

*Workshop on
Digital Libraries: Theory and Practice
March, 2003
DRTC, Bangalore*

Paper: I

Creation of Digital Libraries in Indian Languages Using Unicode

Dr. ARD Prasad

Associate Professor

Documentation Research and Training Centre

Indian Statistical Institute, Bangalore

e-mail: ard@isibang.ac.in

ardprasad@hotmail.com

Abstract

Unicode is 16-bit code for character representation in a computer. Unicode is designed by Unicode consortium. It represents almost all the world script including extinct many of extinct scripts like Bramhi and Kharosthi. ISCII is another code developed for to represent the Indian characters in computer but there are problems with character representation using ISCII. It is found that Unicode can solve the problem. This paper suggests measures for creation of Digital library in Indic languages and the problem associated with Unicode.

1. INTRODUCTION

The main corpus of information is in written form, though one can not ignore other media of information such as symbols, signs, pictures etc. With the advent of Multimedia, much of the attention is paid to displaying or playing multimedia files. Many a format came into existence claiming better performance over its predecessors. However, when it comes to textual data, for a long time, there was no attempt to substitute ASCII for character representation. English has become a dominant language for communication of information and users were complacent with the 8-bit ASCII. Though an 8-bit approach to represent characters has been in use for scripts other than the Roman script, it became clear that one can not have a multi-lingual approach to represent various characters at a given time in a single screen or window. The reason is simple, in an 8-bit (1 byte), we can represent a maximum of 256 characters, whereas to represent characters belonging to various scripts require much more.

India is a country with rich diversity in languages, cultures, customs and religions. The number of languages listed for India is 418. Of those, 407 are living languages and 11 are extinct. Eighteen are constitutionally recognized languages written in a variety of scripts. These are Hindi, Marathi, Gujarati, Punjabi, Bengali, Assamese, Manipuri, Nepali, Oriya, Telugu, Tamil, Malayalam, Kannada, Konkani, Sanskrit, Urdu, Kashmiri and Sindhi.

Many of the Northern Indian languages belong to the family of Indo-European languages. The Southern Indian languages belong to the family of Dravidian languages. English, Persian, Sanskrit, Hindi are related, whereas Kannada, Tamil and Telugu are not related to Sanskrit or Hindi. Although, Indian languages do not belong to the same language family, the scripts of all the Indian languages are derived from Brahmi. The Indian scripts are highly phonetic and there exists greater uniformity in the arrangement of alphabets across all the scripts. The set of vowels and consonants in all the scripts is more or less the same.

It is paradoxical that one third of the total software personnel in the world are Indians, yet Indian languages hardly make a presence on the Internet. Many Indian languages are older than English with rich literature. It is true that many enthusiastic Indians in the world created web sites to provide information about the Indian customs, cuisines, and culture but these sites are mostly in English. It is high time that we should make serious attempts in content creation in Indian languages. Let us look at the existing options and solutions to the problem of content creation in Indian languages and scripts.

2. ASCII

American Standard Code for Information Interchange is a seven-bit code proposed by American National Standards Institute (ANSI) in 1963 and approved in 1968. However, the present systems use a byte (8 bits) for a character, the 8th bit is used for the so-called extended ASCII. In other words, in 7 bits one can represent 128 values, whereas by using 8 bits, one can represent 256 values (i.e. 0-255).

We know that the basic storage unit in digital computers is a bit (binary digit i.e. either 1 or 0) and 8-bits constitute a byte. If all the bits in a byte have the value 1, the total value of the byte is 255 and if all the bits contain 0's the value is 0. Every value in an ASCII byte is assigned a character or key of the computer keyboard. Thus the English letter 'A' is assigned the decimal value 65 i.e. 01000001, so is the letter 'B' which has decimal value 66 and is represented in binary form as 01000010. In addition, to the various keys of the keyboard, ASCII also contains some control characters (non-printable characters) like carriage return, page break, form feed etc. The following table presents the list of ASCII value and the corresponding characters.

ASCII value	Character	Control character	ASCII value	Character	ASCII value	Character	ASCII value	Character
000	(null)	NUL	032	(space)	064	@	096	
001	☺	SOH	033	!	065	A	097	a
002	☻	STX	034	"	066	B	098	b
003	♥	ETX	035	#	067	C	099	c
004	♦	EOT	036	\$	068	D	100	d
005	♣	ENQ	037	%	069	E	101	e
006	♠	ACK	038	&	070	F	102	f
007	(beep)	BEL	039	'	071	G	103	g
008	■	BS	040	(072	H	104	h
009	(tab)	HT	041)	073	I	105	i
010	(line feed)	LF	042	*	074	J	106	j
011	(home)	VT	043	+	075	K	107	k
012	(form feed)	FF	044	,	076	L	108	l
013	(carriage return)	CR	045	-	077	M	109	m
014	♪	SO	046	.	078	N	110	n
015	☼	SI	047	/	079	O	111	o
016	▲	DLE	048	0	080	P	112	p
017	▼	DC1	049	1	081	Q	113	q
018	↕	DC2	050	2	082	R	114	r
019		DC3	051	3	083	S	115	s
020	π	DC4	052	4	084	T	116	t
021	§	NAK	053	5	085	U	117	u
022	☐	SYN	054	6	086	V	118	v
023	↑	ETB	055	7	087	W	119	w
024	↕	CAN	056	8	088	X	120	x
025	↕	EM	057	9	089	Y	121	y
026	→	SUB	058	:	090	Z	122	z
027	←	ESC	059	;	091	[123	{
028	(cursor right)	FS	060	<	092	\	124	
029	(cursor left)	GS	061	=	093]	125	}
030	(cursor up)	RS	062	>	094	^	126	~
031	(cursor down)	US	063	?	095	_	127	☐

Copyright 1998, Jim Price, Com Copyright 1982, Leading Edge Computer Products, Inc.

Fig.1: ASCII Table

One can test these codes with the use of ALT key and the numeric pad. That can be done by holding the 'ALT' key, enter the decimal value corresponding to the character you wish to enter. If one wants to print character 'A' on the screen, press ALT and enter '6' and '5' on the numeric key pad. It demonstrates the ASCII table and it also helps users if any of one of the keys on the keyboard is not functioning because of dust.

IBM still uses a character code called EBCDIC on its mainframes. This code was originally adopted from punch cards in the 1960's. The informational professionals using CDS/ISIS have come across this code. In the 'import' and 'export' options of CDS/ISIS, there is provision for converting from EBCDIC to ASCII using 'gizmo conversion' option. However, overwhelming majority of computer systems use ASCII, thereby one can exchange data from one system to another without using any conversion software.

One of the greatest advantages of ASCII character set is that it is universally adopted. The files using ASCII character set are very versatile. These plain text files in ASCII can be easily read by word processors, data base management systems, spread sheets, etc. The source code of all most all the programming languages is written in ASCII. In a way, ASCII is the lingua franca of computers.

However, the greatest disadvantage with ASCII is that it uses only a byte. It becomes impossible to represent more than one script. In other words, a multi-lingual approach is not

possible with ASCII. Many countries have been using a single byte code to represent their own character set. It should be noted that ASCII is only a code for characters, however, what character should appear for each value in a character code set is the task of the operating system and the application programs.

3. ISCII

When it comes to Indian languages, ASCII may not serve the purpose for the simple reason that India is a country with many languages, although only a few have so called official recognition.

To overcome this problem, the BIS (Bureau of Indian Standards, formerly called the Indian Standards Institution) has adopted Indian Standard Code for Information Interchange (ISCII) in 1991. C-DAC, a member of the BIS committee has developed GIST (Graphics and Intelligence based Script Technology) card as a solution for Indian languages. Many software products like word processors, browser plug-ins were developed using GIST card. Another related standard to ISCII is the Indian Standard FOnT code (ISFOC) evolved for representing Indian scripts in GUI.

One major problem with GIST cards is that it uses 1 byte representation of characters. As only a maximum of 256 values can be stored, GIST card can handle only one Indian script at a time. Gist is compatible with ASCII, this is achieved by assigning the extended ASCII values (values beyond 128) to one Indian script. Therefore, GIST can display simultaneously in English and one of the Indian languages, though it cannot display two Indian languages at a time in a given screen. GIST card displays each Indian script through a hardware switch. By pressing a control character, the screen mode changes to a particular Indian script. GIST card in a way is insufficient to a multi-lingual approach. If one wants to have a bilingual or trilingual or multi-lingual dictionary of Indian languages, GIST falls short of the requirements. The problem lies with the 8-bit code of ISCII.

4. UNICODE

The Unicode Consortium was incorporated in January 1991, under the name Unicode, Inc., to promote the Unicode Standard as an international encoding system for information interchange, to aid in its implementation, and to maintain quality control over future revisions.

The primary goal of the development effort for the Unicode Standard was to remedy two serious problems common to most multilingual computer programs. The first problem was the overloading of the font mechanism when encoding characters. Fonts have often been indiscriminately mapped to the same set of bytes. For example, the bytes 0x00 to 0xFF are often used for both characters and dingbats. The second major problem was the use of multiple, inconsistent character codes because of conflicting national and industry character standards.

The Unicode Standard was designed to be:

- **Universal:** The repertoire must be large enough to encompass all characters that are likely to be used in general text interchange, including those in major international, national, and industry character sets.
- **Efficient:** Plain text is simple to parse: software does not have to maintain state or look for special escape sequences, and character synchronization from any point in a character stream is quick and unambiguous.

- **Uniform:** A fixed character code allows for efficient sorting, searching, display, and editing of text.
- **Unambiguous:** Any given 16-bit value always represents the same character.

Unicode provides two encoding forms:

- UTF-16, a 16 bit code (default)
- UTF-8, a byte oriented approach

With a 16-bit code one can represent 65,000 characters. However, by using escape codes of ISO/IEF 10646 (surrogate region in UNICODE) one can extend the capability to 1 million characters.

The present version is Unicode 3.0 covers 49,194 characters of all the scripts in the world and many other symbols, which includes

- All the European scripts
- All the middle Eastern scripts (that are written from right to left like Hebrew, Persian etc.)
- All Indic scripts
- All other Asian scripts which include the so called unified Han subset of 27,484 ideographic characters covering Chinese, Korean, Vietnamese, etc
- Ethiopic, Canadian Aboriginal Syllabics, Cherokee, Sinhala, Syriac, Myanmar, Khmer, Mongolian, Braille, and additional ideographs
- Punctuation Marks
- Mathematical symbols
- Technical symbols
- Geometrical shapes
- Dingbats (typographical symbol or ornament such as ¶, or <ironcross>)
- Even obsolete or historic scripts like brahmi, kharosti

In other words, Unicode covers alphabetic characters of various scripts including scripts in ideographic characters symbols including mathematical symbols.

Unicode does not include absolutely private scripts, font variants, logos, graphologies such as dance scripts. However, if one wishes, 6,400 code values in the basic 16-bit encoding are reserved for the Private Use Area.

Unicode is fully compatible with:

- ASCII
- ISO/IEC 10646

A code should take into consideration:

- Case (upper case and lower as in Roman script)
- Directionality (right to left also as in Hebrew and Persian)
- Alphabetic properties (combination of character as in many Indian scripts)

However, Unicode should be able to use sorting algorithms within a script. Though it has no meaning when sorting across languages. If we sort a set of multi-lingual words, works in

English may appear before the words in Hindi and it does not mean that it is the universal sorting order.

4.1. Problems with Unicode

Unicode should be supported by

- Operating systems
- Programming languages
- Application software
- Word processors

Although the present operating systems support a partial multi-lingual approach, they do not support all the scripts supported by Unicode. Windows NT uses the 16-bit Unicode character set, which includes scripts in use by major languages. Windows 95 still uses the 8-bit ANSI character set. Even the latest version Windows ME does not support all the scripts supported by Unicode.

The task of the computer text handling comprises of processing and encoding. When the user presses the key 'A', the operating system or the application software is responsible to display the character 'A' and storing the ASCII value in a file. In a GUI (Graphical User Interface – pronounced as GOO-ee) environment, however, the font 'A' can appear in various font sizes (like 10,12, etc) and font type (like Arial, Times Roman, etc).

The Unicode Standard does not define glyph images. That is, the standard defines how characters are interpreted, not how glyphs are rendered. Ultimately, the software or hardware-rendering engine of a computer is responsible for the appearance of the characters on the screen. The Unicode Standard does not specify the size, shape, or orientation of on-screen characters. The characters of various scripts in Windows95/98/ME/2000 environments are normally defined using various '.tff' (true type fonts) files. One can check the availability of these '.tff', in control panel under font folders. Unless these fonts of a specific script is available it is difficult to display them in the operating system. Though multi-lingual word processors like 'Global writer', 'Mithi' use their own fonts, it is difficult for other client software to make use of them.

4.2. Unicode and Web Browsers

Ideally the Internet should contain information in various scripts. Presently the popular web browsers like Internet Explorer and Netscape Navigator support various scripts, it is only partially. Again the problem is rendering combination characters. Irrespective of the Unicode, one can host and access a web site using the following two methods, though they are not ideal solutions.

- One can prepare a manuscript in the script he desires to display the information and scan it as an image and embed it in the HTML page of the web document. The problem with this approach as the information is in text form, it takes longer time for the end-user to get the information on his browser. The problem for the web host will be that it becomes extremely difficult and unwieldy to modify or update the information, as he has to prepare the manuscript and scan it again.
- Another solution is to provide a plug-in to the browser, which can display the fonts of particular script. This is the case with 'Jagran' plug-in, which enables users to get information displayed in Devanagari scripts. Similarly, one can get Telugu script displayed using the 'Pothana' plug-in. However, the disadvantage is that the user has to download the plug-in, which may require a little more extra skill on the part of the

user. Another disadvantage is that even the websites having information in Devanagari may not be using the same.

- As the browsers borrow the fonts from the operating system, unless the operating system supports the fonts of various scripts, the browsers can not display information in various languages.

The HTML supports Unicode. One can enter the Unicode values to get the desired characters displayed in web pages. For example, from the Figure 2, a Unicode table for Devanagari, the decimal value of Devanagari alphabet 'AA' is 2310. This letter can be entered in HTML page as 'आ' or if one wishes to enter the hex code, one can enter it as '�x0906'. However, the main problem is displaying combination letters like 'pra' in 'prasad'. The letter 'pra' is made up of 'pa' and 'ra'. Ideally, if one wishes to display 'pra' in Devanagari, one should enter the code for 'pa' and 'ra' without any spaces in between them as, 'पर'. If there is a space between these codes, the result will be two distinct letters 'pa' and 'ra'. But unfortunately, in Windows 9x/ME operating system in both the cases i.e. with space or without space between the codes the web page displays two distinct letters, as it fails to combine them.

Another problem is the logical display of the combination letters. For example, if one wishes to display the Devanagari 'ki' and 'kee', one should enter the Unicode for 'ka' and vowel 'e' to display 'ki', whereas 'ka' and 'ee' to display 'kee' in the same sequence. However, at present though we get the expected result with regard to 'kee', we do not get the desired result in case of 'ki'. Though it is meaningless, one should enter 'i' first, followed by 'ka', which is not a logical sequence. This is obviously the case with directionality. The problem becomes even more serious when one deals with scripts written from right to left. The only solution is to wait for the future releases of the operating system to support all the scripts.

5. CONCLUSION

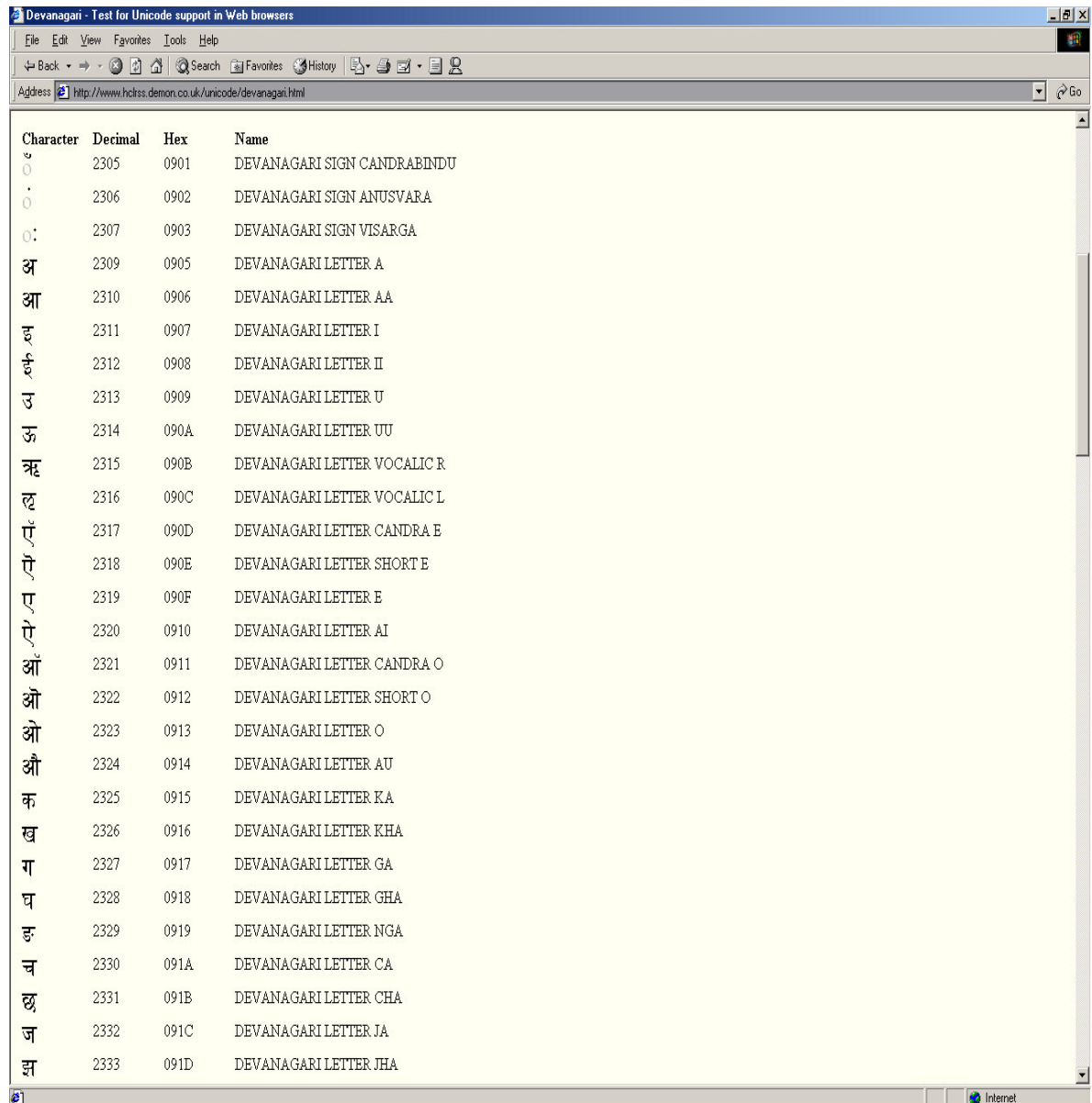
As information is present in various languages and scripts, the importance of Unicode cannot be overemphasized. Library professionals have been handling information in various languages and when the information is to be digitized, Unicode becomes essential and handy.

It is imperative that operating systems, web browsers, word processors, database management systems and a host of application software should support Unicode for the realization of digital libraries.

Most of the full text documents on the web are presented either in HTML (less likely), PDF and *PostScript* form. That is the browsers, Acrobat Reader and software like *GhostView* (for post script files) should be Unicode enabled.

Though one can enter Unicode values in HTML pages, it soon becomes difficult to enter a large text. If the HTML editors support Unicode with virtual keyboard for various scripts, the task of creating web content becomes easier.

In case of cataloging and Machine Readable catalogs, it becomes absolutely necessary to provide titles, transliterated titles and translated titles. Unicode will definitely be useful in automatically generating transliterated titles. The transliteration algorithms are easier to develop in Unicode. One of the research scholars at DRTC is working on developing a multi-lingual MARC using Unicode. Obviously the Unicode based ISO-2709 format becomes necessary for bibliographic data exchange.



Character	Decimal	Hex	Name
ॐ	2305	0901	DEVANAGARI SIGN CANDRABINDU
◌ं	2306	0902	DEVANAGARI SIGN ANUSVARA
◌ः	2307	0903	DEVANAGARI SIGN VISARGA
अ	2309	0905	DEVANAGARI LETTER A
आ	2310	0906	DEVANAGARI LETTER AA
इ	2311	0907	DEVANAGARI LETTER I
ई	2312	0908	DEVANAGARI LETTER II
उ	2313	0909	DEVANAGARI LETTER U
ऊ	2314	090A	DEVANAGARI LETTER UU
ऋ	2315	090B	DEVANAGARI LETTER VOCALIC R
ॠ	2316	090C	DEVANAGARI LETTER VOCALIC L
एँ	2317	090D	DEVANAGARI LETTER CANDRA E
ऐँ	2318	090E	DEVANAGARI LETTER SHORT E
ए	2319	090F	DEVANAGARI LETTER E
ऐ	2320	0910	DEVANAGARI LETTER AI
औँ	2321	0911	DEVANAGARI LETTER CANDRA O
ओँ	2322	0912	DEVANAGARI LETTER SHORT O
ओ	2323	0913	DEVANAGARI LETTER O
औ	2324	0914	DEVANAGARI LETTER AU
क	2325	0915	DEVANAGARI LETTER KA
ख	2326	0916	DEVANAGARI LETTER KHA
ग	2327	0917	DEVANAGARI LETTER GA
घ	2328	0918	DEVANAGARI LETTER GH A
ङ	2329	0919	DEVANAGARI LETTER NG A
च	2330	091A	DEVANAGARI LETTER CA
छ	2331	091B	DEVANAGARI LETTER CH A
ज	2332	091C	DEVANAGARI LETTER JA
झ	2333	091D	DEVANAGARI LETTER JHA

Fig. 2: Table of Unicode for Devanagari script.

As much of the bibliographic information is handles by Database management systems, it is ideal that DBMS packages support Unicode preferably the 16-bit version of Unicode.

Cross-language approach in searching is essential if one wishes to enter a keyword in a language and script known to him. Whether it is a database search or the Internet search engines, one should be able to search in any script and any language.

It is a well recognized that digital libraries should be able to handle multimedia files. However, without multi-lingual capability, the digital libraries can only be partial. It becomes imperative that the content of the Internet and the digital libraries should be in Unicode as early as possible. Though Unicode is compatible with ASCII and one can develop software to convert ASCII files to UNICODE, the volume of information and data to be converted to

Unicode becomes huge. Before we get into a problem similar to that of Y2K (if we are not already deep in that problem), the content of digital libraries and the Internet should adopt Unicode.

India has long history and one of the oldest civilizations in the world with an awesome diversity of languages. Instead of imposing one language and one script, ideally all the languages and scripts are to be given importance, as each language has produced rich literature and scientific information over the centuries. The Internet and Unicode give a wonderful opportunity to various government and non-government organizations and to individuals to publish information in any of the Indian languages.

6. REFERENCES

1. *LANGUAGES and scripts of India.*
from <http://www.cs.colostate.edu/~malaiya/scripts.html>
2. *TEST for Unicode support in web browsers: Devanagari.*
from <http://www.hclrss.demon.co.uk/unicode/devanagari.html>
3. SALOMON, R. *On the origin of the early Indian scripts: a review article.*
from <http://www.ucl.ac.uk/~ucgadkw/position/salomon.html>
4. *THE Unicode standard, version 3.0.* (2000). Massachusetts: Addison Wesley.