Displaying Multi-script Data on the Web

Kiyonori NAGASAKI[1)] and Makoto MINEGISHI[2)] and Izumi HOSHI[3)]

Abstract
Currently, easy computer environments for multi-scripts, such as Windows2000/XP and MacOS X, require the Unicode and Opentype standards. However, even today, it is not possible to install these standards on half of all Web clients. In order to display multi-script texts on the Web, some improvisation techniques are needed for old clients, such as Windows 9x and Me, and for some small clients, such as mobile phones and PDAs. In this paper, I will introduce a technique for displaying multi-script data on almost Web clients. The technique is called Mojiyaki.

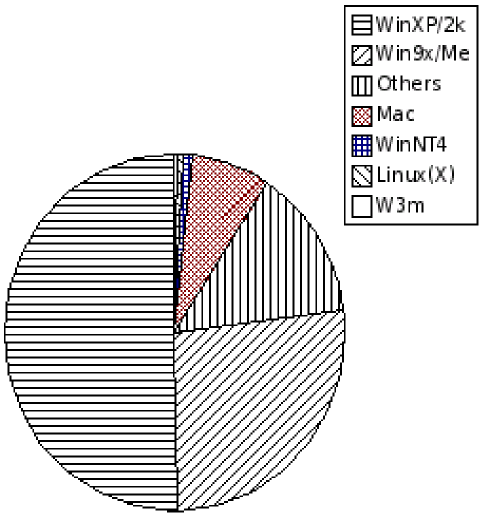
Keywords: Multi-scripts, Mojiyaki, the Web

1 Introduction

On the World Wide Web, there are an enormous number of Web pages, estimated at over three billion, written in many different languages. One problem is to write these languages in their own scripts, due to the fact that there are many character code sets that are not compatible with each other. Recently, with the spread of Unicode and Opentype (e.g., Windows2000/XP, Mac OS X), it is possible to display information in various languages in their own scripts. However, even today, these new computer environments are still not used by half of Web users. For example, an access log connected to "Mojiyaki" indicates that only about a half of the clients can use Unicode and Opentype environments. See Figure 1.

It is, therefore, still necessary to continue utilizing improvisation techniques in order to display multi-script data on the Web. In this paper, I will introduce "Mojiyaki", which can easily display multi-script data by actively creating inline images on the Web.

Figure 1: This graph shows the percentages for OSs accessing to "Mojiyaki"



2 Some techniques for old systems

With Unicode and Opentype, it is possible to display texts on the Web in various scripts, even Indic scripts. However, many Web clients do not have Unicode or Opentype installed, and some languages are not completely supported within Unicode or Opentype. Accordingly, when we create Web pages written in various scripts, and, particularly, when using scripts that are not completely supported by Unicode and Opentype, it is not always possible to make the Web pages with Unicode.

In order to display multi-script texts on the Web, some improvisation methods are needed for old clients, such as Windows9x, and Me.

At least, three such techniques are still in use on the Web:

1. Install fonts for each script.
2. Draw scripts with Java applets on the browser.
3. Draw fonts as images.

These three methods are explained briefly.

1) Faculty of International Studies, Yamaguchi Prefectural University
2) Research Institute for Languages and Cultures of Asia and Africa, Tokyo University of Foreign Studies
3) Research Institute for Languages and Cultures of Asia and Africa, Tokyo University of Foreign Studies

2.1 Install fonts for each script

First, installing fonts is not easy for the average Web user. Various character codes are used on the Web, so it is necessary to install fonts for each code. Even when the same script is used on a Web page, often different character codes are used to display the script.

Second, copyright for fonts is an important problem. It is often not possible to distribute fonts, because many are protected by copyright. Thus, if a Web publisher wants to distribute a font in order to display their Web page, they may have to make their own font to overcome copyright problems.

Third, some Web clients are not able to install fonts, such as Web-enabled mobile phones and PDAs. Most of these devices have insufficient memory to install various fonts, or cannot change the system fonts.

For these reasons, it is not practical to install fonts for each script.

2.2 Draw scripts with Java applets on the Web browser

Drawing scripts with Java applets on the Web browser is one solution, because Java applets can freely draw anything on a Web browser running Java Virtual Machine (JVM), and the Java language supports Unicode. In 1997, this technique of drawing scripts with Java applets was already being used for Chinese, Tamil, Persian, and Turkish.[1]

Subsequently, Maeda [2000][2] presented a Java applet system that could not display multi-script data, but could also be used to write it without the necessary fonts being installed on the Web browser. This technique is convenient because it does not require implementation of the script.

However, in many schools, students often cannot use the latest personal computers, which can run Java applets satisfactorily. Also, in Asian countries that employ scripts that are not fully supported in computers, the people cannot use Java applets, because they are too heavy for their clients. This represents a difficult problem for this technique.

[1] See Mikami [1997][1] .

2.3 Create fonts-as-images.

Creating fonts-as-images is the simplest technique. It easily displays multi-script data on any browsers that can display inline images, without any special software or fonts on the Web clients.[2]

It is hard to create fonts-as-images, if each time a new text is composed, it is necessary to run software to create the images and to write HTML tags to display the images on the Web page. However, with a CGI program, fonts-as-images can be created from the ASCII transliteration texts on the Web server, so that the clients can receive and display the images, without any special software apart from Web browsers that can display inline images.

This technique has also been adopted for some Web pages. One problem of this technique has been that it is slow to create images on the server and to receive the images via the Internet. However, transmission speeds are less of a problem with broadband, which is now common in many homes, and higher performances for computers and software have reduced the time necessary to create images.

Accordingly, in this paper, I will introduce a technique for creating font-as-images to display multi-script data on the Web called "Mojiyaki - Multilingual Imager for Web".

3 Mojiyaki - Multilingual Imager for Web

Mojiyaki is a server for realizing multi-script Web environments, by means of creating various fonts-as-images.

3.1 Software components of Mojiyaki

Mojiyaki consists of the following software.

- Vine Linux 2.1CR (Red Hat 7.2 compatible)
- Apache 1.3.x
- Perl 5.x
- mod_perl 1.2x
- ImageMagick 5.x
- FreeType 2.x

[2] It can even display data on the first version of mosaic, because it implements inline image with an IMG tag.

- Some True Type Fonts

Linux is an open-source and free operating system. It is suitable to use in constructing the Mojiyaki server, because of its stability and flexibility. Vine Linux 2.1CR is one of the distributions of Linux based on Red Hat Linux. Although this version does not include the latest software, it is sufficient to maintain a stable service as an Internet server.[3]

Apache 1.3.x has been the most popular HTTP server, developed by way of open-source. It has a number of useful features as a Web. While version 2.0 has been released, Mojiyaki has adopted version 1.3, because this seems to be more stable than version 2.0.[4]

Perl is a stable, cross-platform programming language. CGI scripts on the Web are often written in this language. It is also suitable for handling multi-script data, because it has supported Unicode since version 5.8. Furthermore, Perl can be greatly extended by adding modules.[5]

mod_perl is a module for Apache. It has also been developed with open-source. It is a persistent Perl interpreter embedded in the Web server. Not only does this eliminate the need to start an external interpreter, but it also avoids the burden of Perl start-up time. As this increases the speed of CGI scripts written in Perl, it can be used to quickly create fonts-as-images.[6]

ImageMagick 5.x can create images through perl modules. Moreover, through the FreeType 2.x library, it can draw any character on images using Truetype fonts.[7]

3.2 Hardware components of Mojiyaki

Mojiyaki runs on inexpensive hardware, including:

- CPU: Intel Pentium 4 1700MHz
- Memory: About 900MB
- HDD: About 40GB
- NIC: 100Mbase-TX

While these are not the latest hardwire devices, they are still sufficient for Mojiyaki

3.3 Supported scripts

Mojiyaki can create various fonts-as-images quickly. It has implemented the following scripts.

- Devanagari (Hindi, Sanskrit)
- Tibetan
- Thai
- Laos
- Arabic Scripts (Arabic, Uighur)
- Japanese, Korean, (Chinese)

In implementing these scripts, specialists of each language assisted us.[8]

3.4 Features of Mojiyaki

The Mojiyaki program is written in mod_perl, so that it can quickly generate images. All scripts can be displayed together on a single Web page, as the program is able to generate all types of ligature. This is possible because of features of FreeType and ImageMagick.

For the Laos version of Mojiyaki, some additional features have been added, such as given an edge-color to characters, by adopting features of the 5.5.7 version of ImageMagick.

To display multi-script data on a single Web page as inline images, Mojiyaki requires to the "IMG" tag from HTML. For example, a HTML Tag to display the words "Tibetan Script" in Tibetan is shown below.

```
<IMG SRC="http://mojiyaki.aa.tufs.ac.jp/
conv_TIB.cgi?
sort=bod+yig" ALT="bod yig">
```

This IMG tag displays the following Tibetan scripts.

[3] For further details of Vine Linux, see [3].
[4] For further details of Apache, see [4].
[5] For further details of Perl, see [5].
[6] For further details of mod_perl, see [6].
[7] For further details of ImageMagick, see [7]. About freetype, see [8].

[8] Makoto MINEGISHI and Izumi HOSHI checked the details to utilize Laos, Thai and Tibetan. Haruko SAKAEDANI (formerly at ILCAA, TUFS) checked for Arabic. Jun SUGAWARA (formerly at ILCAA, TUFS) checked for Uighur. Kazuhiko MACHIDA (ILCAA, TUFS) checked the Hindi

Figure 2: Top page of Mojiyaki

Figure 3: "Tibetan Script" in Tibetan

In Mojiyaki, therefore, texts created as images are written in URI (Uniform Resource Identifiers) with ASCII characters that are exchanged according to the "extended Wylie" rule, which is a rule of the ASCII transliteration. In order to handle multi-script data with ASCII characters, URI encoding and ASCII transliteration are used. The former is a technique to convert multibyte texts into ASCII texts, in order to exchange texts on URI. The latter is a framework for human-readable ASCII texts, to handle various non-supported scripts on computers, such as writing, reading, and analyzing Sanskrit in an Japanese computer environment. It is often used by researchers, and already huge texts have been input according to ASCII transliteration. For example, in Arabic, the writing order is left-to-right. However, transliteration with ASCII characters writes in the order of right-to-left. Furthermore, most Indic scripts include a complicated writing order convention, called "Hindi reverse" and many ligatures.

Figure 4: ASCII transliteration and glyph images for "Hindi reverse"

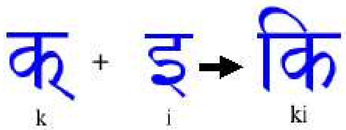
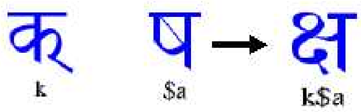


Figure 5: ASCII transliteration and glyph images for a Hindi ligature



Various rules for ASCII transliteration have already been defined. In the case of Sanskrit scripts (Devanāgarī), many rules have been created by individual researchers to input texts, to share them and analyze them on computers.[9]

One rule covers the combination of ASCII upper case and lower case letters to describe phonemes in a left-to-right order. Another rule covers the combination of upper case and lower case and some control characters. However, these rules are all similar in terms of writing order. The difference is in terms of which phoneme applies to which letter. Thus, Mojiyaki implements only one replace routine from left-to-right order for the Hindi reverse order and one combination routine to create the ligatures of Devanāgarī. Other rules are supported by a rule replacement table. Almost all of these can be achieved by only replacing characters. For example, " ख " is transliterates as "k'" in CISA, but[10] replaces "kh" in KH[11] Aiba-shiki and Ono-shiki. "क्षा" is transliterated as "k$aa" in CISA, "kSA" in KH, "k.saa" in Aiba-shiki, and "k@s@a" in Ono-shiki. These are not the same as letters, but the order of writing is the same. It is not difficult to replace these with each other.

The Thai and Laos scripts are alike. They also include special writing order conventions, like the "Hindi reverse" and have some ligatures. Some vowels are placed before a consonant in a syllable.

Arabic scripts have two implementations in Mojiyaki. This is because although the writing order (right-to-left) for Uighur is the same as Arabic, the Uighur writing system differs from other languages using the Arabic script. In Arabic, many letters are connected when written "C(Consonant)-C-C" , becomes " عربي ", with the vowels often being omitted.

However, in Uighur, Arabic letters are separated into each syllable. For example, "V(Vowel)C CVC VC", becomes " ئۇيغۇرچە ".[12]

Thus, Arabic scripts are implemented by two systems in Mojiyaki.

On smaller and older MS-Window systems, such as Windows98 and WindowsME, Chinese, Japanese and Korean scripts are easy to handle within the

[9] Transliteration rules for Sanskrit and Tibetan scripts are listed in [9].

[10] Code for Indic scripts based on standard ASCII

[11] Kyoto-Harvard

[12] For further details of Uighur writing system and computers, see [10].

global IME software. However, with Web-capable mobile phones and PDAs, Mojiyaki is still needed. It is very difficult to exchange these scripts with ASCII transliteration. In such cases, it is possible to easily exchange the scripts into ASCII characters, by means URI-encoding.

3.5 Use of Mojiyaki

Mojiyaki is useful for dynamic contents, such as online electronic dictionaries. It is easy to write HTML tags for Mojiyaki when dynamic contents are created on the server.

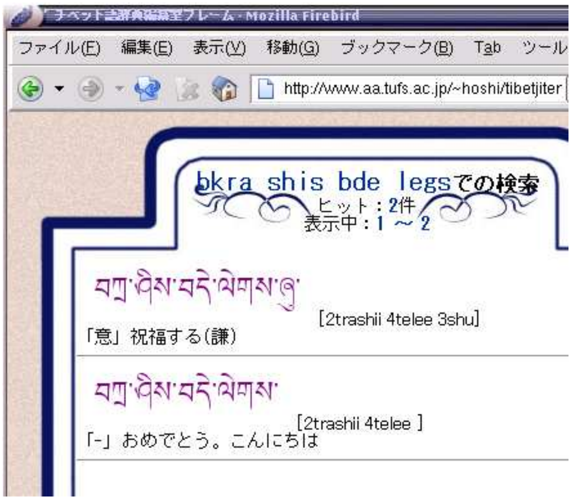
Even with multi-script dictionaries, Mojiyake can display data in the respective scripts.

3.6 Examples of Web sites utilizing Mojiyaki

- ”Tibetan-Japanese Electronic Dictionary” (Research Institute for Languages and Cultures of Asia and Africa, Tokyo University of Foreign Studies)

  <http://www.aa.tufs.ac.jp/˜hoshi/ >

Figure 6: Search result in the Tibetan-Japanese Electronic Dictionary



- ”Welcome to Hindi World” (also at ILCAA, TUFS)

  <http://www.aa.tufs.ac.jp/˜kmach/ >

- ”Tibetan-Sanskrit Search for the Saddharmapu.n.darika (eDic)”

  <http://suzuki.ypu.jp/edic/ >

  This is an example of a Website mixing Tibetan and Sanskrit. See Figure 7.

4 Conclusion

”Mojiyaki” is an improvisation technique, which may shortly become obsolete, as better systems become available. However, although a number of better systems had been released during the last three years ago, they have yet to gain wide-spread use. Moverover, within the last four months, I have just use Mojiyaki for Laos in order to build a Loas learning system. Thus, Mojiyaki may still be needed for some years to come.

Bibliography

[1] MIKAMI [1997] : Yoshihiko MIKAMI, Kenji SEKINE, Nobutoshi KOHARA, *Viewing and Creating Multilingual Web Pages*(O'Reilly, 1997).

[2] MAEDA [2000] : Akira MAEDA, *Studies on Multilingual Information Processing on the Internet*(PhD thesis, Nara Institute of Science and Technology, Sep. 2000).

[3] *Vine Linux Home Page* <http://www.vinelinux.org/ >.

[4] *The Apache HTTP Server Project* <http://httpd.apache.org/ >.

[5] *The Perl Directory at Perl.org* <http://www.perl.org/ >.

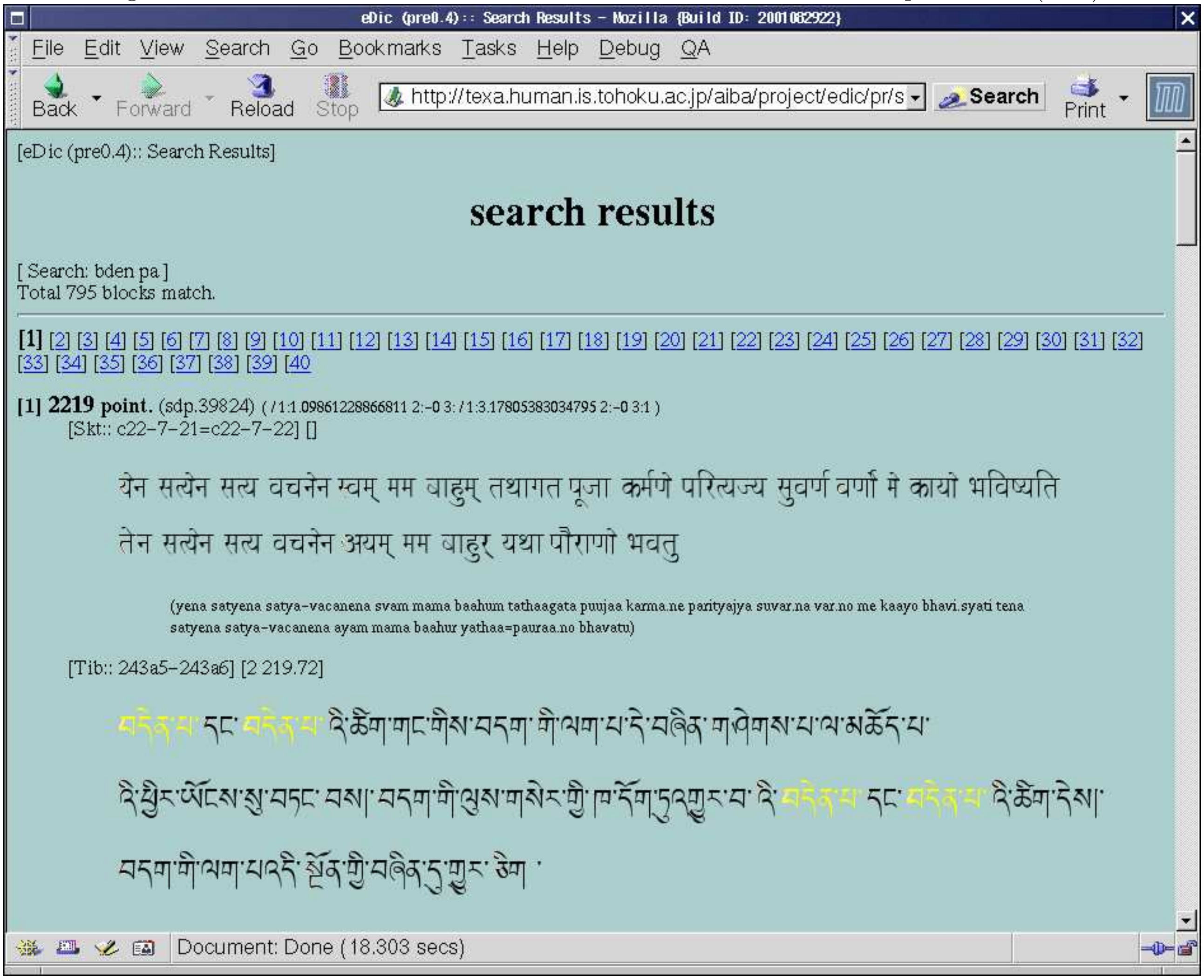
[6] *mod_perl* <http://perl.apache.org/ >.

[7] *ImageMagick.org* <http://www.imagemagick.org/ >.

[8] *the FreeType Project* <http://www.freetype.org/ >.

[9] Tooru AIBA, *Table of Transliteration Schemes for Sanskrit and Tibetan* <http://texa.human.is.tohoku.ac.jp/aiba/codes/table/ >.

Figure 7: Search result in the Tibetan-Sanskrit Search for the Saddharmapu.n.darika (eDic)



[10] Jun SUGAWARA, Gendai UIGURUGO TO Computer, *Islam Sekai*, vol. 58 (2002) <http://www.uighur.jp/works/uigpc/uigpc.htm>.