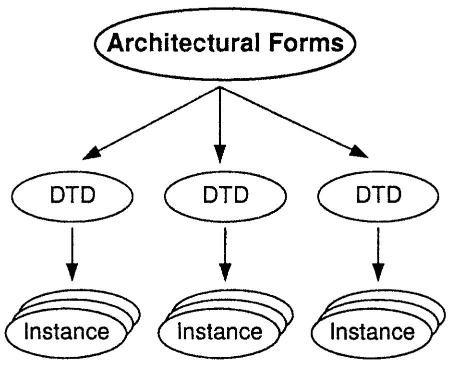

Figure 6: Architectural form and DTD

is recognized according to the value of "HyTime" attribute.

When designing user's DTD, an application designer could use different element type name, add some attributes, delete some attributes, and change the content model under some constraints. Also he/she could use element types which have no relationship to architectural forms.

Relationships between architectural forms, DTDs, and instances are shown in figure 6.

Application programmer could use forms. if an element has attribute "HyTime", it has attributes defined in corresponding architectural form. If an application is made to process according to architectural forms instead of DTD itself, whatever DTDs corresponding to the architectural forms could be processed.

4 Character representations and language identification

4.1 Character representations of multilingual documents

There are some methods to represent characters needed for multilingual SGML documents.

4.1.1 Character entity

SGML provides a method, SDATA entity, to describe characters which have no coded representation in the document character set. For example, some characters are defined in as follows.

```
<!ENTITY curren SDATA "[curren]"
    --=general currency sign-->
<!ENTITY pound  SDATA "[pound ]"
    --=pound sign-->
<!ENTITY dollar SDATA "[dollar]"
    --=dollar sign-->
<!ENTITY cent   SDATA "[cent  ]"
    --=cent sign-->
<!ENTITY yen    SDATA "[yen   ]"
    --/yen =yen sign-->
```

Using these definitions, a yen sign in an SGML document would be like "&yen;10,000". Other characters could be defined.

If there are a few characters outside code set, this method is adequate.

4.1.2 Code extension

Most of existing character code set are designed to fulfill the requirement for a national body. If one who wants to use foreign languages in the same file on computers should introduce the code extension mechanism defined in ISO 2022[9] to enable to mix the different code set.

Latin-based alphabet could be described with 7-bit or 8-bit codes. East asian characters needs 16-bit or more codes.

Since all characters needed to each document are not always included in existing standard code sets, character entities also need to be used.

It is convenient to fix the base character set, which is used as markup and document content, ISO 646 is preferable. Language dependent characters would be described with codes their national bodies define.

4.1.3 Universal code

A character set including large multilingual characters, such as ISO 10646[10], can represent most of needed characters.

Since there are characters that does not defined in such code set in general, character entities also need to be used.

There are few implementations of ISO 10646 conforming system, then this method is not practical.

4.2 Language identification of multilingual documents

4.2.1 Possible technique

There are several approach of how to identify the language.

Element type Element types can identify the language like follows:

```
<english>I am Japanese.</english>
<japanese>私は日本人です。</japanese>
<chinese>我是日本人。</chinese>
```

Marked section Marked sections can also identify the languages of each part.

```
<![ %english; [
I am Japanese.]]>
<![ %japanese; [
私は日本人です。]]>
<![ %chinese; [
我是日本人。]]>
```

Concurrent document SGML allows a document to have multiple structures. Preparing DTDs for each language, each language portion could be identified as follows:

```
<(english)p>I am Japanese.
<(japanese)p>私は日本人です。
<(chinese)p>我是日本人。
```

Attribute SGML allows an element to have some attributes.

```
<p lang=english>I am Japanese.
<p lang=japanese>私は日本人です。
<p lang=chinese>我是日本人。
```

4.2.2 Recommended language identification technique

The "element type" method decreases the flexibility of constructing user's own DTD.

By the "marked section" method, document instances should re-parse when user would like to change the language.

The "concurrent document structure" is difficult to support since few SGML systems support CONCUR feature of SGML.

On this consideration, we recommend the "attribute" method. If we use attributes for this purpose, these attributes should be specified by architectural forms to permit user's own DTD.

Values of this language identifying attribute need be standardized. TEI suggests to use the codes of ISO 639 to this purpose[11]. For example, "ja" for Japanese, "zh" for Chinese, and "ko" for Korean.

4.2.3 Default value for attributes

In Multilingual documents, some attributes, such as for language identification, need be specified. But it is tedious to write such attribute values explicitly to each element. Then setting default values is useful.

There are two ways to set an default attribute value with SGML:

- Specifying default value to an element type.
- Specifying "CURRENT" value to an element type.

By both of them, default values can be set only to an element type.

Using HyTime facility "default value list", the assignment of attribute values can be controlled in the element structure of a document.

When default value lists as below are declared in an instance,

```
<dvlist id=dvc>lang=Chinese</>
<dvlist id=dvj>lang=Japanese</>
<dvlist id=dvk>lang=Korean</>
```

we can specify default values for impliable "lang" attribute of subelements as follows:

```
<section subdvl=dvj>...<p>...<p>...
</section>
```

In this example, if "p" has a impliable "lang" attribute, its value is assigned to "Japanese".

5 Hyperlink representation

Relationships among different language portions could be described as hyperlinks. To represent them, HyTime, which is an application of SGML, is needed.

HyTime provides following two types of hyperlinking (see figure 7). Of course these types can be combined together.

CLINK A clink has object in its content and its linkend represents the location of the related object.

ILINK A ilink has several linkends which represent the locations of the related objects. It could have one of the related objects in its content.

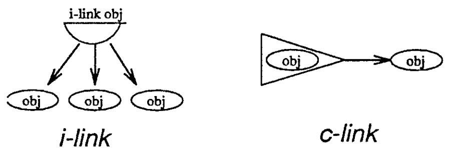

Figure 7: Basic Types of Hyperlinks

CLINK can not describe relationships among three or more object, but ILINK can.

The two basic types of description and the two basic types of hyperlink make wide variety of representation of multilingual documents (see figure8).

Interlaced with CLINK A file contains all language portions and relationships among them are represented with CLINK.

```
<clink linkend=C1 lang=japanese>
私は日本人です。</>
<p id=C1 lang=chinese>
我是日本人。</>
```

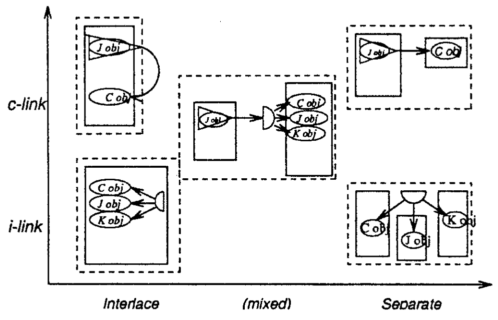

Figure 8: Varieties of multilingual representation

Interlaced with ILINK A file contains all language portions and relationships among them are represented with ILINK.

```
<p id=J2 lang=japanese>私は日本人です。
<p id=C2 lang=chinese>我是日本人。
<p id=K2 lang=korean>..........
<ilink linkends="J2 C2 K2">
```

Separated with CLINK A file contains one language. Relationships among different language portions are represented with CLINK.

```
   ----- (in "jdoc.txt") -----
<!ENTITY cdoc SYSTEM "c:cdoc.txt">

<nameloc id=L3><nmlist nametype=entity
 docorsub=cdoc>C3</>
<clink lang=japanese linkend=L3>
 私は日本人です。</>
   ----- (in "cdoc.txt") -----
<p id=C3 lang=chinese>我是日本人。</>
```

Separated with ILINK A file contains one language. Relationships among different language portions are represented with ILINK.

```
   ----- (in "jdoc.txt") -----
<p id=J4 lang=japanese>私は日本人です。</>
   ----- (in "cdoc.txt") -----
<p id=C4 lang=chinese>我是日本人。</>
   ----- (in "kdoc.txt") -----
<p id=K4 lang=korean>..........</>
   ----- (in "main.txt") -----
<!ENTITY jdoc SYSTEM "c:jdoc.txt">
<!ENTITY cdoc SYSTEM "c:cdoc.txt">
<!ENTITY kdoc SYSTEM "c:kdoc.txt">

<nameloc id=LJ4><nmlist nametype=entity
```

```
 docorsub=jdoc>J4</>
<nameloc id=LC4><nmlist nametype=entity
 docorsub=cdoc>C4</>
<nameloc id=LK4><nmlist nametype=entity
 docorsub=kdoc>K4</>
<ilink linkends="LJ4 LC4 LK4">
```

6 Architectural forms for multilingual documents

We propose following architectural forms for multilingual documents. They conform to HyTime hyperlinking facility.

In these forms, MULLING attribute shows the compliance to the architecture of these forms. Attribute "language" identifies the language of the element content. Its value is one of ISO 639 codes.

```
<!attlist all-mlelm
    language NAME    #IMPLIED
    -- Constraint: ISO 639 code --
>
<!-- Multilingual Independent Link -->
<!element ilink - O (%HyBrid;)*>
<!attlist ilink
    HyTime   NAME    #FIXED     ilink
-- other HyTime ilink attributes --
    MULLING  NAME    #FIXED     mul-ilink
>
<!-- Multilingual Contextual Link -->
<!element clink - O (%HyBrid;)*>
<!attlist clink
    HyTime   NAME    #FIXED     clink
-- other HyTime clink attributes --
    MULLING  NAME    #FIXED     mul-clink
>
```

7 Conclusion

There are many problems to use multilingual documents. Texts in a variety of languages could be displayed and printed, large number of ideographic characters could be entered, etc.

We proposed some techniques to represent multilingual document using SGML and HyTime. HyTime's abilities and the concept of architectural forms are very useful to the area of multilingual documents.

Now, there are few implementations of HyTime engine that it is hard to make and process HyTime documents. But we hope such engine would be popularized and these techniques could be applied.

Acknowledgements

We would like thank CJK DOCP members that discussed on East Asian multilingual document processing.

References

[1] ISO 8879. *Information Processing — Text and Office Systems — Standard Generalized Markup Language (SGML)*, 1986.

[2] JIS X 4151. *文書記述言語 SGML*, 1992.

[3] Charles F. Goldfarb. *The SGML Handbook*. Oxford University Press, New York, 1990.

[4] Eric van Herwijnen. *Practical SGML 2nd edition*. Kluwer Academic Publishers, Boston, 1994.

[5] ISO/IEC 10744. *Information technology — Hypermedia/Time-based Structuring Language (HyTime)*, 1992.

[6] JIS X 4155. *ハイパメディア及び時間依存情報の構造化言語 (HyTime)*, 1994.

[7] Steven J. DeRose and David G. Durand. *Making Hypermedia Work: A Use's Guide to HyTime*. Kluwer Academic Publishers, Boston, 1994.

[8] (社) 日本事務機械工業会. 平成 *5* 年度 *(1993* 年度*)* 実装規約小委員会報告書, 1994.

[9] ISO 2022. *Information processing — ISO 7-bit and 8-bit coded character sets — Code extension techniques*, 1986.

[10] ISO/IEC 10646-1. *Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane*, 1993.

[11] C. M. Sperberg-McQueen and Lou Burnard, editors. *Guidelines for electronic text encoding and interchange (TEI P3)*. Text Encoding Initiative, 1994.