

# Challenges of enabling IT in the Sinhala Language

Gihan V. Dias  
ICT Agency of Sri Lanka  
gihan@cse.mrt.ac.lk

## Abstract

Although Sinhala is the national language of Sri Lanka, even in 2004 most computer operating systems, databases and applications were in English and only a handful of Sinhala websites existed. Many people thought that “computers don’t work in Sinhala”.

Sinhala was included in Unicode in 1998, but there were no implementations even by 2002.

This paper first examines why IT in Sinhala was slow to develop. It then describes the development of Sinhala computing over the last two years, and the challenges faced.

One reason for the non-adoption of Unicode was that the Unicode standard, as published, did not specify some common symbols and ligatures, which led to a perception that Unicode did not properly support Sinhala. Unicode was also complex, and difficult to understand. Another issue was the lack of operating system support, especially in Microsoft Windows.

We were successful not only in defining the full representation of Sinhala script in Unicode, but also the specification and deployment of Sinhala computer keyboards.

We learned that perception and prioritisation are as important as technical issues in enabling IT in a language.

## 1 Introduction

The Sinhala language is spoken by about 15 million people and is the national language of Sri Lanka which has a high literacy rate of over 90%. Computers are in reasonably wide use in the country, especially in offices.

However, even today, there are no significant databases, very few websites or other on-line documents, and almost no use of e-mail and messaging in Sinhala.

This is in stark contrast to European languages, such as Swedish, with far fewer speakers. However, other Indic languages such as Tamil and Bangla are in a similar situation.

### 1.1 History

The first efforts to introduce Sinhala language computing in Sri Lanka occurred in the 1980’s. Due to the lack of bitmapped devices, they required that the fonts be embedded in the displays and printers, and were thus limited to specific hardware. This limited their adoption, and they were eventually discontinued.

Thereafter, a Sinhala font was developed for the Macintosh, which was widely used in publishing. With the advent of MS-Windows, several organisations produced Sinhala fonts and word processing packages, which are currently in use.

## 1.2 Reasons for the Limited use of IT Sinhala

The reason for the low use of IT in Sinhala is not the lack of fonts, etc., which are widely available.

In the author's opinion, one reason for the low usage of Sinhala is that an appreciable number of Sinhala speakers also know some English. Many people consider that using English is "better" than using Sinhala, and prefer to do so, even if they are not completely fluent in English. There is also a perception that "computers work in English" and many people have not made an effort to use Sinhala on computers.

Another reason is the use of a non-Roman script. Until the wide availability of bitmapped displays and printers, use of non-Roman scripts required special hardware.

However, the main reason for the low use of Sinhala was the lack of a single standard. Hardware vendors were hesitant to provide Sinhala hardware (e.g. keyboards) without a standard. Content providers and database designers too were hesitant to use Sinhala in the absence of a standard.

## 1.3 Our Objectives

The initiative described in this paper was started in early 2003 by the Council for Information Technology (and continued by the ICT Agency). Our objective was to ascertain why the usage of Sinhala (and Tamil, the other official language) was so low, and to take steps to increase its use. Our target was for computers sold in the country to support Sinhala and/or Tamil by default, and their use to be as easy as using English.

Our work consisted of standard setting [11], developing hardware and software, advocacy and training, in the following areas:

- encoding of Sinhala characters
- development of Sinhala fonts
- standardisation of a Sinhala keyboard and
- standards-based applications and utilities (such as spelling checkers).

# 2 Sinhala Encoding

## 2.1 Existing Encodings

Most non-Unicode Sinhala fonts use a keyboard-based encoding. i.e., the characters are encoded at the same positions as the ASCII character at that keyboard position. Based on this principle, the two major classes of Sinhala fonts are based on the Wijesekera and "phonetic" keyboards. However each font vendor uses a slightly different encoding, making document transfer difficult.

A proposed encoding of Sinhala by researchers based in Europe was first brought to our notice in the late eighties. This encoding had several glaring errors and omissions. For example, the letters  $\text{අූ}$  and  $\text{ආූ}$  (which do not appear in other Indic languages), had been moved to the end of the character set. This showed us the dangers of national bodies not being proactive, and allowing non-native speakers to define our language. A Sinhala character set and encoding were developed by the Committee on Adaptation of National Languages in IT (CANLIT) in 1990 [5].

A standard Sinhala encoding, known as SLASCII, was approved by the Sri Lanka Standards Institute as SLS 1134 in 1996 [3]. SLASCII has a structure similar, but not identical, to the ISCII standards for Indian languages. SLASCII was the basis for, but differs in many aspects

from, the Unicode encoding for Sinhala.

In 1997, Sri Lanka submitted a proposal for the Sinhala character code at the Unicode working group meeting in Crete, Greece, which competed with proposals from UK, Ireland and the USA. The Sri Lankan draft was finally accepted with slight modifications and was included in Unicode [4] Version 3.0. SLS 1134 was also accordingly revised in 2001. More details are in [1].

## 2.2 Features of Sinhala

As in other Indic languages, the Sinhala script comprises consonants and vowels. A vowel following a consonant is represented by one or more strokes arranged around the consonant. The shape and position of a stroke may vary, depending on the base consonant. The default vowel is indicated by the absence of a stroke. A pure vowel is generally used only at the beginning of a word, and has a distinct symbol [6].

Strokes may not only be placed on any side, i.e., above, below, to the left, or right of a consonant, but a vowel may be represented by two or more strokes. For example the vowel modifier for long o (ඹ) consists of the three strokes *kombuva* (before the consonant), *aela-pilla* and *al-lakuna* (after the consonant), e.g. ක+ඹ = කඹ.

Unlike North Indian scripts such as Devanagari, and like South Indian scripts such as Tamil, each Sinhala letter is generally written discretely, without touching adjacent characters. A *pure* consonant (i.e. without an associated vowel) is normally represented by adding an *al-lakuna* to the consonant symbol, and not by combining it with the subsequent letter.

However, there are some exceptions, for example:

The letters *ya* (ය) and *ra* (ර) following a pure consonant are normally represented by the strokes *yansaya* (෪) and *rakaransaya* (෫) respectively, although they may infrequently be written with an explicit *al-lakuna*.

Some pairs of consonants may be joined to form a conjunct letter, e.g., *ksha* (කෂ). Also the letter *r* (ර) preceding a consonant may be shown by the stroke *rephaya* (ආ). These constructs are optional, and have become less common in contemporary use.

An encoding of Sinhala should be able to correctly and efficiently represent all of the above features, as well as others not discussed here. It is possible to correctly represent Sinhala text by linear sequence of symbols, some of which appear above or below another symbol. In this sense, Sinhala is *not* a complex or bidirectional script.

## 2.3 The Unicode Encoding of Sinhala

The Unicode encoding of Sinhala, as defined in [4], follows the model of other Indic languages. It contains codes for vowels, consonants and vowel modifiers. Encoding is based not on symbols or glyphs, but on *linguistic* units. For example, although the *kombuva* is written before a consonant, it is encoded following the consonant.

However, this results in a complex mapping from the keyboard input to the Unicode representation, and from Unicode to the screen display.

## 2.4 Deficiencies in the Unicode encoding

It was observed that the Unicode encoding suffered from a number of shortcomings. These are:

- lack of encodings for conjunct letters such as කෂ,

- lack of encodings for the *yansaya*, *rakaransaya* and *rephaya*,
- lack of guidance on the use of multiple vowel modifiers and
- lack of guidance on the encoding of non-standard letters, such as ක, ජ and ඵ.

The Unicode block for Sinhala simply lists the encoded symbols, and gives no guidance on implementation. While listing the symbols may be sufficient for encodings in which each symbol is assigned a code, it is not possible to implement Sinhala in Unicode without additional information. Although the Unicode website clearly states “Unicode support requires considerably more than providing glyphs for characters”, a casual visitor gets the impression that the encoding of Sinhala in Unicode is incomplete, and is not guided to sources of further information.

Although chapter 9 of the Unicode standard [4], purports to provide such guidance, it does not describe Sinhala fully. Instead, it considers Devanagari as the canonical form, and covers Sinhala in just three paragraphs. As explained by Constable [9], it is not correct to assume that Devanagari is representative of other Indic languages. We should develop appropriate encodings for each language.

The Working Group developed specifications for encoding all valid constructs, and documented them as a revision to the SLS 1134 standard [2]. A summary of the specifications is in [12]. By using these specifications, which are fully conformant to Unicode version 3.0 (and later), it is possible to represent not only all contemporary Sinhala writing, but also classical writing including Pali and Sanskrit texts written in Sinhala script.

In the following sections, we describe some of the issues which arose during the above process.

## 2.5 Code Block Issues

### Symbols omitted and included in the Sinhala code block

The code block for Sinhala does not include symbols such as *yansaya* (ය්) and *rakaransaya* (ර්), nor conjunct letters such as ඌ. The representations of these symbols are *not* specified anywhere in the Unicode Book (version 4.0).

Although the use of such constructs in other Indic languages is documented and Sinhala constructs may, by inference, be represented in a similar manner, most Sinhala speakers have no inclination to study the standard and make such inferences, and come to the conclusion that these symbols are not included in Unicode. This issue is not limited to Sinhala, but for all scripts where symbols are represented by code sequences.

Another issue is the inclusion of archaic and unused letters in the code table. Although such letters are not used in Sinhala, their inclusion in the table implies that they should be implemented. This complicates an implementer’s task.

### Collation Order

The Unicode documentation clearly states that the placement of symbols in the encoding is *not* indicative of the collation order. However, many people, not having read the documentation, conclude that Unicode implements an incorrect collation order.

## 2.6 Implementation Issues

### Kombu Re-Location

Unicode encodes all vowel modifiers following the consonant. However the *kombuva* symbol, which may occur as a single vowel modifier, or as part of a two- or three-part modifier, appears *to the left of* a consonant. Unicode therefore classifies Sinhala as a complex script. However, this usage is similar to the vowel sound in the English word “came”, which is split into two symbols on the sides of the consonant “m”, although English is not a complex script.

Although the encoding scheme used for Sinhala (and other Indic scripts) is linguistically superior, it raises implementation issues. For example, although MS-Windows 2000 supports Unicode, it does not support Sinhala as it does not support the re-location of the kombuva. An unintended consequence of this decision taken in the 1990's is the current (Dec. 2004) impossibility of representing Sinhala on stock MS-Windows, or Linux.

### The Zero-Width Joiner

The zero-width joiner is used extensively in Sinhala, as in other Indic scripts. However, it is not in the same code block as the other Sinhala symbols. Therefore, some implementations (incorrectly) do not recognise it to be a Sinhala code, and do not process it properly. One reason is that Unicode does not use these joiners consistently. For example, they are never used in European scripts. As many implementers do not understand the usage of these characters, they may make errors in their implementation.

## 3 Keyboard

Three types of computer keyboard input methods are used for Sinhala.

- The Wijesekera keyboard, which was originally developed for typewriters,
- “phonetic” keyboards, in which key assignment is based on the English key layout, and
- transliteration schemes, in which text is typed as a sequence of English letters.

None of these keyboard types directly map to the Unicode character encoding model. One option was to design a *consonant-vowel sequence* keyboard, in which a consonant is typed first, followed by a vowel modifier, which would map cleanly to the Unicode encoding. Although this method may be more efficient, users were not convinced that it was an improvement. The feedback we received was “we don't want a change”. Therefore, we decided to standardise a Wijesekera- (or typewriter-) based layout. However we made some modifications, especially by assigning each vowel stroke to a single key, irrespective of the shape of the stroke. This layout is specified in SLS 1134 [10].

Interestingly, we found that although no Sinhala computer keyboards were produced before our initiative, manufacturers were interested in producing them as soon as the standard was announced. Currently at least three manufacturers produce Sinhala keyboards.

The standard Sinhala keyboard can produce Unicode only by using a state table-based driver. Although the software itself is straightforward, we found that the system keyboard drivers in both MS-Windows [8] and Linux did not support such drivers, which must display intermediate symbols as text is generated and then *substitute* such intermediate symbols by other symbols as further keys are pressed. In this instance too, we felt that operating system designers have overlooked the need for such drivers.

## 4 Deployment Issues

The two major operating systems used in Sri Lanka are MS-Windows and Linux. We wanted to ensure that both these systems fully support Sinhala, ideally “out of the box”. Although we also want support for other systems such as Macintosh, PDAs and mobile phones, we decided to initially concentrate on the first two.

The critical component of MS-Windows needed for Sinhala Unicode support is a driver named USP10.DLL. Although Microsoft has a version of this driver which supports Sinhala, it is not included with the current version of Windows. As Microsoft does not allow users to install this

driver in the system folder, and a service pack enabling this driver was released only in January 2005, we had to develop work-arounds to allow Windows to be used in Sinhala.

We found that a sense of urgency to enable Sinhala in Windows was lacking, as Sri Lanka is a small market. However, for a Sinhala user, this is a critical update, as he otherwise cannot use Windows in his language.

In Linux, on the other hand, the Lanka Linux Use Group (LKLUG) were able to enable Sinhala support without undue difficulty. The challenges faced were the multiplicity of display systems, and the difficulty of differentiating the Sinhala encoding model from the Devanagari one.

We initially envisaged that developing fonts for Sinhala would be difficult. However, with help from a font design expert, we were able to develop a number of Unicode compatible fonts. Some problems we faced were the lack of documentation of `usp10.dll`, and the complexity of the OpenType format. We also found that OpenType [7] did not have built-in support for features such as split vowel modifiers.

One unanticipated problem we encountered was the name of the language in English. The commonly used name for the language is “Sinhala”. However, a number of implementations and standards used the name “Sinhalese”, which is somewhat archaic. It took us some time and effort to reach consensus on the name.

## 5 Conclusion and Recommendations

The task of standardising and implementation of Sinhala in IT was very much a collaborative effort. I wish to thank the members of the working groups, driver and font developers, and all others who made it possible.

On the way, we encountered a number of impediments, mostly unintentional. I note some points which may be relevant to other languages as well. These observations and recommendations are:

### **More local involvement**

When encoding contemporary (as opposed to historic) languages, ensure that native language speakers, scholars and engineers give their informed input to the process. We are grateful for the support and encouragement given by the Unicode community in our standardisation process, but note that many people, both within and outside Sri Lanka, still view Unicode as a group of uninformed foreigners. It would be helpful if the local community were engaged early in the standardisation process.

### **List all common symbols in the code table itself**

The Unicode code table is a mapping of codes to symbols. Some language constructs may be formed by sequences of codes. The symbols, codes and sequences needed to encode a language may be found in many parts of the table. I recommend that in addition to the code-wise listing, Unicode produce listings of all the symbols and constructs in each language, which specify their encoding. This listing should be in the correct collation order, and accompanied by the notes and examples needed to understand the encoding.

### **Separate historical and depreciated symbols from contemporary ones**

The above listing should separate historical and depreciated symbols and codes from the ones which need general implementation.

### **Do not attempt to generalise Devanagari**

Unicode appears to consider Devanagari to be the canonical Indic script. However, it is quite different from Sinhala and South Indian scripts. We should not attempt to force-fit a language into the model of another language.

### **Consider Implementation issues**

When producing encodings, consider ease of implementation, even by unsophisticated and ill-informed persons. Keyboard input methods should also be addressed when deciding on encoding.

### **Seeing is Believing**

The availability of Sinhala keyboards changed the perception of the feasibility of Sinhala computing. Hardware should be used to drive adoption.

On the whole, we find that the Unicode provides a good encoding of Sinhala, and would only recommend a few changes to the current encoding. We do consider, however, that the same concepts should be used for encoding all scripts, rather than using different sets of features for different scripts.

## **References**

- [1] V.K. Samaranayake, S.T. Nandasara, J.B. Disanayaka, A.R. Weerasinghe, H. Wijayawardhana, *An Introduction to UNICODE for Sinhala Characters*, UCSC Technical Report 03/01, University of Colombo School of Computing, 2003.
- [2] Sri Lanka Standards Institute, *Sri Lanka Standard SLS 1134:2004 - Sinhala Character Code for Information Interchange*, Revision 2, SLSI, 2004.
- [3] Sri Lanka Standards Institute, *Sri Lanka Standard SLS 1134:1996 – Sinhala Character Code for Information Interchange*, SLSI, 1996.
- [4] The Unicode Consortium, *The Unicode Standard 4.0*, Addison-Wesley, 2003. Available at <http://www.unicode.org/standard/standard.html> .
- [5] S.T. Nandasara, J.B. Dissanayake, V.K. Samaranayake, E.K. Seneviratne and T. Koannantakool, *Draft Standard for the Use of Sinhala in Computer Technology*, CINTEC, March 1990.
- [6] J. B. Disanayaka, අක්ෂර හා පිළි (*Letters and Strokes*), Godage, 2000.
- [7] Adobe Corp., *An Introduction to OpenType*, [online]. Available: <http://www.adobe.com/type/opentype/main.html> .
- [8] Michael S. Kaplan and Cathy Wissink, “Unicode and Keyboards on Windows”, in *Proc. 23rd Internationalization and Unicode Conference*, March 2003. Available: <http://www.microsoft.com/globaldev/handson/dev/Unicode-KbdsonWindows.pdf>
- [9] Peter Constable, *Proposal on Clarification and Consolidation of the Function of ZERO WIDTH JOINER in Indic Scripts*, Review document, Unicode Consortium. Available: <http://www.unicode.org/review/pr-37.pdf>
- [10] Sri Lanka Standards Institute, *The standard Sinhala keyboard layout* (incorporated in SLS 1134), 2004. Available: <http://www.fonts.lk/doc/sin-kbd-layout5.pdf>
- [11] Gihan Dias and Aruni Goonetilleke, “Development of Standards for Sinhala Computing”, in *Proc. 1<sup>st</sup> Regional Conference on ICT and E-Paradigms*, Colombo, Sri Lanka, 2004. Available: [http://www.fonts.lk/doc/sinhala\\_standards.pdf](http://www.fonts.lk/doc/sinhala_standards.pdf)
- [12] Gihan Dias, *Representation of Sinhala in Unicode*. Available: <http://www.fonts.lk/doc/Representation of Sinhala in Unicode.pdf>