1st Regional Conference on ICT and E-Paradigms
24th – 26th June 2004, Colombo, Sri Lanka

Development of Standards for Sinhala Computing

Gihan Dias and *Aruni Goonetilleke*
ICT Agency

Abstract

Information technology has been used in Sri Lanka for about 20 years, but to a great extent, its use has been limited to those with a knowledge of English. And as these are a minority of the population, the larger proportion of Sri Lanka's citizens has not been able to make use of the information revolution.

A number of initiatives to introduce Sinhala computing have been made from the 1980s. We outline the progress of this work, and describe the work done by CINTEC and ICTA to support Sinhala computing.

The encoding of Sinhala in a standard manner to facilitate information interchange, the development of Sinhala fonts, and Sinhala keyboards are described. We present the rationale of the design, and not simply the final standard.

1. Introduction

Currently most computer operating systems, databases and applications in Sri Lanka work only in English. However, the majority of Lankans are more familiar with Sinhala or Tamil, and prefer to use IT in their own language. This has resulted in a gap between how people want to use computers, and what today's computers can do.

Although the corporate sector in this country operates mainly in English, small businesses and the government work mostly in Sinhala or Tamil. Individuals use computers for personal work, such as e-mail, at home or in telecentres. Lankans working abroad need to communicate with their relatives in Sri Lanka. The dispersion and effective use of IT in these sectors require that they support our languages.

Countries such as Japan, Korea and Thailand, among others, had similar problems when they started using computers. However, Japanese, Korean or Thai language is now standard on computers used in those countries, and it is common for people in those countries to use computers and applications in their own language.

Why then didn't local-language computing become common in Sri Lanka? One reason is that unlike in the above countries, a number of Lankans, including many decision and opinion makers, *are* proficient in English. Also, people assume that using IT requires a knowledge of English, and have not demanded a change. Another reason is the small size of the Lankan market.

This situation is now changing. A significant number of non-English-speaking persons want to use Information Technology (IT). This is not limited to the use of computers, but devices such as phones, game consoles etc.

This paper documents the initiatives, by a number of persons and organisations, to make IT readily available to Sinhala speakers. A similar initiative, not covered in this paper, is being carried out both in Sri Lanka and abroad, on IT in Tamil.

2. Local Language Support Requirements

A number of inter-related elements are required to support a given language in a computer system. These are:

Character Encoding: how letters and words are represented in a system. Each letter or other symbol is represented by a code. For documents to be portable across systems, they must be encoded in a standard format. For example, the ASCII code is one (but not the only) method of encoding English text.

Fonts: how text is represented on a screen or printer. The same character may be written in several ways, with scope for artistic expression.

Text input: from a keyboard, pen, voice recognition system, etc. The most common text input method is the keyboard. Both the keyboard layout, i.e., the assignment of keys to letters, and key sequences, i.e., what sequence of keys yields a given character, should be defined.

Application support for the language. In addition to text handling, each application may have local language menus, error messages, help screens, etc.

Utilities: such as spelling checkers.

3. Review of Current Sinhala Technology

A number of Sinhala fonts and applications are currently available. These may be categorised as::

fonts – which may be used with any application and

packages which bundle an application (e.g. a word processor) with a set of fonts.

Although these systems are disparate, many of them share a number of features. These are:

8-bit character set: Almost all current fonts are based on an 8-bit character encoding, which limits them to less than 256 symbols.

Character codes based on keyboard layout: Many current fonts map Sinhala symbols to the codes used by Roman letters in ASCII. The two most common mapping schemes are based on the Wijesekera keyboard layout and the "phonetic" layout (i.e., where Sinhala letters are placed on the same keys as their English sound-alikes, e.g. ක is on the 'k' key and ග is on the 'g' key).

Consequently, the codes allocated to Sinhala letters are based on the codes of the corresponding English letters. In a “phonetic” mapping the letter ක has the code Hex 6B, which is the ASCII code for the letter k. In a Wijesekera mapping, on the other hand, ක is on the key for 'l' (el), and is therefore assigned the code Hex 6C (ASCII for l).

In addition to the vowels (අ, ඉ etc.) and consonants (ස, ඡ etc.), the *kombuva* (ෙ) and other symbols (ා ෑ ු etc.) are each allocated a key and a code. Symbols such as the *al-lakuna*, *ispili* and *paapili* are displayed above or below the previous character. A Sinhala letter is represented by one or more codes; e.g., (කෙ = ෙ+ ක, කො = ෙ+ ක + ා, නු = න + ු).

As Sinhala letters are of differing widths (and heights), some systems provide two or more *pili* of differing widths, to be used with various letters, e.g., පු and නු. Also, some letters, e.g., කු, රු and ළු, take non-standard forms, and are thus represented by individual codes.

Use of an 8-bit character code, especially one using the same codes as English letters for Sinhala symbols, simplifies usage as applications need no modification to use Sinhala. As far as an application is concerned, it deals with the same codes in Sinhala as in English.

3.1 Issues with existing Sinhala packages

Although existing Sinhala packages serve their intended purpose, i.e., allowing Sinhala documents to be produced, and to use applications in Sinhala, they face a number of problems, namely:

Lack of needed letters: Due to the constraints of the 8-bit encoding, some existing systems do not support all Sinhala letters, e.g. ඎ (iru-yanna). In addition, due to keyboard limitations, some letters need to be entered using function keys, keycodes, or the “insert symbol” feature.

Problems with Pili: some systems provide only one *keti-ispilla*, for example, which is used for both tall letters (e.g., ව) and short ones (e.g., ග). Such fonts are not visually attractive. A similar problem arises with the length of the *pili*. Another problem is the separation of letters and associated *pili* when intra-word spacing is increased for justification, etc.

Collation and Searching do not work properly, as the encoding of letters is not based on their collation sequence. Also, the use of separate codes for the *kombuva* and *pili* complicates the collation algorithm.

3.2 Non-Standardisation

The most significant problem with existing systems, however, is not any of the above, but the lack of a standard. Text entered using one system must be read using the same system. Even though most Wijesekera-based systems use similar encodings, they differ in certain characters. It has therefore become mandatory to send a copy of the font together with a Sinhala document, unless one knows that the recipient already has the font. This has made Sinhala e-mail impracticable, and slowed the use of Sinhala on the Web.

The lack of a standard Sinhala keyboard not only causes difficulties for typists, but has made it impractical for manufacturers to provide Sinhala keyboards. This in turn has created a further roadblock for Sinhala computing.

4. Review of the Development of Computing in Sinhala

A number of efforts were made to introduce Sinhala language computing in Sri Lanka. Two pioneering efforts by DMS and Metropolitan in the 1980s did not gain widespread use. A patent dispute between these two companies threatened the development of Sinhala language computing, but was eventually settled out of court. It should be noted that bit-mapped displays and printers were not in widespread use at that time, and both these systems used downloaded fonts.

Thereafter, Wijeya Graphics produced a Sinhala font for Macintosh, which was widely used in publishing. The University of Colombo developed a Sinhala screen output for television displays that was used to provide election result displays in the three languages Sinhala, Tamil and English.

4.1 The Sinhala Alphabet and Alphabetical Order

The requirement for a standard code was identified in the mid-eighties and the Computer and Information Technology Council of Sri Lanka (CINTEC) together with the Natural Resources, Energy and Science Authority of Sri Lanka (NARESA) formed the Committee on Adaptation of National Languages in IT (CANLIT), which agreed on a unique Sinhala alphabet and alphabetical order. No immediate action was taken on Tamil, due to the work being undertaken in India.

CANLIT defined the Sinhala alphabet as having 16 vowels, 2 semi consonants and 41 consonants as shown in the CINTEC publication of 1990 [5]. 13 consonant modifiers were also identified. A new character to denote “fa” (ෆ) was introduced. CANLIT also agreed on the alphabetical order with a slight modification. This exercise took a representative group of language and technology experts several months to arrive at a consensus solution

A standard Sinhala encoding, known as SLASCII, was approved by the Sri Lanka Standards Institute as SLS 1134 in1996 [3]. SLASCII was the basis for, but differs in many aspects with, the Unicode for Sinhala approved later in 1998.

4.2 Unicode Compatible Sinhala Code

A standard encoding known as Unicode has been developed to handle all the world's scripts. Unicode includes provision for many languages, including Sinhala and Tamil, among other Indic languages. Most modern computer systems, including Windows XP and Unixes, support Unicode.

The existence of a draft code for Sinhala proposed to the ISO 10646 / Unicode Working Group, by researchers based in Europe was first brought to our notice in the late eighties. This distorted the Sinhala character set with several glaring errors and omissions. For example, the letters ඇ and ඈ (which do not appear in other Indic languages), had been moved to the end of the character set.

In 1997, Sri Lanka submitted a proposal for the Sinhala character code at the Unicode working group meeting in Crete, Greece This proposal competed with proposals from UK, Ireland and the USA. The Sri Lankan draft was finally accepted with slight modifications. This was ratified at the 1998 meeting of the working group held at Seattle, USA and the Sinhala Code Chart was included in Unicode [4] Version 3.0. SLS 1134 was also accordingly revised in 2001 [1].

4.3 Sinhala Computer Keyboard

As a large number of Sinhala typists were using the government approved Wijesekera Keyboard, CINTEC first developed and obtained government approval for the "*Extended Wijesekera Keyboard for Electronic Typewriters*", the intention being the introduction of Daisywheel and Golf-ball electronic typewriters then used as an interface for microcomputer output. This included the new character fa (ෆ) and 3 other additional key positions. This layout was again modified for use with a standard 101-Key computer keyboard [5].

4.4 Recent initiatives

In December 2002, CINTEC formed a committee on Sinhala Fonts.

The committee decided that work should be carried out in four major areas in order to enable computing in Sinhala. These were:

- A standard encoding of Sinhala characters
- Development of Sinhala fonts
- A standard Sinhala keyboard
- Availability of standards-based applications and utilities (such as spelling checkers)

The functions of this committee were incorporated into the ICTA Language Requirements working Group in January 2004. The standardisation work has been carried out by the SLSI Sinhala Working Group since August 2003.

The first three are described in the following sections. Work on the fourth is still on-going.

5. Encoding

An encoding is a standard for representing letters and words as binary data. The committee agreed on the following requirements of the character encoding.

- It must be able to represent all contemporary and classical Sinhala text.
- It should facilitate collation and searching.
- It should be efficient

A language such as Sinhala may be encoded in several ways. For example, each symbol may be assigned a code; e.g., කො may be represented as ෙ ක + ා. Alternatively, each letter, such as කො, may be assigned a single code.

Unicode represents each vowel or consonant by a code, and uses another code for each vowel sign, which modifies a consonant; e.g., කො is represented by two codes, ක + ො . This approach follows the linguistic structure of Indic languages such as Sinhala, whose basic unit is a syllable formed by following a consonant with a vowel.

The committee considered the encodings used in existing Sinhala implementations, which are all symbol-based, and decided that none of them met the above requirements. It was therefore decided to follow the Unicode standard, which is identical to SLS 1134 and ISO 10646.

However, it was observed that the Unicode encoding suffered from a number of shortcomings, which nevertheless can be rectified without modifying the encoding. These are:

- lack of encodings for *bandi akuru* such as ක්‍ෂ,
- 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 කෘ, ඍ and ළු.

It was decided to document these as a revision of the SLS 1134 standard [2]. In addition, the sections of the standard on definitions and the description of the Sinhala language were clarified.

When doing so, the committee decided to avoid any changes to the encodings specified in Unicode, and also to follow Unicode conventions whenever feasible.

5.1 Conjunct letters (බැඳි අකුරු)

Unicode does not encode any conjunct letters, such as න්‍ද. Such letters are a shorthand for writing a pure consonant (i.e., a consonant with the intrinsic vowel removed) followed by another consonant. E.g. න්ද = න්‍ද. Both representations may be used interchangably. It was therefore decided to use the Unicode *zero width joiner* (ZWJ) code to indicate when a conjunct letter is to be used. E.g., න + ්+ ද = න්ද, න + ්+ zwj + ද = න්‍ද. The zero-width joiner is a special Unicode code which is used to indicate that the two adjacent letters are joined.

The committee compiled a list of conjunct Sinhala letters as a guide to font developers.

5.2 Yansaya and Rakaransaya

A frequent critisism of Unicode is the lack of codes for the *yansaya* and *rakaransaya*. The Unicode documents fail to mention these two symbols. However, we realised that this omission was deliberate, as neither of these symbols are Sinhala letters. Rather, they are abreviations for the letters ය and ර respectively, when they follow a pure consonant. E.g., සත්‍ය is the conventional way of writing සත්ය and මිත්‍ර represents the sequence මිත්ර.

As such words are generally spelled using the *yansaya* or *rakaransaya*, it was initially proposed to represent a ය or a ර following a pure consonant using the relevant symbol. However, some words, such as මල්රාජ් and පස්යාල, do not use the constructs. We were thus faced with two alternatives:

- encode the common case without any special codes; e.g., ත+ ්+ ර = ත්‍ර and use the code *zero-width non-joiner* (zwnj) to indicate when the construct should *not* be formed; e.g., ත+ ් + zwnj + ර = ත්ර
- use the code *zero-width joiner* (zwj) to indicate when the *yansaya* or *rakaransaya* should be formed; e.g., ත + ්+ zwj + ර = ත්‍ර, ත + ්+ ර = ත්ර.

The first alternative yields a shorter code sequence for the more common case, and also follows the Unicode convention that the common case is encoded without special codes. However, the committee selected the second alternative for two reasons.

Keying the sequence ක + ් + ර would otherwise have automatically produced a ක්‍ර, even if not desired by the user. As the recommended keyboard has specific keys for the ්‍ය and ්‍ර users would use these keys to generate the *yansaya* and *rakaransaya*, and the above key sequence for producing ක්ර.

Using the zwj to produce the *yansaya* and *rakaransaya*, which are forms of conjunct letters, allows us to use the same representation for all conjunct letters.

5.3 Rephaya

The *rephaya*, though less common today, is used to represent the letter ර් preceding another letter. For example the word කර්ම may also be written as කර්‍ම. This construct represents a pure consonant (ර්) followed by a letter, and may be encoded similarly to a conjunct letter using a zero-width joiner; e.g., ක + ර + ් + zwj + ම = කර්‍ම.

The sequence ර්ය presents an interesting case. It may be written as ර්ය, ර්‍ය or ර්‍ය්‍ය [6]; e.g., ආර්ය, ආර්‍ය or ආර්‍ය්‍ය. The second form is represented by ර + ් + zwj + ය. The sequence for the third form was not obvious, but was finally defined as ර + ් + zwj + ය + ් + zwj + ය.

5.4 A Model of Sinhala text

When considering the above cases, it was seen that conjunct letters, *yansaya*, *rakaransaya* and *rephaya* all represent combinations of two or three letters, where the initial letter or letters are pure consonants. Such combinations may also be written as separate letters. We constructed the following model for the encoding of all Sinhala letters in Unicode:

- A base Sinhala letter; i.e., a vowel or a consonant, is represented by a single code.
- A pure consonant, or a consonant with a vowel, is represented by two codes, one for the consonant, and one for the associated sign (vowel sign or *al-lakuna*).
- All conjuncts are represented by a sequence of letters joined by zero-width-joiner codes. All letters other than the last are pure consonants.

5.5 Non-Standard Letters

Some Sinhala letters are written in non-standard forms, and are represented by separate codes in current fonts, e.g., ළු. However, ළු is not a different letter, but simply the form of the letter *murthaja-la* with a *paapilla*. Therefore, it is represented by the sequence ළ + ු

Some *pili* take special forms when combining with some characters. For example: ක + ු = කු, ර + ු = රු. In these cases too, we took the position that the encoding should be based on the *logical* vowel sign, and not the shape displayed.

6. Fonts and Font Display

A font is a representation of the symbols in a script, for display on a screen or on a printer. Legacy fonts comprise a one-to-one mapping between codes and *glyphs*, i.e., the shapes displayed. However, the Unicode encoding for Sinhala (and other Indic languages) use a *sequence* of codes to represent each glyph. Therefore, a Unicode Sinhala font must be able to handle such mappings,
e.g., න + ් + zwj + ද + ් + zwj + ර = න්‍ද්‍ර.

Newer font technologies can handle such many-to-one mappings. One such technology, which is supported by both the Windows XP and Linux platforms, is OpenType [7]. As this technology is widely used, we decided to concentrate our font development efforts on OpenType.

One method of implementing a Sinhala font is to produce a glyph for each letter, such as කා, කෝ or ක්‍ෂා. The total number of such glyphs, if every possible letter is to be represented, will exceed 6000. The Lake House has implemented such a font, as it allows them to have full control over the appearance of each letter. However, such fonts are large in size.

Another approach is to have a glyph for each base letter, and to position the *pili* around it. This yields a very small font. However, in letters such as ටි or ගු, the *pili* are not simply placed above or below the letter, but form an integral part of the letter. Other letters such as ර් and ඊ, have the *pili* in non-standard positions. Additionally, since letters are of differing widths, the widths of the associated *pili* should also differ. For these reasons, although *pili* positioning may be suitable for applications such as mobile devices, it does not provide the quality required for print applications.

Therefore, most general-purpose Sinhala fonts take a hybrid approach, and include some glyphs with in-built *pili*, and some, e.g., the *kombuva*, ෙ and *aelapilla*, ා, as separate glyphs. This aproach leads to another problem, namely, that the *kombuva* visually precedes the consonant, but is logically stored following it, and that some vowel signs, e.g., in කො, surround the consonant on two sides. OpenType cannot handle such cases, which need operating system support.

6.1 Operating System Support

Older operating systems such as Windows 95 only supported 8-bit fonts. Newer systems, such as Windows 2000, support Unicode, but do not handle “complex scripts” such as Sinhala properly. Although Windows XP does support complex scripts, it does not correctly handle Sinhala.

Unicode support on Windows is provided by a module called the Uniscribe Unicode Script Processor. We collaborated with Microsoft to ensure that this incoprorated the features needed for Sinhala. The version of Uniscribe released with Office 2003 supports Sinhala, and correctly positions the *kombuva* before its base letter.

Linux systems employ a number of text handling mechanisms. The Pango library provides Unicode support in Linux, and is expected to support Sinhala.

Microsoft produced an OpenType Sinhala font named “Iskoola Pota”. We worked with Microsoft to ensure that this font incorporated the features specified in the SLS 1134 standard. This font is currently in Beta release.

7. Keyboard

Although text may be input to a computer in many ways, such as OCR, handwriting and speech, the most common method of text entry remains the keyboard. The 101-key computer keyboard has, with minor variations, become the standard keyboard which is used for many languages. Thus our objective in the keyboard area was to specify standards for Sinhala text entry on such keyboards.

We identified four types of keyboards for further consideration.

- The Wijesekera keyboard, which is used with both typewriters and computers
- “Phonetic” keyboards, in which key assignment is based on the English key layout.
- Transliteration schemes, in which text is typed as a sequence of English letters.
- Consonant-vowel sequence keyboards, in which the consonant is typed first, followed by the vowel modifier.

India has defined a keyboard layout, called Inscript, which provides a standard way to type all Indian scripts. Microsoft had developed a modification of Inscript for Sinhala as well. This was a consonant-vowel sequence keyboard.

The committee considered this keyboard, and was of the opinion that it did not meet our needs. This position was validated at the Sinhala Language in IT seminar held in June 2003, where it was unanimously decided to use the Wijesekera keyboard. Therefore, the committee decided to initially standardise the Wijesekera keyboard. Nevertheless, it was recognised that the Wijesekera keyboard had several shortcomings, and that we should design a more optimal Sinhala keyboard. Such a keyboard may well be a consonant-vowel sequence type.

It was also realised that the sequence of keys used to produce a given letter was independent of the keyboard layout, and that several layouts (e.g., a Wijesekera-based one and a phonetic one) may use the same key sequence.

7.1 The Wijesekera Keyboard

This form of keyboard was originally designed for manual typewriters. It has the feature that all of the *dead keys*, i.e., where the carriage does not move when the key is struck, are one the extreme left of the keyboard, due to design requirements of manual typewriters. Therefore, the *al-lakuna*, *ispili*, *paapili*, etc. are all on the leftmost keys. Also, as manual typewriters only support one language, no consideration was given for compatibilty with English keyboard. For example, the number keys were shited to the right by two places, and the puncuation marks appeared in different places.

We had two conflicting objectives in standardising the Wijesekera computer keyboard.

1. Retain compatibility with the Wijesekera typewriter keyboard.
2. Retain compatibility with the English (US-ASCII) keyboard.

In addition, we needed to be compliant with the Windows keyboard model, and allow efficient typing.

7.2 Design Principles

After considering several options, the committee decided on the following principles in the design of the Wijesekera-compatible keyboard.

1. Retain all common letters in the same places as the typewriter keyboard.
2. The 1st-row (number-row) keys (except the left-most) to be mapped to the same numbers and symbols as in the US-ASCII keyboard.
3. Have only one form of the *al-lakuna* and each *pilla* on the keyboard (the typewriter keyboard has separate keys for the different forms of the *al-lakuna* the *paapili*).
4. Do not have any “half letters” on the keyboard (as the first part of a *bandi akura*). *Bandi akuru* to be constucted pressing a *join* key between the two consonants.
5. The sequence of keys to be typed to produce a character be the same sequence as in writing;
   e.g., ෙ+ ක + ා+ ් = කෝ; ක + ්‍ර+ ි= ක්‍රි.

   Although this requires more keystrokes than in a consonant-modifier method (e.g., ක + ඕ-modifier = කෝ), It was adopted to retain compatibility with typewriting.

The keyboard layout was designed based on the above [2]. Most letters were retained on the same key as on the typewriter keyboard, but the symbols which were on the 1st row were moved to other keys. Although we would have liked to have used only the unshifted and shifted keys, as there are more letters and *pili* than the available (unshifted+shifted) keys, we needed to place some symbols elsewhere.

Possible alternatives were to place these symbols on control- keys or alt- keys. The standard followed by most European etc. keyboards, is to use the right-hand alt- (also known as ctrl-alt or alt-gr) key to enter additional characters, leaving the left-hand alt- key for applications [8]. We decided to follow the same convention.

Infrequently used letters such as ඏ (ilu) and kundaliya, ෴, were placed on alt-gr- keys.

In the typewriter keyboard, *sagngnaka* letters, such as ඟ and ඳ, were produced by pressing a special key before the corresponding “ordinary” letter. In order to make it easier to remember, it was decided to assign each *sagngnaka* letter to the alt-gr state of the same key as its “ordinary” letter; e.g., alt-gr-ග = ඟ. The total number of keystrokes to type a sagngnaka letter is thus the same as in the typewriter keyboard.

Keys were also assigned to the *yansaya*, ්‍ය*rakaransaya*, ්‍ර *rephaya*, ර්‍ and punctuation marks. We attempted to keep punctuation either in the same position as on either the US-ASCII keyboard, or the Sinhala typewriter keyboard, but were not completely successful. In particular, the single and double quotes are placed on the “Z” key, which is not too satisfactory

7.3 Other Keyboards

When designing the Wijesekera-based keyboard, it was accepted that better keyboard layouts and key sequences exist, just as the DVORAK keyboard layout is better than the QWERTY layout. However, it was recognised that attemting to design a new keyboard layout at this stage would not only consume more time, but also hinder acceptance among current typists who are familiar with the Wijesekera keyboard. Therefore, the design of an "optimal" keyboard for Sinhala was left for a later stage. Such a keyboard may well use a consonant-modifier keying sequence, as used in the Tamil99 keyboard.

Some current fonts, e.g. Kaputa, use a "phonetic" keyboard layout, in which each Sinhala symbol is mapped to the key of a similar-sounding English letter. This approach has the drawback that some Sinhala letters, such as ඤ, ඳ and ඹ, have no English equivalents. Also, the assignment of the *pili* is somewhat arbitrary.

Another approach is to use transliteration, in which a string of English characters (with or without capitals) is mapped to each Sinhala letter, e.g., chi = චි, bhoo = භූ. Samanala is one such scheme which has been used for many years. Many people use such schemes informally to send Sinhala e-mail, SMS, etc. to friends using Roman characters.

We recognised that many casual Sinhala users, who do not use a pre-printed Sinhala keyboard, need either a phonetic or transliteration-based keyboard. We did not, however, attempt to standardise is such a keyboard at this time.

8. Application Support

To use Sinhala effectively, applications should support the language. At the base level, applications should correctly input, store and display Sinhala text, possibly in conjunction with other languages. Conformance with Unicode simplifies this task, as any Unicode-compliant application will handle Sinhala without modification.

We tested a number of common applications such as web browsers, e-mail, etc. with the current system, and found that they handle Sinhala properly. However other applications, such as Microsoft Word, exhibited strange behaviour. Also, we observed that although Microsoft Access and Excel can handle Sinhala text, they had problems with sorting, etc. This is attributed to the incompleteness of the current Sinhala support, and is expected to be rectified soon.

The next stage is the localisation of applications, so that they display menus, help, etc. in Sinhala. A large amount of work needs to be done in this area.

The third level is Sinhala-aware applications and utilities, which "understand" Sinhala, such as spelling checkers, optical character recognition (OCR), etc. The development of such applications is expected to be accelerated by the adoption of standards for Sinhala.

9. Conclusion

Our initial objective in this subject was to increase the use of Sinhala on the Internet, and thereby make the Internet more useful to non-English speakers. However, whilst undertaking this work, we realised that our task was broader. Our present objective is to make using computers, and other devices such as mobile phones, in Sinhala as convenient and *obvious* as it is in English.

We believe that our work in the standardisation of the character encoding and the keyboard will assist in this task, but is only the first step.

The next step is to disseminate the language support, keyboards, etc. througout the country, and to build awareness among users, potential users, web developers and application developers, that this technology exists, and is advantageous. We have embarked on this process, and have started training and awareness programmes, as well as approaching hardware and software vendors.

We have also initiated a parallel programme to identify, and address, issues of using IT in Tamil.

The authors wish to thank the members of the CINTEC, Sinhala Fonts Committee, the ICTA Language Requirements Working Group, and the SLSI Sinhala Working Group, as well as all others who provided assistance in this endeavour.

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, Draft Sri Lanka Standard Sinhala Character Code for Information Interchange, available at http://www.fonts.lk/doc/sls1134.pdf , 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, අකුරු හා පිලි (Letters and Strokes) p. 53, Godage, 2000.

[7] Adobe Corp., An Introduction to OpenType, available at http://www.adobe.com/type/opentype/main.html .

[8] Michael S. Kaplan and Cathy Wissink, "Unicode and Keyboards on Windows", 23rd Inter-nationalization and Unicode Conference, March 2003.