Multilingual Text Processing in a Two-Byte Code

Lloyd B. Anderson
Ecological Linguistics
316 "A" St. S. E.
Washington, D. C., 20003

ABSTRACT

National and international standards committees are now discussing a two-byte code for multilingual information processing. This provides for 65,536 separate character and control codes, enough to make permanent code assignments for all the characters of all national alphabets of the world, and also to include Chinese/Japanese characters.

This paper discusses the kinds of flexibility required to handle both Roman and non-Roman alphabets. It is crucial to separate information units (codes) from graphic forms, to maximize processing power.

Comparing alphabets around the world, we find that the graphic devices (letters, digraphs, accent marks, punctuation, spacing, etc.) represent a very limited number of information units. It is possible to arrange alphabet codes to provide transliteration equivalence, the best of three solutions compared as a framework for code assignments.

Information vs. Form. In developing proposals for codes in information processing, the most important decisions are the choices of what to code. In a proposal for a multilingual two-byte code, Xerox Corporation has made explicit a principle which we can state precisely as follows:

> Basic codes stand for independently functioning information units (not for visual forms)

The choice of type font, presence or absence of serifs, and variations like boldface, italics or underlining, are matters of form. Such choices are normally made once for spans at least as long as one word. We do not use ComPLeX mIXturEs, but consistent strings like this, THIS, this, or THIS. By assigning the same basic code to variations of a single letter (as a, a, A, A), all variants will automatically be alphabetized the same way, which is as it should be. The choice of variant forms is specified by supplementary "looks" information. (The capitalization of first letters of sentences, proper names, or nouns, is a kind of punctuation.)

Identical graphic forms may also be assigned more than one code because they are distinct units in information processing. Thus the letter form "C" is used in the Russian alphabet to represent the sound /s/, but it is not the same information unit as English "C", so it has a distinct code. So far this seems relatively obvious.

The same principle is now being applied in much more subtle cases. Thus the minus sign and the hyphen are assigned distinct codes in recent proposals because they are completely distinct information units. There are even two kinds of hyphens distinguished, a "hard" hyphen as in the word father-in-law, which remains always present, and a "soft" hyphen which is used only to divide a word at the end of a line, and which should automatically vanish when, in word-processing, the same word comes to stand undivided within the line.

We can now frame the question "what to code?" as a matter of empirical discovery: what are the independently functioning information units in text? Relevant facts emerge from comparing a range of different alphabets.

What is a "letter of the alphabet"? -- the problem of diacritics and digraphs. The most obvious question turns out to be the most difficult of all. Western European alphabets are in many ways not typical of alphabets of the world. They have an unusually small number of basic letters, and to represent a larger number of sounds they use digraphs like English sh, ch, th, or diacritics as in Czech š, č. It seems at first entirely obvious that digraphs like sh should be coded simply as a sequence of two codes, one for s plus one for h. Indeed English, French, German and Scandinavian alphabets do alphabetize their digraphs just like a sequence, s plus h etc. But these national alphabets are not typical. Spanish, Hungarian, Polish, Croatian and Albanian treat their native digraphs as single letters for purposes of alphabetical order. Spanish ll is not a sequence of two l's, but a new letter which follows all lo, lu sequences; similarly ch follows all c sequences, & ñ follows all n sequences as a separate letter.

There is just as much variation in handling letters with diacritics. The umlauted letter ö is alphabetized as a separate letter following o in Hungarian, and at the end of the alphabet in Swedish, but in German it is mixed in with o. In Spanish, ñ is treated as a separate letter, but the Slovak ň representing the same sound is mixed in with ordinary n.

In Table 1., the digraphs and letters with diacritics which are not in parentheses or brackets are alphabetized separately as distinct single units. Those in parentheses are alphabetized as a sequence of two or more letters or (Slovak and Czech ľ, ň, ť, ď) are treated as equivalent to the simpler letter, completely disregarding the diacritic. Combinations in brackets are used to represent sounds in words borrowed from other languages. Double dashes mark sounds for which an particular alphabet has no distinctive written symbol. (In Russian, palatal consonants are marked by choice of special vowel letters, while Turkish has a different kind of contrast, hence the blanks.)

Even when a digraph or trigraph is treated as a sequence of letters for alphabetization, there may be other evidence that it functions as a single information unit. In syllable division (hyphenation), English never divides the digraphs sh, ch, or th when they function as single units (heath-er, fa-ther) but does when they represent two units (hot-house). The same is true of other letter combinations in all national standard alphabets where a single sound is represented by a combination of letters.

Within certain mechanical constraints, typewriter keyboards also put each distinct information unit on a separate key. Thus Spanish ñ or Czech š, ž, č are produced by single keys, not by adding a diacritic to a base letter. Mechanical limits have forced a sequence of two letters (like the Spanish ch, ll) to be typed with two separate keystrokes whether or not they represent a single functional unit, but occasionally we see exceptions, as in Dutch where the ij digraph appears as a ligature on a single key and is printed in one space not two.

Unit unanalyzable letters exist in Serbian and Macedonian for most of the sound types (the columns) of Table 1. Icelandic has single letters "thorn" and "edh" for the two rightmost columns. Even where the other languages use digraphs or letters with diacritics, there is evidence from syllabification and usually also from alphabetical order that these are functionally independent information units. For transliteration from one national alphabet into another, these symbol equivalences are needed. The principle stated on the preceding page thus implies that unique codes be available for English sh, ch, th and unitary digraphs in other languages so these can be used when needed in information processing. (Information processing is not the shuffling of bits of scribal ink!) The principle does not compel use of those codes -- English th can be recorded first as a sequence of two codes, then converted into a single code only when needed, by a program which has a dictionary listing all words containing unitary th.

Spatial arrangement of printed characters. In alphabets of Europe, letters (and information units) almost always follow each other in a line, from left to right. This is not true of many

Table 1. Some Consonant Characters in Europe

Sound	rś/ř	ĺ	ń	ť	ď	š	ž	č	ǧ	s	ḫ	ts	dz	θ	ð
Russian						ш	ж	ч	[дж]	с	х	ц	[дз]	--	--
Macedonian		љ	њ	ќ	ѓ	ш	ж	ч	џ	с	х	ц	ѕ	--	--
Serbian		љ	њ	Ћ/ћ	Ђ/ђ	ш	ж	ч	џ	с	х	ц	[дз]	--	--
Hungarian	--	ly	ny	ty	gy	s	zs	cs	[dzs]	sz	--	c	[dz]	--	--
Croatian	--	lj	nj	ć	đ	š	ž	č	dž	s	h	c	[dz]	--	--
Slovak	--	(ľ)	(ň)	(ť)	(ď)	š	ž	č	(dž)	s	ch	c	[dz]	--	--
Czech	ř			(ť)	(ď)	š	ž	č	(dž)	s	ch	c	[dz]	--	--
Latvian	ŗ	ļ	ņ	ķ	ģ	š	ž	č	(dž)	s	--	c	(dz)	--	--
Polish	--	l	ń (ni)	ć (ci)	(dź) (dzi)	(sz)	ż (rz)	(cz)	(dż)	s	(ch)	c	(dz)	--	--
German	--	--	--	--	--	(sch)	--	(tsch)	[dsch]	s	(ch)	z	[dz]	--	--
Albanian	--	lj	nj	q	gj	sh	zh	ç	xh	s	h	c	x	th	dh
Turkish						ş	j	ç	c	s	h	[ ]	[ ]	--	--
Romanian	--	(...)	(...)	--	--	ş	j	{(ci) (ce)	{(gi) (ge)	s	--	ţ	[ ]	--	--
French	--	(...)	{(...) (gn)	--	--	(ch)	j	[tch]	[dj]	{s ç	--	[ts]	[dz]	--	--
English	--	(...)	(...)	--	--	(sh)	(...)	(ch)	j	s	--	[ts]	[dz]	th	th
Spanish	--	ll	ñ	--	--	x	[ ]	ch	[ ]	s	j	[ts]	[dz]	--	--

important alphabets elsewhere in the world. Arabic and Hebrew, when they write short vowels, place them above or below the consonant letters. What we transcribe as *kitabu* appears (in a left-to-right transform of the Arabic arrangement) as shown on the right. These vowel symbols are independent information units, not "diacritics" in the sense of the European alphabets. They keep a constant form, combining freely with any consonant letter. Alphabets of India and Southeast Asia place vowels above, below, to right or to left of a consonant letter or cluster, or in two or three of these positions simultaneously. There can be further combinations with marks for tones or consonant-doubling.

```
  a u
k t b
i
```

The Korean alphabet arranges its letters in syllabic groups, so that *mascot* would be a shown to the right if written in the Korean manner. The independently functioning information units are still consonants and vowels, for which we need codes, and we need one additional code to mark the division between syllables. This is just as much an alphabet as our familiar English and is not a syllabary. (Since there are only about 400 syllables, a *printing* device might store all of them, but these would not normally be useful in information processing.)

```
m a   c o
  s     t
```

A flexible multi-lingual code for information processing must be able to handle the different spatial arrangements described here, but it need not (except in input and output for human use) be concerned with what that spatial arrangement is, only with what significant information units it contains. Even in Europe, Spanish accented vowels *á*, *é*, *í*, *ó*, *ú* show a vertical superimposition of the basic vowels with a functionally independent symbol of accentuation. These are not new letters in the sense that Croatian *š*, *ž*, *č* and *ć* are, but are alphabetized just like simple *a*, *e*, *i*, *o*, *u*.

Criteria for a two-byte code standard. We can now consider alternative methods of coding for multilingual information processing. Three basic criteria are given first, followed by discussion of alternative solutions and further criteria.

A) Each independent character or information unit shall have available a representation in a two-byte code (whether it is graphically manifest as a base letter, digraph, independent diacritic, letter-plus-diacritic unit, syllable separation, punctuation mark, or other unit of normal text, and independent of position in printing).

B) It shall be possible to identify the source alphabet from the codes themselves. [Since "C" in Czech represents the sound /ts/, it is not the same unit as English "c"; in library processing it is important to know that German *den* and *die* are articles like English *the*, to be disregarded in filing, but English *den* and *die* are headwords.]

C) The assignment of information units to codes shall maximize the possibilities for use of one-byte code reductions through long monolingual texts, minimizing shifts between different blocks of 256 codes. [This is especially important in reducing transmission costs.]

Each of the following three solutions has certain advantages. The third is far superior in the long run.

Solution 1. *Incorporate existing 7-bit or 8-bit national code standards*, one in each block of 256 codes. Use the extra space as codes for information units which are not single spacing characters, This satisfies all of the basic criteria (A,B,C) and uses existing codes, adding only a first byte as an alphabet name to make a two-byte code. There is no transliteration-equivalence and elaborate transliteration programs would be necessary for each conversion, *N* x *N* programs for *N* alphabets.

Solution 2. *Systematically code all basic letter forms and all their diacritic modifications* thus allowing for expansion, use of new letter-diacritic combinations. Despite their differences, Latin-based alphabets share a common core of alphabetical order, which can be reflected in a coding to minimize shuffling. This is attempted in Table 2., which includes all characters from ISO/TC97/SC2 N 1255 1982-11-01 pp.60-61 plus additions from African and Vietnamese alphabets. Code ordering is downwards within columns, starting from the left.

Table 2. Alphabetical order of letters and diacritics as a basis for coding

a	â	b	ɓ	c	d	ɗ	đ	ð	e	ɛ	f	ƒ	g	h	ħ	i	ı	ĳ	j	k	ƙ	l	ł	m	n	ŋ	o	œ	ơ	ɔ	p	q	r	s	β	t	ŧ	u	ů	ư	v	ʋ	w	x	y	ƴ	z	þ	å	æ	ø
				č	ď				ě													ľ			ň							ř	š		ť											ž			
á	ấ			ć					é				ǵ			í				k		ĺ			ń		ó		ớ				ŕ	ś				ú		ứ					ý		ź			
				ċ										ġ																																					ż			
				ç										ģ							ķ		ļ			ņ								ŗ	ş		ţ															
à	ầ								è							ì										ò		ờ									ù		ừ					ỳ						
ã									ẽ							ĩ									ñ		õ		ỡ									ũ		ữ											
â									ê							î										ô												û						ŵ		ŷ						
ā									ē							ī										ō												ū									ȳ						
ä									ë							ï										ö												ü									ÿ						
																													ő												ű														
ą									ę							į										ǫ												ų														
ă													ğ																																								
ả	ẩ								ẻ							ỉ										ỏ		ở									ủ		ử											
ạ									ẹ							ị										ọ		ợ									ụ		ự											

This solution satisfies none of the criteria (A,B,C), and does not provide codes for many kinds of information units. It appears to be economical in Europe, where 20 national alphabets can fit in 48 x 13 = 624 code cells if only letter forms are considered. But for non-Latin alphabets there can be no similar savings. Here there are (considering only living alphabets) about 55 alphabets based on 38 distinct sets of letters.

Solution 3. Transliteration-equivalent units assigned identical second bytes in their two-byte code. Transliteration between any two alphabets simply changes the first byte of the code naming the alphabet, requiring minor programming only when an alphabet has non-recoverable spellings or cannot represent certain sounds. This solution depends on the fact that there is a small number of types of information units which have ever been represented in a national standard alphabet. In the tentative arrangement of Table 3., most of the sound types noted are represented by single unanalyzable characters in some national alphabet (as Georgian, Armenian, Hindi, ...), and most of the rest by clearly unitary digraphs. Despite the strange symbols, this is not a list of fine phonetic distinctions, it is a list of distinct categories of written symbols.

The idea for this solution came from the one-byte code adopted in India, structured identically with transliteration-equivalence for each of the alphabets of India. A printer with only Tamil letters can simply print a Tamil transliteration of an incoming Hindi message.

In the two-byte version presented here, there is provision for any alphabet to add characters representing sounds of some other alphabet, and a small amount of space to add unique information units which are not matched in other alphabets. This is the right amount of space for expansion.

Applications to transliteration and library processing. With newer capabilities of printers and screens, a speaker of any language can soon request a data base in its original alphabet or in any transliteration of his choice, either one using many diacritic characters like Croatian and special symbols to avoid ambiguity, or one more adapted to his native alphabet, for example French or Hungarian. Records can be kept in the codes of the original alphabet, always ensuring complete recoverability. There would be a gentle encouragement for each national alphabet to use a consistent transliteration for each sound independent of the source alphabet, because this would be automatic.

Summary. The third solution described above is designed to handle all the structures and functions found in national standard alphabets and to fit them like a well-made glove, allowing the maximum capabilities of information processing, but never compelling their use. This type of solution could be a primary international standard, with code translations to reach existing 7-bit and 8-bit standards and an ESCAPE sequence to allow processing directly in the older standards (solution 1. above incorporated as an alternate). Since mathematical and scientific symbols are international, they would require only single blocks of 256 codes. The first column of 16 blocks of 256 each could provide 4096 two-byte control codes, and the second column could eventually be added to the 96 alphabet blocks allowing transliteration of numerals. The right 128 blocks of 256 codes each remain for Chinese/Japanese characters or other purposes, but even these can be coded alphabetically in terms of character components and arrangements (partly achieved in a keyboard now installed at Stanford and the Library of Congress).

ACKNOWLEDGEMENTS

I would like to thank Mr. Thomas N. Hastings, chairman of the ANSI X3L2 committee, and Mr. James Agenbroad, APO, Library of Congress, for indispensable information and discussions. They of course bear no responsibility for claims or analyses presented here.

Table 3. Transliteration-equivalent information units found in national standard alphabets

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Control Codes	Control codes or numbers and accent marks	SPace		k	q	ts/c	ć/ť	č	t̲š/cz	ţ	ṭ	t		p	kʷ
1			.	:	kʔ	qʔ	tsʔ/cʔ	ćʔ	čʔ		ţʔ		tʔ		pʔ	
2			,	;	kʰ	qˣ	tsʰ/cʰ	ćʰ	čʰ	t̲šʰ	ţʰ		tʰ		pʰ	hʷ
3			-	/	x	χ	s	ś	š	š̲/sz	ṣ	ṣ	θ	ɬ	Φ̄	f
4			¿	⁝	g		dz	ǵ/d́	ǧ	d̲ž/dż	ḍ	ḍ	d		b	gʷ
5			?	!	ɠ			ƴ	(y)		ɗ̣		ɗ		ɓ	(w)
6			"	'	gh				ǧh		ḍh		dh		bh	(ɾʷ)
7			"	'	ɾ	ƙ	z	ź	ž	ž̲/ż	ẓ	ẓ	ð̆	ɮ	β	v
8			(	[	h	ḥ	r	ŕ	ř	r̲	ṛ					
9			)	]	ʔ	ʕ		ĺ	(y)	l̲/l	ḷ	ḷ	l	l	ɫ	(w)
A			INitial-CAPS	SUPerscript	ŋ			ń		n̲	ṇ		n		m	(ŋʷ)
B			ALTern.-CHAR	DIACritic	aŋ	əŋ	a̧/an	ən	i̧/in	-am □̂	ų/un	□̆		ȩ		ǫ
C			SYLLable-SEPAR.	INSULator	ṛ	ḷ	a	(yσ)	i	(y)	u	(w)	e	æ	o	ɔ
D			REPeat	MARKER (Eng. e̲)	ṝ		ā	(ya)	ī	□b	ū	(yu)	ē	ǣ	ō	ɔ̄
E			DIGraph-LINK	SILent LETter	ü	ö	ɒ	œ	ɨ	(ɪ)	ư/ɯ	(ʊ)	ə	ɛ/ɛ̆	ơ/ʌ/ɣ	ɔ/ɔ̆
F			DOUBle CONSon.	NO VOWEL	ǖ	ȫ	ɒ̄	œ̄	ɨ̄	(ye)	ɯ̄	(yo)	ə̄	ai	ơ̄	au