A TRANSLITERATION ENGINE FOR ASIAN LANGUAGES

Sathiamoorthy Manoharan
Department of Computer Science, University of Auckland, Auckland, New Zealand

Keywords: Transliteration, e-Learning of ethnic languages, Internationalization, electronic document preparation.

Abstract: Transliteration maps the alphabets of one script using alphabets of another script. It is commonly used when proper nouns of one language need to be written in the script of another language. This typically involves constructing approximate phonetically equivalent words. For instance, the phonetic equivalent of New Zealand in Japanese is "niyuu jiilando" which in Katakana is ニユー ジーランド. This paper illustrates the design and development of a transliteration engine which suits syllabary and alphasyllabary scripts. A syllabary is an alphabet set that represent syllables. The Japanese Katakana and Hiragana scripts fall under this category. An alphasyllabary is an alphabet set that represent consonants, vowels, and syllables composed of consonants and vowels. Scripts such as Thai, Sinhala, Burmese, and most of the Indian scripts such as Devanagari, Tamil, and Malayalam come under this category. The engine is useful in teaching and learning ethnic scripts. It is a useful tool for the internationalization and localization of computer programs, publishing ethnic scripts over the Internet, and to compose electronic documents in ethnic scripts.

1 INTRODUCTION

Transliteration maps the alphabets of one script using alphabets of another script. This maps the alphabets or combination of alphabets from one script to the alphabets of the other. This involves constructing approximate phonetically equivalent words. For example, the phonetic equivalent of New Zealand in Japanese is "niyuu jiilando" which is ニユー ジーランド in Katakana.

Typical use of transliteration includes the following.

1. Teaching and learning of ethnic scripts. A student learning Japanese in New Zealand cannot easily access a Japanese keyboard to compose an essay in Japanese. Soft keyboards and keyboard mappings can be cumbersome to use. Transliteration allows typing the letter using the Roman alphabet and automatically mapping it to Japanese alphabet.
2. Small-scale electronic publishing. Language phrases can be easily composed using Roman alphabets. For example, foreign language phrases in this paper are composed using transliteration.
3. Internationalizing and localizing software products. For example, the caption for the home button of a web browser in an English version may read Home, but in a Japanese version of the application this needs to read ホーム. This can easily be generated by transliterating to "hoomu" which is how home is pronounced in Japanese.

This paper illustrates the design and development of a web-based transliteration engine that suits syllabary and alphasyllabary scripts. A syllabary is an alphabet set that represent syllables. The Japanese Katakana and Hiragana scripts fall under this category. An alphasyllabary is an alphabet set that represent consonants, vowels, and syllables composed of consonants and vowels. Scripts such as Thai, Sinhala, Burmese, and most of the Indian scripts such as Devanagari, Tamil, and Malayalam come under this category. Refer to the *Unicode Standard* (The Unicode Consortium, 2003) for a description of these alphabets.

The rest of the paper is organized as follows. Section 2 reviews some related work. Section 3 presents the design and development of the transliteration engine. Section 4 evaluates the transliteration engine and outlines its possible uses. The final section concludes with a summary.

Manoharan S. (2008).
A TRANSLITERATION ENGINE FOR ASIAN LANGUAGES.
In *Proceedings of the Fourth International Conference on Web Information Systems and Technologies*, pages 376-379
DOI: 10.5220/0001519903760379

2 RELATED WORK

Sakai, Kumano, and Manabe describe a transliteration system for automatic information retrieval (Sakai, Kumano, and Manabe, 2002). Their system is designed to support both transliteration and back-transliteration of Roman scripts to Japanese Katakana. They show that the system produces over 75% correct results when transliterating, and over 55% correct results when back-transliterating. One of the reasons why back-transliteration does not work as well as transliteration is the fact that certain sounds are not possible to represent in Katakana (for example, there is no distinction between “l” and “r”).

Kawtrakul et al propose a similar back-transliteration system for information retrieval (Kawtrakul et al, 1998). Their system transliterates Roman scripts to the alphasyllabary Thai script. The main use of their system is to be able to automatically generate foreign words (such as proper nouns and technical terms) in Thai scripts. The back-transliteration system they use consists of three steps: syllable formation, phonetic transcription, and a fuzzy search for a matching English word. Similar to the system of Sakai, Kumano, and Manabe, there are issues with back-transliteration, because some of the sounds in English do not exist in Thai.

Kang and Kim propose an English-to-Korean transliteration scheme (Kang and Kim, 2003). This scheme operates in three steps: constructing phonetic sequences that represent all possible transliterations of a given phrase; checking the validity of each of the transliterations; and choosing the most probable transliteration.

Grefenstette, Yan, and Evans describe a Katakana transliteration scheme based on finding matches on the Web (Grefenstette, Yan, and Evans, 2004). This is somewhat similar to the scheme of Kang and Kim in that both schemes look at a number of possible outcomes, and choose one based on a criterion. The criterion that Grefenstette, Yan, and Evans use is the highest hit score for a phrase on the Web.

Al-Onaizan and Knight propose a transliteration scheme from Arabic to English based on probabilistic finite state machines (Al-Onaizan and Knight, 2002). Their evaluation indicates transliteration accuracies from 15% to 55%, depending on whether the phrase to be transliterated is of Arabic, English, or other origin. The absence of short vowels in Arabic and the existence of silent letters in English (such as the P in Psychology) cause major transliteration inaccuracies.

3 TRANSLITERATION ENGINE

The goals of the transliteration engine are threefold: to be able to produce text phrases that can be pasted onto electronic documents; to be able to produce XHTML Unicode strings that can be used for publishing on the Internet; and to be able to produce C strings that can be used in C-like programming languages.

The transliteration script mappings in the transliteration engine use the Hepburn Romanization system. This system was originally developed in 1867 by Reverend James Hepburn to transcribe Japanese words into Roman alphabets. There are some variants of the system. The system used in the transliteration engine is called the modified Hepburn system where long vowels are indicated by doubling the vowel. Table 1 illustrates a partial script mapping for Japanese Katakana using the modified Hepburn system.

Table 1: A partial script mapping for Katakana.

Source string	Target string	Unicode symbol
b	\u30C4	ツ
ba	\u30D0	バ
baa	\u30D0\u30FC	バー
be	\u30D9	ベ
bee	\u30D9\u30FC	ベー
bi	\u30D3	ビ
bii	\u30D3\u30FC	ビー
bo	\u30DC	ボ
boo	\u30DC\u30FC	ボー
bu	\u30D6	ブ
buu	\u30D6\u30FC	ブー
bya	\u30D3\u30E3	ビャ
byo	\u30D3\u30E5	ビュ
byu	\u30D3\u30E7	ビョ

The source for the script mapping is a set of words in Roman alphabet and the target is a set of words in Unicode. This makes it possible to configure the transliteration engine for a variety of scripts. The engine processes the input text by looking for the longest match of source letter sequences within the script mapping.

3.1 Script Mapping

There are challenging issues to consider in formulating the script mapping.

A fundamental requirement for using transliteration is familiarity with the target language. It is important that the user knows how a given word is pronounced in the target language, and use this

knowledge to carefully construct the source language word. For example, the Japanese word "konichiwa" needs to be transliterated as ko-n-ni-chi-ha (and not ko-ni-chi-wa) so that it will correctly transliterate intoこんにちは.

It is preferred that the transliteration process is lossless, so that one could re-construct the original phrase from the transliterated one. However, this may not be practical to achieve. This is because not all sounds are available in all languages. For example, in Japanese there is no distinction between the sounds R and L. Both "road" and "load" will transliterate into ロード. Japanese also has no distinction between V and B. Similarly, in Tamil, one of the South Indian languages, there is no distinction between the sounds P and B. Both "pat" and "bat" will transliterate into பற். This results in information loss and will lead to ambiguity if back-transliteration is required.

There are also a number of syllables in Asian languages that are alien to English. This too makes back-transliteration difficult. For instance, Tamil and Malayalam have a ழ syllable which is hard to pronounce for non-native speakers. This syllable is commonly transliterated in English as "zh".

Given a transliterated phrase such as "konnichiwa", the transliteration engine produces either the text こんにちは which allows pasting into documents, or the Unicode HTML string "こんにちは " for direct use in HTML documents, or the Unicode C string "\u3053\u3093\u306B\u3061\u306F" which can be used in C-like languages for outputting Unicode text.

The transliteration engine currently has script mappings for Japanese Hiragana and Katakana, Devanagari (used to write Hindi, Sanskrit, Marathi, etc.), Tamil, and Telugu. These mappings are based on the modified Hepburn system. Table 2 illustrates a partial mapping for Tamil. Note that there is no distinction in Tamil between B and P and thus both map onto the same Unicode symbols. Table 2 shows the B sounds.

The script mappings in the engine can be extended to include other scripts (e.g. Thai, Burmese, Sinhala). Providing mappings for syllabary and alphasyllabary scripts is quite simple. Using the modified Hepburn system for script mapping makes it easy to understand and extend.

Table 2: A partial script mapping for Tamil.

Source string	Target string	Unicode symbol
b	\u0BAA\u0BCD	ப்
ba	\u0BAA	ப
baa	\u0BAA\u0BBE	பா
bi	\u0BAA\u0BBF	பி
bii	\u0BAA\u0BC0	பீ
bu	\u0BAA\u0BC1	பு
buu	\u0BAA\u0BC2	பூ
be	\u0BAA\u0BC6	பெ
bee	\u0BAA\u0BC7	பே
bai	\u0BAA\u0BC8	பை
bo	\u0BAA\u0BCA	பொ
boo	\u0BAA\u0BCB	போ
bau	\u0BAA\u0BCC	பௌ

4 EVALUATION

The transliteration engine is a useful tool for electronic composition and publishing of foreign language phrases. This helps language teachers and learners to compose ethnic scripts using a Standard English keyboard.

The engine can also produce portable HTML Unicode strings to enable direct embedding in HTML documents.

Besides, the engine can produce C-style Unicode strings that can be used in C-like programming languages. This is useful when an application needs to be localized for various cultures.

One of the drawbacks of the engine is that it does not do back-transliteration. The reason for this is the ambiguities involved in back-transliteration. When more than one sound symbol of the source maps onto a single sound symbol of the target, information is lost, and thus back-transliteration becomes ambiguous. One of the implications of not providing back-transliteration is that a transliterated phrase cannot be edited for corrections – it should rather be replaced with a new (possibly correct) phrase.

A comparable transliteration engine is provided with Microsoft Office®. This is part of the input method editor (IME), and allows direct input for some languages such as Japanese and Korean. Office allows saving to HTML documents, but does not have a mechanism to generate C-style Unicode strings. Besides, support for Indian languages such as Tamil is limited and is soft-keyboard-driven.

5 CONCLUSIONS

This paper discussed some aspects of transliteration and presented the development of a transliteration engine which suits syllabary and alphasyllabary scripts. The engine is useful in teaching and learning ethnic scripts. It is also a useful tool for the internationalization and localization of computer programs, publishing ethnic scripts over the Internet, and to compose electronic documents in ethnic scripts.

REFERENCES

Al-Onaizan, Y. and Knight, K., Machine transliteration of names in Arabic text. In *Proceedings of the ACL-02 workshop on Computational approaches to Semitic languages*. Philadelphia, Pennsylvania, (2002) 1-13.

Grefenstette, G., Yan, Q., and Evans, D. A., ,Mining the Web to Create a Language Model for Mapping between English Names and Phrases and Japanese, In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*. (2004) 110-116

Kang, I-H. and Kim, G., English-to-Korean transliteration using multiple unbounded over-lapping phoneme chunks, In *Proceedings of the 18th International conference on Computational linguistics*. Saarbrücken, Germany, (2000) 418-424

Kawtrakul, A., et al. Backward transliteration for Thai document retrieval. In *Proceedings of the Asia-Pacific conference on Circuits and Systems*. Chiangmai, Thailand, (1998) 563-566

Knight, K. and Grehl, J., Machine Transliteration. In *Computational Linguistics*, Vol.24, No.4, (1998) 599-612

Sakai, T., Kumano, A., and Manabe, T., Generating transliteration rules for cross-language information retrieval from machine translation dictionaries, In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, (2002) 6-12

The Unicode Consortium.The Unicode Standard, Version 4.0. Addison-Wesley, Boston, USA (2003)

APPENDIX

Figure 1 in this appendix shows a screenshot of the transliteration engine's desktop user-interface. The engine also has a web-based interface (not shown here).

Figure 2 illustrates the possible syllables in Japanese and the corresponding Katakana symbols.

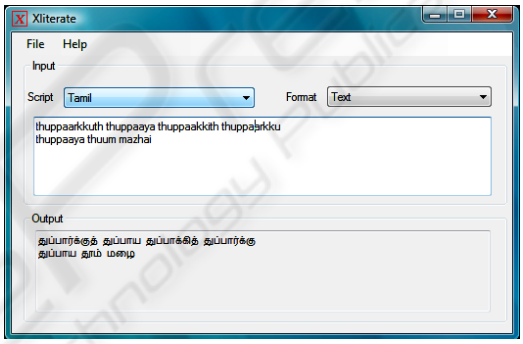

Figure 1: A screenshot of the transliteration engine's user-interface. The interface allows a choice of scripts (such as Katakana and Tamil) and output formats (text, HTML, or C string). The output can be pasted to a variety of applications.

a	ア	i	イ	u	ウ	e	エ	o	オ
ka	カ	ki	キ	ku	ク	ke	ケ	ko	コ
ga	ガ	gi	ギ	gu	グ	ge	ゲ	go	ゴ
sa	サ	shi	シ	su	ス	se	セ	so	ソ
za	ザ	ji	ジ	zu	ズ	ze	ゼ	zo	ゾ
ta	タ	chi	チ	tsu	ツ	te	テ	to	ト
da	ダ	ji	ヂ	zu	ヅ	de	デ	do	ド
na	ナ	ni	ニ	nu	ヌ	ne	ネ	no	ノ
ha	ハ	hi	ヒ	fu	フ	he	ヘ	ho	ホ
ba	バ	bi	ビ	bu	ブ	be	ベ	bo	ボ
pa	パ	pi	ピ	pu	プ	pe	ペ	po	ポ
ma	マ	mi	ミ	mu	ム	me	メ	mo	モ
ra	ラ	ri	リ	ru	ル	re	レ	ro	ロ
wa	ワ							wo	ヲ
ya	ヤ			yu	ユ			yo	ヨ
				n	ン				

Figure 2: Japanese Katakana alphabet with their pronunciation.