Multilingual Information Exchange through the World−Wide Web

TAKADA Toshihiro
NTT Basic Research Labs.
Nippon Telegraph and Telephone Corp.

Abstract

Multilingual facilities are indispensable to make the World−Wide Web really worldwide. However, most of the current implementations of the WWW support only Latin−1 (ISO 8859−1) character set. This paper describes a method for exchanging multilingual information through the WWW, based on our experience on development of Multi−Localization Enhancement of NCSA Mosaic for X. Our experimental enhancement for Mosaic provides use of various national characters. This paper also describes future issues of multilingual information exchange.

1 Introduction

The World−Wide Web (WWW) [1] gives us a chance to touch with the world of global information. Since the WWW has been born, it has rapidly grown, and now we can enjoy visits to over 40 countries from our own room. It has been growing not only geographically but also to be used by people of all ages. We can enjoy even hyperdocuments made by elementary school students.

However, we still have some problems. One of the problems of the current WWW is lack of a multilingual environment. Most of the current implementations of the WWW support only Latin−1 (ISO 8859−1) character set. In this world, there are many languages and characters. It is natural for elementary school students in Japan to write their personal pages in Japanese. When some people are going to introduce cultures, histories or languages of their countries, it is easy to explain if their national characters or languages can be used as an example. Implementing a multilingual environment are indispensable to make the WWW really *worldwide*.

In this paper, we describe a method for exchanging multilingual information through the World−Wide Web, based on our experience on development of Multi−Localization Enhancement of NCSA Mosaic for X [2]. Our experimental enhancement for Mosaic realizes use of various national characters, including Czech, Cyrillic, Greek, Hebrew, Turkish, Chinese, Korean, Japanese and more. The multilingual facilities which provided by this enhancement include:

- On−the−fly user's choices of appropriate character sets
- An automatic code detection mechanism based on ISO 2022 character sets designation
- Ability to display bi−directional languages (e.g. Hebrew) in Visual or Implicit mode
- Implementation of the language−preference feature of HTTP, namely *Accept−Language:* request field

In the rest of the paper, we first show what are problems to provide a multilingual environment with the WWW, and briefly introduce our Multi−Localization Enhancement of NCSA Mosaic for X. We then explain how we solve the problems in our experimental enhancement for Mosaic. Future issues of multilingual information exchange such as encoding methods of multilingual characters within the WWW, its character sets and language specifications will be discussed finally.

2 The Problems of Multilingual Documents

In this section we show what are problems to provide a multilingual environment with the WWW.

2.1 Character Set

In a multilingual environment, documents may contain not only ASCII (or Latin−1) characters, but also Hebrew, Korean and many other characters from different character sets. For example, over 20 character sets, some are 8−bit character sets and others are 16−bit character sets, are registered in ECMA (European Computer Manufactures Association) as of today. A multilingual system has to manage these character sets.

2.2 Encoding Methods

The another issue is an encoding method. In this paper, *Encoding* means how to use different character sets without getting them confused, as well as how to represent a character of a particular character set as some bit pattern.

Many encoding method, both standardized and private, exist and actually used in the world. A multilingual system has to deal with these encoding methods as much far as possible.

2.3 Handling Bi−directionality

Some languages such as Hebrew and Arabic are read from right to left. The problem is that these languages can have bi−directional data such as mixed Hebrew and English on the same line. A multilingual system has to handle such bi−directionality.

2.4 Negotiation Algorithm for Language Preferences

It is also an important feature to multilingual system that allows a user to specify his/her preferable languages. Let us consider about creating a bilingual WWW server.

The most popular way to create a bilingual server, for example in either English or Japanese, is providing both versions of a web, and putting a language switching toggle button into each document. Many bilingual WWW servers are currently implemented in this way.

Another way to create a bilingual server is using a user preferences feature defined in the HTTP protocol [3]. This feature makes possible that the same URL can be used to retrieve a document in different format. For example, with this feature, a user can say:

```
I want to know about a history of Cambodia.
I can read English, but French is acceptable.
```

The server requested as above try to find a hyperdocument in English firstly. A history hyperbook in English will be returned to the user if it will be found, if not and if French version found, one in French will be returned. The later way is more easy to create for a server builder and easy to walk around for users than the former way.

3 Multi−Localization enhanced of NCSA Mosaic for X

Multi−Localization Enhancement of NCSA Mosaic for X (Mosaic−L10N) is upper compatible with the original NCSA's Mosaic for X, and makes use of various national character sets. A user can switch among character sets in *one* Mosaic. Figure 1 and 2 show snapshots of Mosaic−L10N window.

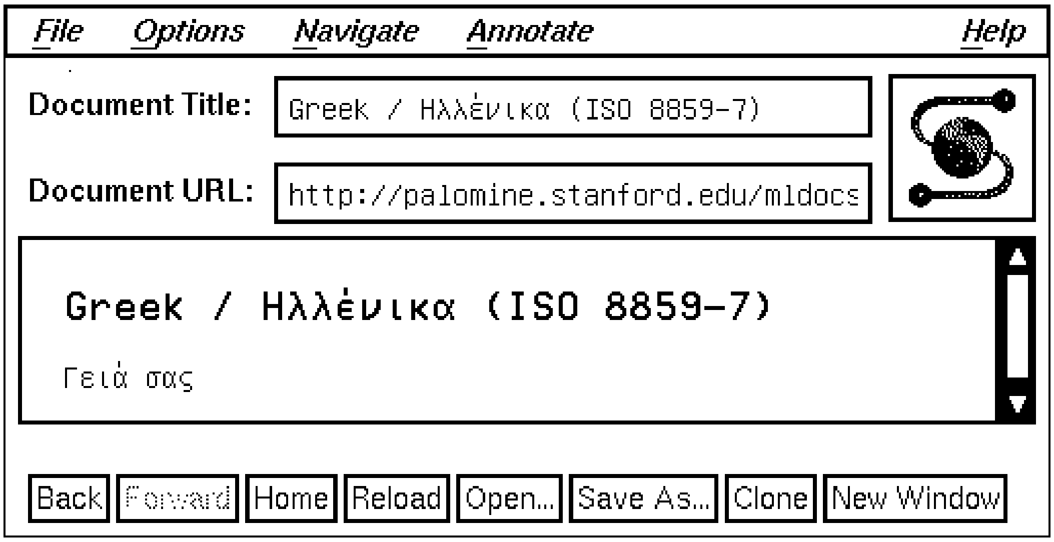

Figure 1: A snapshot of Mosaic−L10N window (in Greek)

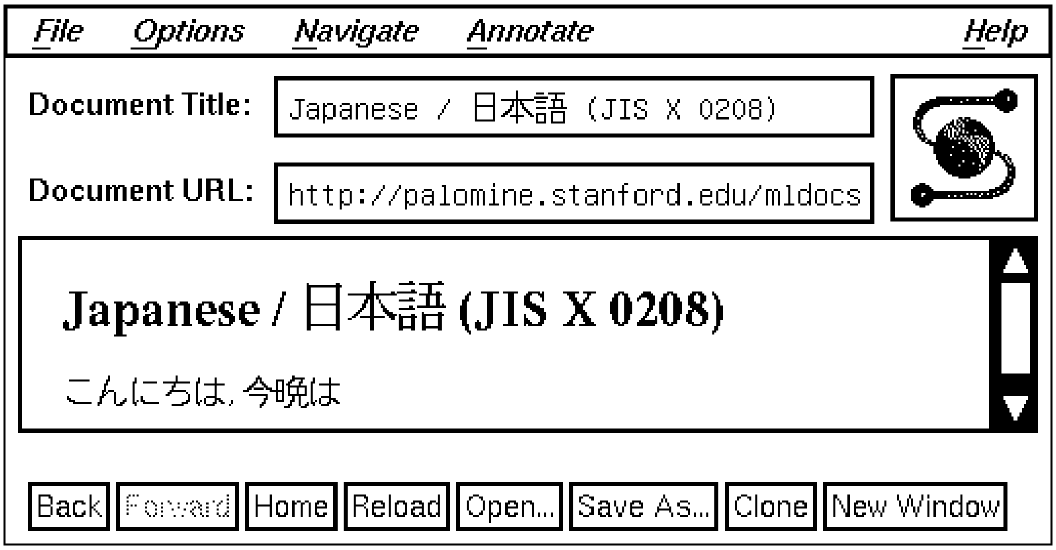

Figure 2: A snapshot of Mosaic−L10N window (in Japanese)

4 How Mosaic−L10N Solves The Problems

In this section, we describe how we solved the problems, shown in the section 2, in Mosaic−L10N.

4.1 Character Sets

Mosaic-L10N can handle character sets that meet technical requirements standardized in ISO 2022 and satisfies format requirements of ISO 2375. Among character sets registered in ECMA, that is the Registrations Authority for ISO 2375, Mosaic-L10N currently supports those shown in Table 1.

```
Character Set
==============================
ISO 646 USA (ASCII)
ISO 8859-1 Latin alphabet No.1
ISO 8859-2 Latin alphabet No.2
ISO 8859-3 Latin alphabet No.3
ISO 8859-4 Latin alphabet No.4
ISO 8859-5 Cyrillic alphabet
ISO 8859-7 Greek alphabet
ISO 8859-8 Hebrew alphabet
ISO 8859-9 Latin alphabet No.5
GB 2312-1980 Chinese
KSC 5601-1987 Korean
JIS X 0208-1983 Japanese
------------------------------
KOI-8 Cyrillic alphabet
Big5 Chinese

Table 1: Currently supported character sets
```

As shown in the Table 1., Mosaic-L10N can also handle a few private character sets (e.g. KOI-8 for Cyrillic and Big5 for Chinese).

However, the current implementation of Mosaic-L10N is not truly multilingual system. This means that we can only use two character sets (for example, Latin-1 and Greek) in a single document. Users have to change character sets to appropriate on by using *Font* menu (Figure 3).

4.2 Encoding Methods

Mosaic-L10N supports a subset of ISO 2022's character sets designation escape sequences as its encoding method. For example, documents encoded in `ISO-2022-JP` (RFC-1468) [4] or `ISO-2022-KR` (RFC-1557) [5] are automatically recognized to the documents in Japanese or Korean. However, documents in ISO 8859-X, EUC-C/J/K, KOI-8 and Big5 cannot be judged what character set is used without external information. Mosaic-L10N supports the ISO 2022 initial designation sequences shown in Table 2. (Note: We will always use G0 and G1 for GL and GR, respectively.) If a document includes the one of these escape sequences, the document is displayed by appropriate fonts without manual character set choice by users.

```
"<ESC> - A"    designate right-hand part of ISO 8859-1 into G1
"<ESC> - B"    designate right-hand part of ISO 8859-2 into G1
"<ESC> - C"    designate right-hand part of ISO 8859-3 into G1
"<ESC> - D"    designate right-hand part of ISO 8859-4 into G1
"<ESC> - L"    designate right-hand part of ISO 8859-5 into G1
"<ESC> - F"    designate right-hand part of ISO 8859-7 into G1
"<ESC> - H"    designate right-hand part of ISO 8859-8 into G1
"<ESC> - M"    designate right-hand part of ISO 8859-9 into G1
"<ESC> $ ) A"  designate GB 2312-1980 into G1
```

```
"<ESC> $ ) B"  designate JIS X 0208-1983 into G1
"<ESC> $ ) C"  designate KSC 5601-1987 into G1
"<ESC> ( B"    designate 7-bit ASCII graphics into G0
"<ESC> $ B"    designate JIS X 0208-1983 into G0
```

Table 2: Currently supported ISO 2022 sequences

For KOI–8, Big5 and GB in HZ encoding, we have no way to specify their own character sets, because they are private character sets or private encoding methods.

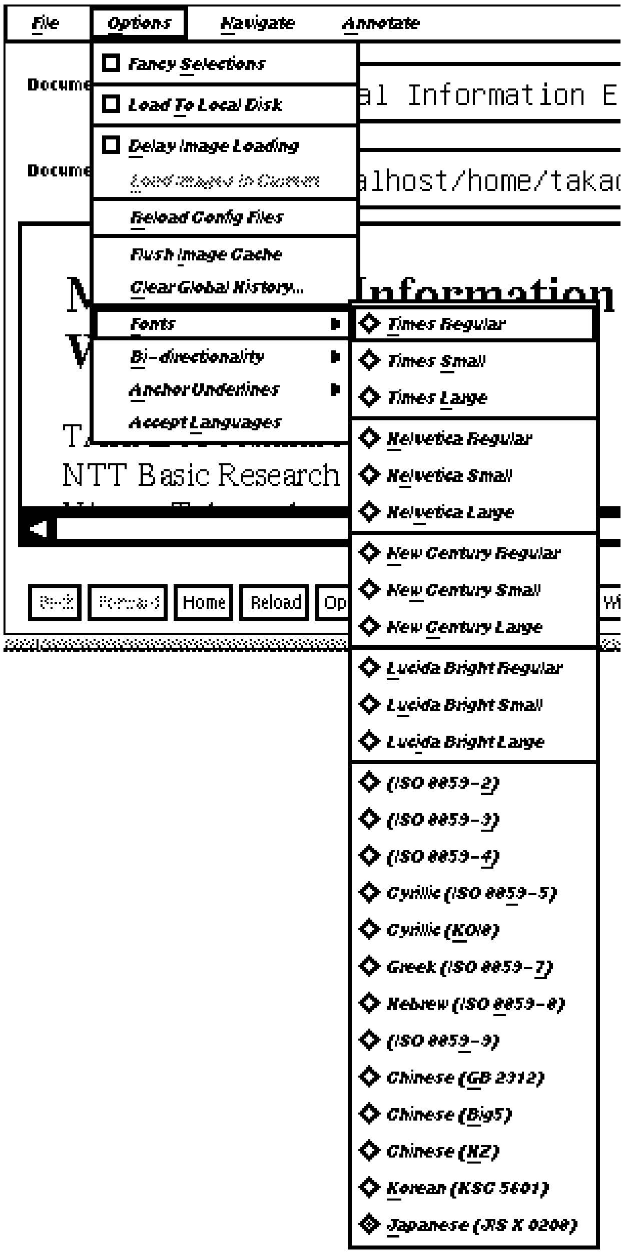

Figure 3: A snapshot of Mosaic–L10N Font Menu

4.3 Handling Bi−directionality

Mosaic−L10N supports bi−directional languages (e.g. Hebrew). There are three methods (shown below) for bi−directional data [8].

- **Visual:** Visual directionality is a presentation method that displays text according to the primary display direction only, which is left to right.
- **Implicit:** Implicit directionality is a presentation method in which the direction is determined by an algorithm according to the type of characters and their position relative to the adjacent characters and according to their primary direction.
- **Explicit:** Explicit directionality is a presentation method in which the direction is explicitly defined by using control sequences (defined in ECMA TR/53 [5]) which are interleaved within the text and are used for direction determination.

Currently, both *Visual* and *Implicit* directionality is supported, but *Explicit* directionality is not supported yet.

In RFC−1555 [7] and RFC−1556 [8], three different MIME character sets are defined for each bi−directionality. But current implementation of Mosaic−L10N does not distinguish these MIME character sets, but allows a user to choose a bi−directionality among them by menu (for technical details, see [8]).

4.4 Negotiation Algorithm for Language Preferences

HTTP protocol defines a negotiation mechanism for user preferable languages, namely *Accept−Languages:* request header field. A user can specify the list of languages which is preferable in the response, and a server which understand this request is possible to return documents desirable for the user from the same URL. With Mosaic−L10N, the value of *Accept−Languages:* header field which users specify is directly passed to HTTP server.

As of today, a strict interpretation of this request header field is not determined, but use of a language code, which is an ISO 639 language code with an optional ISO 3166 country code to specify a national variant, is encourage now. For example, `en_UK` means that the content of the message is in British English, while `en` means that the language is English in one of its forms (for technical details, see [3]).

5 Future Issues

5.1 Character Sets and Encoding Methods

As we told in the section 4.1, Mosaic−L10N is not truly multilingual implementation. The main reason why is there are neither character sets nor encoding methods, which is widely accepted in the WWW community, for the multilingual documents. There are several candidates, for example,

- Using ISO 2022
- Using ISO 10646 or Unicode

but there are many ways of an encoding multilingual text along either ISO 2022 or ISO 10646, though.

One of the other ways is using attributes in HTML+. The most recent HTML+ specification [9] has LANG attribute as following.

```
<Q LANG="fr">Je m'aveugle.</Q>
```

However, it is used only for permitting language dependent layout and hyphenation decisions, but not for specifying character sets or encoding methods.

Using character set type parameter in MIME sense is another candidate. We can use *Content−type:* with character set type parameter in HTTP headers as following.

```
Content-type: text/html; charset=iso-2022-jp
```

However, almost all of current WWW clients not only cannot understand type parameters, but also misunderstand as its Content−type is `"text/html; charset=\"iso-8859-2\""` itself.

5.2 Language Specifications

As we described in the section 4.4, the interpretation of *language−preference request* is still open problem. Let us consider about another example. When a user say

```
I want to learn Japanese language in French.
```

to the WWW, he/she will get a textbook for Japanese language written in French. These kind of textbook includes both Japanese and French characters. What is *the language type* of a such kind of bilingual document? it is not specified yet.

6 Summary

We have developed Multi−Localization Enhancement of NCSA Mosaic for X to realize use of various national characters in the WWW. It provides:

- On−the−fly user's choices of appropriate character sets
- An automatic code detection based on ISO 2022
- Ability to handle bi−directional languages
- Implementation of the language−preference feature of HTTP

We are now considering implementing truly multilingual version of Mosaic. It will be provided for Macintosh and Windows as well as for X. However, we have a few open questions related multilingual documents in the WWW.

- Multiple character sets and its encoding methods
- Definition of the language type for multilingual documents

We have to make a consensus for these problems, to make the WWW truly multilingual,

Acknowledgements

First of all, I am deeply grateful to Tim Berners−Lee, WWW people at CERN, Marc Andreessen, Eric Bina and Mosaic people at NCSA for their great works. Mosaic−L10N is strongly based on Japanese localization for xmosaic−1.2 done by Youichi Watanabe. I would like to thank him. I also thank many people for their helpful comments and encouragements through Mail and News.

References

[1] Tim Berners−Lee, et. al.:
The World Wide Web Initiative, In proceedings of INET '93, Internet Society, 1993.

[2] Eric Bina and Marc Andreessen:
About NCSA Mosaic for the X Window System,
<http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/Docs/help−about.html>, 1994.

[3] Tim Berners−Lee:
HTTP: A Protocol for Networked Information,
<http://info.cern.ch/hypertext/WWW/MarkUp/MarkUp.html>, 1993.

[4] Jun Murai, Mark Crispin and Erik M. van der Poel:
Japanese Character Encoding for Internet Messages, RFC−1468, 1993.

[5] Uhhyung Choi, Kilnam Chon and Hyunje Park
Korean Character Encoding for Internet Message, RFC−1557, 1993.

[6] ECMA:
Handling of Bi−Directional Texts, European Computer Manufacturers Association, 1992.

[7] Hank Nussbacher and Yehavi Bourvine:
Hebrew Character Encoding for Internet Messages, RFC−1555, 1993.

[8] Hank Nussbacher:
Handling of Bi−directional Texts in MIME, RFC−1556, 1993.

[9] David Raggett:
HTML+ (Hypertext Markup Format),
<http://info.cern.ch/hypertext/WWW/MarkUp/HTMLPlus/htmlplus_1.html>, 1994.

Appendix: Availability of Multi−Localization Enhancement of NCSA Mosaic for X

Multi−Localization Enhancement of NCSA Mosaic for X is freely and publicly available and distributed. Both source level patch from original NCSA's Mosaic and binary executables for some platforms is available by anonymous FTP from:

```
ftp://www.ntt.jp/networking/WWW/Mosaic-l10n
```

Up−to−date documentations for Mosaic−L10N is also available at

```
http://www.ntt.jp/Mosaic-l10n/README.html
```