EMERGING TECHNOLOGIES
Multilingual Computing

Robert Godwin-Jones
Virginia Commonwealth University

Language teachers, unless they teach ESL, often bemoan the use of English as the lingua franca of the Internet. The Web can be an invaluable source of authentic language use, but many Web sites world wide bypass native tongues in favor of what has become the universal Internet language. The issue is not just one of audience but also of the capability of computers to display and input text in a variety of languages. It is a software and hardware problem, but also one of world-wide standards. In this column we will look at current developments in multilingual computing -- in particular the rise of Unicode and the arrival of alternative computing devices for world languages such as India's "simputer."

Character Sets

A computer registers and records characters as a set of numbers in binary form. Historically, character data is stored in 8 bit chunks (a bit is either a 1 or a 0) known as a byte. Personal computers, as they evolved in the United States for English language speakers used a 7-bit character code known as ASCII (American Standard Code of Information Interchange) with one bit reserved for error checking. The 7-bit ASCII encoding encompasses 128 characters, the Latin alphabet (lower and upper case), numbers, punctuation, some symbols. This was used as the basis for larger 8-bit character sets with 256 characters (sometimes referred to as "extended ASCII") that include accented characters for West European languages. ASCII has been around since 1963 and was extended by the ISO (International Organization for Standardization) in 1967 to allow for use of character codes for non-Latin alphabet languages such as Arabic and Greek. Later, to satisfy the need for use of languages such as Russian and Hebrew, the standard called ISO 2022 was established, later expanded into ISO 8859-1 (often called "Latin-1") which is widely used today for the interchange of information across the Web in Western languages. Actually, Latin-1 is one of 10 character sets, all 8-bit, defined by ISO 8859; others target eastern European languages, Turkish, Hebrew, Greek, Icelandic, and Celtic. The variety of ISO 8859 encodings is evident in the multiple character encodings which can be set in contemporary Web browsers.

ASCII does a fine job for working in English, since that was what it was designed to do. Likewise, 8-bit ISO 8859 character sets are adequate for displaying most of the world's writing systems. But they are not capable of dealing with languages with many more characters such as Japanese or Chinese. What is needed for such languages is at minimum a 16 bit or two-byte system which can handle thousands of characters. Sixteen bit encoding was not used initially on personal computers not just because of monolingual shortsightedness but also for technical reasons -- early computers had very little memory and storage capacity. With the current capacity of personal computers one might ask why not simply adapt a 3-byte or even a 4-byte character set system which would supply a virtually limitless number of characters to be displayed, thus guaranteeing the encoding of any of the world's languages. The problem is that such encoding systems would tend to use many more resources than is necessary for the display of most linguistic data, thereby slowing down network transmission and making display problematic on smaller devices with less processing power and memory. Also, computer operating and networking systems were designed to handle 8-bit data chunks; keeping 8-bit systems helps transactions progress smoothly and avoids the necessity for universal system upgrades.

Unicode

The problem remains of how to encode all the ideographs for Asian languages as well as the alphabets for other writing systems. The solution which recently emerged is known as Unicode, which, for all practical purposes, is identical to the official ISO 10646 standard. Unicode is not the ideal solution some advocates initially envisioned, a limitless (32-bit) system in which the emphasis is on inclusiveness rather than efficiency. But, like other systems that have been modified to accomplish goals not originally intended (such as the World Wide Web), Unicode is being cleverly adapted to work within today's computing and networking environment and to provide a way to encode all human languages, past and present. It is a compromise, one which takes a roundabout way to accomplish its goal, but in the long run the wide adoption of Unicode promises considerable benefit to the language community, in particular to those interested in preserving and promoting indigenous languages.

Unicode, as so many other innovations in modern computing, grew in part out of research done at Xerox corporation. A multilingual word processor called ViewPoint was developed for use on Xerox's Star Workstation in the 1980's. It could process text not only in multiple Latin-based languages but in a variety of Asian languages as well. While the system was too expensive to be commercially successful, the Xerox multilingual character code generated considerable interest among computer specialists and linguists. Eventually an industry project was launched by a number of U.S. firms and called Unification Code or Unicode, the goal of which was to unify all the world's alphabets into a single, very large, character set. A Unicode Consortium was founded and a prototype developed in 1991, with continual development since then. Parallel efforts at ISO have thankfully been merged with the Unicode initiative.

The most recent version of Unicode is 3.2. The original goal was to use a single 16 bit encoding. However, this allows for just 65,536 distinctive characters, clearly not sufficient to represent all current and past human languages. To expand the capacity, 16 additional sets of 65,536 characters (called "supplementary planes") have been added with a total possible character set of about one million. While each character represents at most 4 bytes, the Unicode standard actually defines 3 encoding forms (UTF-8, UTF-16, and UTF-32) that allow the same character to be transmitted in a byte, word, or double work format (i.e., in 8,16, or 32 bit per code unit). One of the key advantages of how Unicode has been implemented is the fact that the first 256 characters are identical to those defined by ISO 8859-1 (Latin-1), which in turn means that the first 128 characters are identical to ASCII. This provides backwards compatibility with systems and applications which are not Unicode aware. Also, as Unicode grows, compatibility with earlier versions is guaranteed.

Before Unicode, 8-bit character encoding with its built-in limit of 256 characters could have the same character code number represent a different character in different alphabets. In Unicode, each character is assigned a unique number. This means the use of Unicode can eliminate character set variations across systems and fonts, avoiding the "alphabet soup" of multiple ISO 8858 ***8859? character sets. Theoretically a single, very large, font could contain all characters defined by the Unicode standard. In practice, Unicode fonts tend to focus on a single language or family of languages. Unicode is designed to work on any operating system, including hand-held devices. In fact, the Newton Messagepad was an early user of Unicode in its operating system. Some Unicode compatibility has been included on Microsoft Windows since Windows 95 and on Macintosh since MacOS 8.5. Windows XP (as well as NT 4 and Windows 2000) offer robust native support of Unicode. MacOS X offers somewhat lesser support of Unicode as do Linux and other Unix systems. Finding applications which support Unicode and enable true multilingual computing is not always easy, although there are currently a number of word processors and text editors with Unicode support.

The display of Unicode encoded text is dependant on the presence of the fonts needed to display the characters represented by the Unicode code sequences. The core fonts for Windows are all now Unicode based and many Microsoft applications such as MS Office come with additional Unicode fonts. A

ubiquitous Windows Unicode font is Arial Unicode MS (downloadable from Microsoft) which features 51,180 glyphs (i.e., characters), including among others Cyrillic, Greek, Armenian, Hebrew, Arabic, Thai, Bengali, Tibetan, Chinese, Korean, and Japanese. The down side of such a font is that at 23 MB, its use can slow down your system. There are also shareware Unicode fonts, such as Code2000 (from James Kass). It is very easy in Windows to change keyboard layouts to type in different languages, assuming the needed fonts are present, as outlined in Alan Woods' Unicode Resource pages. On the (classic) MacOS, most applications that include Unicode support require Apple's Language Kits. The new MacOS X offers better Unicode support, as discussed in a series of recent articles on Unicode and MacOS X in the Mac newsletter, Tidbits. Windows Unicode fonts can be used in MacOS X.

Unicode and the Web

Characters other than standard ASCII display on Web pages have traditionally been encoded in HTML through the use of so-called escape sequences, or as they are officially called, character entity references, so that an umlauted lower case *a* ( ), for example, is encoded as `&auml;` or its numerical equivalent (`&#228;`). While this system works fine for major West European languages, for which character escape codes were defined early on, this does not hold true for non-Roman alphabets. Although there have been display possibilities from the early days of the Web for character sets such as Cyrillic and Japanese, browser set-up is problematic with no possibility of the mix and match of languages with different alphabets on the same Web page (such as annotating Chinese text in French).

The situation has improved considerably since Unicode has arrived on the scene together with the HTML 4.0 standard. HTML 4.0 specifies Unicode 3.0 as the default character set. It also calls for support of RFC 2070 ("Internationalization of HTML") which stipulates support for right-to-left languages like Arabic and Hebrew. One should note that Microsoft's Internet Explorer has taken this one step further, with support of top-to-bottom script systems such as Mongolian. Virtually all recent Web browsers support the 8-bit encoding of Unicode known as UTF-8 and are also able to use more than one font to display a multilingual page. How does Unicode get squeezed into a 8-bit system? Through a complex algorithm that interprets pointers to characters contained in the upper ranges of the "extended ASCII" range of numerical values (129-256); the process is explained in detail by Roman Czyborra. The UTF-8 encoding is the magic which allows the existing Web interface to display Unicode.

Unicode characters can be incorporated into a Web page the same way as character entities by using the decimal numeric character reference preceded by an ampersand and hash and followed by a semicolon (i.e., `&#0228;` for). Clearly, this kind of manual editing is fine for testing purposes, but for real Unicode authoring a Unicode aware text editor or HTML authoring environment is needed. Of course, the end user needs to have the necessary Unicode fonts installed in order to view the page correctly. It is sometimes a good idea to use styles to specify preferred fonts, as in this example from Alan Wood:

```
<style type="text/css">
.thai {font-family:Cordia New,Ayuthaya,Tahoma,Arial Unicode MS;}
</style>
```

It is then possible to reference the Thai text as follows:

```
Latin text followed by <span class="thai" lang="th">Thai text</span>
and more Latin text.
```

It should be noted also that UTF-8 should always be specified as the character encoding system with text containing two or more non-Latin scripts. This can be done with a META tag as follows:

```
<meta http-equiv="content-type" content="text-html; charset=utf-8">
```

In addition to Alan Wood's pages, Dan Tobias has good information on using Unicode on the Web. The advantage of using Unicode for including multiple languages on the Web is illustrated by Andrew

Cunningham in his discussion of how the Maribyrnong Library (Australia) used Unicode to add Amharic and Somali to its Web site.

Multilingual Devices

While Unicode provides a necessary means to display the character sets of virtually any language, there still remain a number of issues that make multilingual computing difficult. This is particularly the case if one considers potential users outside developed countries, where fundamental issues like illiteracy, poverty, and lack of infrastructure (power, Internet, phone) combine to erect formidable barriers to users. Mark Warschauer discusses these issues in his upcoming book on *Technology and Social Inclusion* (in press, MIT Press), to which I am indebted for information in this section.

Several recent projects have been started to try to overcome these handicaps and to bring networked computing to unserved populations, such as the Morphy One from Japan or the Pengachu Project (MIT). The Brazilian government is supporting deployment of a stripped-down personal computer known as the "computador popular" or People's Computer, developed at the Federal University of Minas Gerais, which is to be made available for the equivalent of US $250 to $300. To save money on software, it uses the Linux operating system, rather than Microsoft Windows, and relies on flash chip memory rather than a hard drive. Initially the People's Computer will go to schools, libraries, and health centers.

The emphasis on community sharing of computing resources is also at the core of India's "simputer" ("simple inexpensive mobile computer"). The simputer also uses Linux but in a hand-held computer designed to be used even by those unable to read. Slated to cost about US $200, the simputer has a built in Web browser and text-to-speech program which is capable of reading back Web pages in Indian languages such as Hindi, Kannada, and Tamil. Input is through a stylus and a program called tapatap which works by having the user connect dots to form characters. One of the interesting innovations of the simputer is the use of a removable smart card which holds personal data for individuals, allowing a single device to be shared by many people. Several companies have begun simputer production, including PicoPeta and Encore Software. It is too early to know if the simputer will be a success. Given its low pricing, commercial viability will only be possible if a mass market is created.

The principal output of the simputer is designed to be in the forms of audio or graphics, thus bringing networked computing to village populations of India for whom traditional keyboard input and text display could be problematic. Interestingly there are also efforts underway in India to develop local software for working in Indian languages, such as the tools available from Chenai Kavigal. This is clearly a necessary development in other parts of the world whose languages are not currently included in mainstream software development. Mark Warschauer discusses a set of Hawaiian language Web tools (Leok) and points to the benefits such tools have brought in support of the Hawaiian language learning community. Indeed, the Internet by offering connectivity among geographically separated users of languages can offer a powerful medium for both initial learning and language maintenance for languages for which local speakers or classes are not available.

Resource List

Standards and Organizations

- International Organization for Standards
- World Wide Web Consortium
- Unicode Consortium
- A short overview of ISO/IEC 10646 and Unicode
- RFC 2070: Internationalization of HTML

- ASCII chart
- Character Entity List (HTML 4)

Character Sets

- W3 Consortium's pages on interanationalization
- Coding the World's Writing -- from the "Babel" site
- Multilingual Computing: Introduction -- from Middlebury College
- Multilingual Computing and the Problem of Character Encoding and Rendering -- annotated guide
- Text and Fonts in a Multi-lingual Cross-platform World -- from the Digital Odyssey (UCLA)
- A Brief History of Character Codes in North America, Europe, and East Asia -- by Steven J. Searle
- ASCII: American Standard Code for Information Infiltration -- by Tom Jennings
- ISO-8859 briefing and resources
- Omniglot -- a guide to writing systems
- The ISO 8859 Alphabet Soup -- good explanation by Roman Czyborra
- Open Language Archives Community
- Ethnologue -- database of over 7000 languages
- The ISO Latin 1 character repertoire - a Description with usage notes

Unicode

- Why do we need Unicode?
- A short overview of ISO/IEC 10646 and Unicode -- by Olle Jrnefors
- Character Encodings Concepts -- good overview
- What document(s) define(d) the Unicode standard? -- good overview of successive versions of Unicode (by Roman Czyborra)
- The Unicode Standard
- Unicode Code charts -- glyphs and codes in PDF format
- Unicode and Multilingual Editors and Word Processors for Windows
- Finding Fonts for Internationalization FAQ
- Keyboard Layouts -- includes Arabic, Armenian, Azerbaijani, Ethiopic, Kazakh keyboards
- Free Recode -- program to perform code conversions
- Tutorial on character code issues -- by Jukka Korpela
- Unicode and Multilingual Web Browsers -- up-to-date list from Alan Wood
- Unicode and Multilingual Programs and Utilities -- information on word processing in Unicode with links to programs
- Setting up Windows Internet Explorer 5, 5.5 and 6 for Multilingual and Unicode Support
- Unicode fonts for Windows computers -- from Alan Wood
- Unicode and MacOS X -- from Tidbits

Web

- Setting up Windows Netscape Browsers for Multilingual and Unicode Support
- Test pages for Unicode character ranges -- wide variety of languages represented
- UTF-8 Sampler
- Using national and special characters in HTML
- Notes on Internationalization -- older but still informative information on HTML
- On the use of some MS Windows characters in HTML
- English - the universal language on the Internet?
- Babel: Multilingual bookmark page -- page in various character encodings
- Babel: Towards communicating on the Internet in any language...
- Developing your Multilingual Web sites -- from Babel
- Unicode Transformation Formats: UTF-8 & Co. -- wealth of information on how Unicode is dumbed down to 8-bit encoding

Devices

- Low-Cost Computers for the People
- Simputer
- Info in Encore Software's Simputer
- PicoPeta -- another initial Simputer manufacturer
- Computador popular vai rodar Linux -- article in Portuguese on the "computador popular"
- Low-cost 'people's computers' target developing nations to get poor on-line
- Morphy One Project -- open hardware Palmtop PC from Japan
- Project Pengachu -- wireless Linux project (MIT)
- Chennai Kavigal -- Indian language computing
- Kualono -- information on Leok