Designing Code Tables with Application to Nepali

Pat Hall
The Open University, UK

An essential first step in the localisation of software is the proper encoding of the scripts of the new language. Drawing upon experience in Saudi Arabia in the 1970s and now in Nepal, I set out a number of principles to guide the production of code tables, and then analyse alternative proposals for Nepali. From this it is shown that existing code tables in Unicode have shortcomings and should be reviewed, while the proposals under discussion for Nepali should be given time to be worked through.

It is desirable for computers to work in the scripts and languages of the users of the computers. A large proportion of the world’s population does not speak English. Even in India where English is a common lingua franca, only 5% of the population speaks English sufficiently well to use computers effectively. For these communities to benefit from computers to the extent that western communities have, their computers must work in their own local languages.

Computers were developed in the west, mostly in the U.S., and thus were developed to interact with their users in American English. But as costs have plummeted and capabilities risen, computers have become widespread commodities so that more than half the sales of computers today are outside the U.S. (eg. Taylor, 1992; Hall and Hudson, 1997). This in turn has led to pressure for localised versions of software. Suppliers of hardware and software have responded to this demand, though not always with the willingness that these markets would deserve.

During the late 1970s computers began to be introduced into the Arab world in large numbers, and slowly the Arab users began to ask that they should work in Arabic. The systems were supplied almost entirely by companies from Europe and the United States, who sometimes responded that it was not possible, and sometimes rose to the challenge. Their first challenge was to make devices that had assumed writing was from left to right to operate right to left, while the second challenge was imaging a set of characters that required detail of a much higher resolution, like dots above or below the characters, not required for the Roman Alphabet except for a few diacritics. The alphabet to be input and output could be found in any text book on Arabic, and it was assumed that providing that those shapes could be typed in and printed, all would be well.

But it was not, and naive computer professionals made serious blunders—the most singular of these was done by a well known UK company operating in Kuwait who noted that there was a circular shape like an “O”, but a little more egg shaped, which they then gave a single code and a single place on the keyboard - in fact there were two different characters of Arabic involved, one being the number five (Khamza) and the other a special letter Teh Marbuta used at the end of words (strictly written with two dots above it which are often omitted).

This situation in Arabic computing led me and several colleagues to investigate the whole issue of how Arabic should be handled in the computer (Hall and Hussein 1978, Hall 1978, Hall and Waghorn 1979, Hall 1981 ) leading a number of us active in the informal Riyadh Computer Users Group to make a proposal for a set of Arabic codes which we offered to...
the Saudi Arabian Standards Association. In drawing up this proposal, we were influenced by an earlier proposal from Iraq which had decided that it was necessary only to code the letters and not the forms of the letters, an idea which had also been investigated in Lebanon. This led to a patent for a simpler form of typewriter in which the typist only selected the letter needed and not the form of letter, leading to significantly higher typing speeds. Both these proposals assumed an intelligence in the computer which could work out which form of letter to use following some simple rule. However, there were some problems, and the rules were not that simple for cases like Hamza (a glottal stop always written as a small diacritic but on a “carrier” which varied following rules determined grammatically).

Our ad hoc standards drafting committee built on this earlier work and implicitly used a set of design rules to determine the code tables of our proposal, but we did not feel it appropriate to publish those rules. Now, twenty years later, I am working in Nepal and find a situation not unlike that which I found in Saudi Arabia in 1978. There is a plethora of code tables, and an urgent need to bring coherence to these, before too much data gets stored and then needs conversion before it can be shared. In the process, we will have to undertake a systematic design of the code tables for Nepali, and those rules used implicitly for Arabic are going to be important.

The objective in this paper is to set out those rules, suitably updated and improved by knowledge acquired since. Then those rules will be applied to codes for Devanagari script, and in particular the use of the script for the Nepali language. Then, when at some later time, some new writing system must be supported by computer, these rules can be applied to determine what should be coded, and how it should be coded. It is also suggested that existing codes should be fully reviewed.

**Basic Concepts and Terms**

When we look at computers and their use in a particular language, what we see are the input devices, typically keyboards, and the output in the writing system of the language on the screen and on paper. It is very tempting to focus on these and ask what characters should be printed, and where should these appear on the keyboard for input. But the situation is much more subtle than this, and we will need to distinguish a number of concepts important in the area, but which may be confused in everyday discourse. We will go through these terms, turning both to the Concise Oxford Dictionary for general use and to the Penguin Dictionary of Language and Languages (Crystal, 1992) for linguistic usage.

First, we need to distinguish between a language and the way that it is written, using a script (the collection of graphic symbols known as graphemes or glyphs) and an orthography (which includes the script and the prescribed spelling and punctuation rules). The same script, or essentially the same script, may be used for many languages, as in the European languages most of which use the Roman script. These languages do not even need to be of the same general structure (linguistic group) as in the use of Devanagari (Brahmi) derived scripts used for languages of the Indo-Aryan group (where these scripts originated) but also for Tibeto-Burman languages (e.g., Tibetan and Thai) and Dravidian language (e.g., Tamil). Equally well, the same language may be written in many scripts, as has happened historically with Turkish which switched from the Arabic script to the Roman script in 1928 as part of the “modernisation” programme of Kemal Ataturk. It does happen in Japanese with the ideographic Kanji writing system and the Kana syllabaries Katakana and Hiragana and the proposed (Matuura, 1997) Romanisation of Japanese.

Scripts can be broadly of two types—concept and phonemic. Concept systems typically use ideographs, each ideograph representing some concept in the world of the users—typically words in other systems: the archetypal concept system is Chinese, where the written form is intelligible across most of China whereas the spoken form is not; pictogram systems are also concept systems, the most well known being the ancient Egyptian hieroglyphs from 3000B.C. “Concept system” is not a term used by linguists, though Crystal has the entry for ideogram as “A symbol used in a writing system to represent a whole word or concept,” but it does seem to be a useful term.

Phonemic systems concentrate on the spoken form of the language and aim to encode that form—or at least they originally had that intention. Concept systems lead to a very large set of ideographs, while phonemic systems are very much smaller.

Phonemic systems are to two types: alphabets in which letters represent single phonemes, or syllabaries in which the characters represent syllables, typically consonant vowel pairs. Alphabets are by far the most common, typified by the Roman and Arabic alphabets: the syllabary that is viewed as typical is the Kana system of Japan. Not all experts agree on where to place a script—so for example, the Devanagari system is claimed by some to be an alphabet in which the letters can take different forms depending upon their position in the word, so that vowels may be recorded as signs in the middle of words, and consonants may be represented as partial characters when used medially in conjuncts; while others view Devanagari as a syllabary in which the base characters are the composites of consonant plus vowel sign and the conjuncts. Typically syllabaries are significantly larger than alphabets, as will be appreciated from the combinatorics of the situation (if you have 20 consonants and 5 vowels, you have an alphabet of 25 letters, but a consonant-vowel syllabary of potentially 100).

**Writing Systems**

Sometimes the term “writing system” is used as a synonym for orthography (e.g., Crystal, 1992) but we will distinguish it and use writing system in a more comprehensive technical manner to encompass the processes of reading and writing as well as the script with its glyphs and graphemes, and the orthography prescribing rules of combination in sequence
Related Content

Demand for Number Portability in the Korean Telecommunications Market: Contingent Valuation Approach
[www.igi-global.com/article/demand-number-portability-korean-telecommunications/3648?camid=4v1a](www.igi-global.com/article/demand-number-portability-korean-telecommunications/3648?camid=4v1a)

Standardization and Other Coordination Mechanisms in Open Source Software
[www.igi-global.com/chapter/standardization-other-coordination-mechanisms-open/19028?camid=4v1a](www.igi-global.com/chapter/standardization-other-coordination-mechanisms-open/19028?camid=4v1a)

Internet-Based Spatial Decision Support Using Open Source Tools
[www.igi-global.com/chapter/internet-based-spatial-decision-support/19022?camid=4v1a](www.igi-global.com/chapter/internet-based-spatial-decision-support/19022?camid=4v1a)

Adaptive Strategies of Firms in High-Velocity Environments: The Case of B2B Electronic Marketplaces
[www.igi-global.com/article/adaptive-strategies-firms-high-velocity/3601?camid=4v1a](www.igi-global.com/article/adaptive-strategies-firms-high-velocity/3601?camid=4v1a)