Non-English ICT Computer Language Support

With Simple Solutions to Five Common Use Cases

By

Stephen Fierbaugh
<stephen_fierbaugh@tsco.org>
The Seed Company

Tags: ICT4D, literacy, non-roman scripts, unicode, utf-8, minority language development

Abstract

Android, Apple OS X and iOS, Linux (including Balsa and Wasta), and Microsoft Windows 7 and 8, all provide excellent support for reading documents in languages other than English. Most provide good support for creating documents in major languages. Support for languages of wider communication (LWC) is mixed. Support for minority languages is best provided through Keyman ($23.99 for Windows) or Multiling Omni Keyboard (free for Android). Apple iOS should be avoided for non-major languages.

Documents should be stored in UTF-8 Unicode, the default of most modern systems. Additionally, a language-appropriate virtual keyboard and language-specific font are required. Arial Unicode MS and Droid Sans are both safe fonts for any language; Gentium Plus is a good font for Latin-based alphabets.

The appendix includes simple flowcharts for five common Use Cases involving documents in non-English.
Non-English ICT Computer Language Support, With Simple Solutions to Five Common Use Cases

© 2014 Stephen Fierbaugh

International Standard Book Number: XXX-X-XXX-XXXXX-X

This work is distributed under the Creative Commons “Attribution-ShareAlike 3.0 United States” license. Limited duplication is permitted.

You are free to:

- to Share – to copy, distribute, display, and perform the work
- to Remix – to make derivative works

Under the following conditions:

- Attribution. You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work).
- Share Alike. If you alter, transform, or build upon this work, you may distribute the resulting work only under the same, similar or a compatible license.

For any reuse or distribution, you must make clear to others the license terms of this work. Any of the above conditions can be waived if you get permission from the copyright holder. Apart from the remix rights granted under this license, nothing in this license impairs or restricts the author's moral rights.

**Your fair use and other rights are in no way affected by the above.** This is a human-readable summary of the Legal Code (the full license), located at “http://creativecommons.org/licenses/by-sa/3.0/us/legalcode”.
# Table of Contents

Introduction .......................................................................................................................... 5

Key Concepts ......................................................................................................................... 5

Document Formats .................................................................................................................. 5

  The Bad Old Days .............................................................................................................. 5

  Unicode ............................................................................................................................... 6

  UTF-8 ................................................................................................................................. 6

  Future-Proof Documents .................................................................................................. 6

Keyboards .............................................................................................................................. 7

  QWERTY ............................................................................................................................ 7

Virtual Keyboards .................................................................................................................. 7

  Multistroke Characters .................................................................................................... 7

  Predictive Text .................................................................................................................. 8

  T9 ....................................................................................................................................... 8

  Swype ............................................................................................................................... 8

Fonts ..................................................................................................................................... 9

  Unknown Characters ....................................................................................................... 9

The Bad Old Days ................................................................................................................... 10

The Bad New Days ............................................................................................................... 10

Common Font Families ......................................................................................................... 11

  GNU Unifont ..................................................................................................................... 11

  Arial Family ....................................................................................................................... 12

  SIL Font Family ............................................................................................................... 12

  SIL Encore IPA and SIL IPA93 ....................................................................................... 13

  Modern SIL Fonts .......................................................................................................... 13

The State of Non-English Language Support on Major Platforms ........................................ 14

  Android ............................................................................................................................... 14

  Multiling Omni Keyboard ............................................................................................... 14
Balsa or Wasta?..................................................................................................................16

Microsoft Windows.............................................................................................................17
   XP........................................................................................................................................17
   Win 7...................................................................................................................................18
   Win 8...................................................................................................................................18
   Keyman..............................................................................................................................18
   InKey.................................................................................................................................19
   Ekaya.................................................................................................................................19

Appendix: Simple Solutions to Five Common Use Cases..................................................19
   I need to read a non-English document...........................................................................19
   I need to create a non-English document.......................................................................19
   What OS and font should my new project use?..............................................................19
   I need to read an old document containing garbled text...............................................19
   I need to edit an old document containing garbled text.................................................20

Bibliography......................................................................................................................21
Introduction

Support for using languages other than English on ICT devices (Information & Communications Technologies; ie computers and phones) has come a long way in the last few years. Once the domain of esoteric geeks, now on some platforms it is easy and on many others it is getting better.

Albert Einstein said, “Things should be as simple as possible, but not simpler.” This whitepaper assumes no prior knowledge of linguistics, and tries to keep its explanations as practical as possible. The appendixes covering common use-cases may be used – at least at a superficial level – without additional explanation. None-the-less, understanding at least three key concepts will significantly clarify and assist in making wise international computer decisions.

Key Concepts

There are three important concepts to understand about using non-Roman scripts with modern computers, tablets, and phones:

• Document Formats
• Keyboards
• Fonts

Each of these must work in tandem, or users end up with unreadable documents.

Document Formats

The study of characters as they are written down, printed, or typed into a computer is called orthography.

The Bad Old Days

All computer documents ultimately store words as strings of coded numbers. The formats may vary widely, but sooner or later at the lowest level, it's all numbers. A = 1, B = 2, C = 3, and so forth.

“Document” in this case means in the broadest sense possible and includes PDFs, Microsoft Word DOC files, LibreOffice ODT files, web pages, ParaText files, and anything else of interest on the computer. Unfortunately, all of the complexity comes from which number represents which letter?

The first computers were English and American-centric. 26 letters, double it to handle both upper and lower case, throw in some punctuation, and all the characters fit comfortably within 256 characters with room left over for smiley faces. Computers like numbers less than 256 because it lets all the characters possible be represented in one simple small number called a 'byte'.

The standard character set that is used in the United States is called 'Latin' and comprises all of the letters of the alphabet and punctuation commonly used in written English. In the past, the computer
version of Latin (or subsets of it) has also been called ASCII, ANSI, and Windows-1252.

Many western languages use Latin or some variation of it for their writing, but many languages from China, India, and other areas use non-Roman scripts. Some of these languages need thousands of symbols for their written orthographies. Additionally, many languages use mostly Latin, but include sounds and letters not found in English.

For a while, each software system created a different character set for each language, and assigned each character a different number at their own discretion. So in ASCII, the number 65 represented the uppercase letter 'A', but in Microsoft's character set of Chinese it represented something very different, and in IBM's version of Chinese it might mean yet another Chinese character. In IBM's Hindi, it meant yet another character. Many translation projects defined private character sets to deal with sounds and symbols not covered by any other system.

This led to big problems. Any software had to understand both the character set and the specific character representing every single character in a document. This was mass chaos, and many software makers didn't even try to support non-Roman scripts.

Unicode

Unicode is a standardization of all known written characters in a way that allows them to be used in computers. It assigns all characters (glyphs) in all languages a single unique number. Symbols used in English, Chinese, and Hindi all have different Unicode numbers, as do hundreds of other writing systems. Unicode also has support for accents, right-to-left and top-to-bottom writing, and many other important linguistic concepts. This reduces the confusion considerably.

All human languages together take about 100,000 symbols. Fortunately, Unicode supports over a million symbols, so when SETI is successful, we should have no trouble extending it to alien orthographies. Unicode future-proofs documents. A document using Unicode will mean the same thing and be just as readable today, tomorrow, and a hundred years from now.

UTF-8

There is still one more problem, though. How do we represent Unicode on the computer? There were several proposals. The simplest solution is to use one big number for each character. This worked well when computers only used 256 English characters which fit neatly in a single byte, but is wasteful for a million Unicode characters. The system becoming most popular is called UTF-8. It is elegant and efficient, representing common characters in small amounts of space, and rarer characters in larger amounts.

Future-Proof Documents

The important take-away is that documents that are stored in UTF-8 Unicode are readable on a wide range of systems and will remain readable far into the future. Fortunately, most new systems use UTF-8 Unicode as a default. Unfortunately, there are many legacy documents that are stored in a wide variety of other formats. The character set (or ‘code page’) and font that an old document was created in must be known in order for it to be read.
**Keyboards**

Computers generally have only a single physical keyboard. Most tablets don't have any keyboard, while if a phone has a keyboard, it is just as likely to be just a numeric keypad. Software must be able to map a physical keyboard (if any) to a specific keyboard layout which actually interprets what each physical key means.

Most computer operating systems support using one keyboard layout at a time, generally for the official language of the country. So Americans typically use an English keyboard, while a French speaker can map it to French, which is similar but slightly different. For non-Roman scripts, the differences are much larger. Keyboards don't physically have enough keys.

Virtual keyboard layouts solve this problem. They provide – in a variety of ways – for characters to be entered into a computer document. For the remainder of this document, 'keyboard' refers to the virtual keyboard layout instead of the physical keyboard.

**QWERTY**

The standard keyboard most westerners are used to is referred to as the QWERTY layout because of the order of the first row of alphabetic keys. This is not the most efficient layout (indeed, was specifically designed to deal with mechanical typewriters), but is ubiquitous because so many people have so much training and muscle memory invested in it. There are a number of variants on QWERTY, for different European languages and for different sized computer layouts (laptops vs desktops), but none of these is significantly different.

There have been a number of attempts over the last century to replace QWERTY with something more efficient, but none have become popular until quite recently. In the last handful of years, for the first time in over a century, there are several new layouts that have become popular.

**Virtual Keyboards**

Virtual keyboards must do more than just one-to-one mapping of a keypress on a physical keyboard to a single Unicode character. They must provide for a wide variety of accent marks, and allow some way to enter (for many non-Roman Scripts) thousands of characters more than there are physical keys.

**Multistroke Characters**

Most systems with a physical keyboard attached use some form of multistroke character input. There is a default virtual keyboard (which may be changeable on the fly, but at any one time there will only be a single keyboard; e.g. it can be English or French, but not at the same time) which controls how physical keystrokes are interpreted as characters.

When a character is needed that is not in the current default keyboard, a special *NULL* or “dead” keystroke is used in combination with one or more other keys. The NULL key does not result in a character, but rather changes the meaning of the other keys. Together, these keys combine to insert a single character that isn't physically on the keyboard.

As an concrete example, the IPA or International Phonetic Alphabet, is used by linguists to represent...
every sound that the human mouth is physically capable of producing. Much of the IPA reuses normal English alphabetic characters, but there are also symbols which do not appear in English. Using the KMFL keyboard software, the common English letters can be typed as normal. When the “ash” character æ (which should appear as a single character combining an 'a' and an 'e') is needed, the 'a' key is pressed, followed by the less than symbol '<', which is designated by KMFL as a NULL keystroke. Instead of the two 'a<' characters, the single character 'æ' is entered.

One important caveat: In Microsoft Windows, the NULL keystroke is typed first, followed by the other keystrokes, while on most other systems, the other keystrokes are typed first, followed by the NULL keystroke.

Multistroke Characters have the benefit of allowing experienced typers to quickly enter far more characters than would ever physically fit on a single keyboard. There can be multiple NULL keys, so they handle even large scripts efficiently. Most Multistroke Character systems also provide an on-screen virtual keyboard to help learners.

Predictive Text

Many virtual keyboards use Predictive Text. Predictive Text is a technique that tries to figure out in realtime with every additional key press, the top half-dozen or so words that could be meant based upon the given (potentially ambiguous) key presses. It pulls words from standard dictionaries, the user's addressbook (for proper names), and custom dictionary. Predictive Text also considers the user's past typing in ranking possibilities.

T9

T9 stands for 'Text on 9 Keys' and sparked the teenage text messaging craze. Phone keypads have long had three letters associated with each number, but using this to enter text in the past meant laboriously pressing a number multiple times to cycle through each possible letter.

Instead, T9 uses predictive text algorithms to determine which word is meant. Within only a handful of key presses, the correct word can usually be selected from the built-in dictionary.

T9 has a number of advantages: it works on even low-end feature phone hardware. It is easily extended to any alphabetic language. Dictionaries can be expanded dynamically by the user. It is simple to understand. An experienced T9 user (any teenager) can type on a feature phone keypad quite rapidly.

T9 gets unwieldy when extended to non-alphabetic orthographies.

T9 is primarily of interest because it is widely supported and extremely common on feature phones used in the developing world. Nokia has done significant work in the past to localize it for Languages of Wider Use around the world.

Swype

Swype is the first keyboard technique in over a century to gain widespread popularity as a direct replacement for inputing text by pressing each key on a QWERTY style keyboard. Like “Frisbee”, the
term Swype is commonly used both to refer to the general technique and to the trademark of Nuance Communications, the company who popularized the technique.

With Swype, a user simply traces their finger back and forth across an on-screen touchscreen keyboard, moving from letter to letter in a word without lifting their finger. Combined with predictive text, it is astonishing how quickly text can be entered. An experienced user can swype faster than most people can write on paper, but not as fast as an experienced typist can touch type on QWERTY. It is easy for even a new Swype user to achieve good speeds with little or no training.

Swype detects and crowdsources new words for addition to the Swype dictionary automatically. It is available in over 50 Languages of Wider Use.

Swype's predictive text is accurate enough that it is common to not bother glancing at the selected word to ensure it is correct. However, if there are two common words with similar keyboard pathing, this can lead to embarrassing mistakes called a “swypo”.

**Fonts**

AaBbCcDdEeFfGg  (Times New Roman)
AaBbCcDdEeFfGg  (Arial)
AaBbCcDdEeFfGg  (Courier New)

The three lines above contain the same characters. At the deepest level of this UTF-8 Unicode computer document, they all have the same computerized numbers representing them. The uppercase letters have a different number than the lowercase letters ('A', 65, is different from 'a', 97; 'B', 66 from 'b', 98, etc.), but the letters on all three lines are the same numbers behind the scenes: all three 'A's are 65 and all three 'a's are 97.

However, all three lines look different because they use three different common fonts to show the *glyph*, or precise visual shape and picture of each character. So several different fonts can all represent the same character intelligibly.

Notice that the spacing is slightly different on each line; all the letters in Courier New are the same width, reminiscent of a typewriter, while Times New Roman and Arial are proportional width, but also different from each other. Fonts must contain information such as how to figure out where to put accent marks in relation to characters. This can get extremely complex on non-Roman scripts.

**Unknown Characters**

Any particular font only contains glyphs for certain characters. For example, if Times New Roman didn't have a glyph for the number 65, then it could not display an uppercase A. Instead, it would display a generic mark to indicate a missing character. Depending on the system, it could be a blank box, a black diamond, a little question mark, or a box containing the hexadecimal code of the character.

This is a particular problem for documents written in other languages. If a font which only contains glyphs for Latin characters is used to try to display - for example - a Chinese document, the computer will have no way to draw the characters.
When a document is displayed and either has nothing but missing character symbols, or sometimes intelligible text mixed with missing character symbols, it means that the font being used to display a particular character does not contain a glyph for the character.

Most systems have both configurable font substitution and fallback fonts, which are used to display a character if the font specifically requested is not installed on the system. For documents written in western Latin languages, this works well to produce at least intelligible – if not pretty - content. However, if the fallback font does not contain a glyph for a character, it too will not be able to display it.

**The Bad Old Days**

In pre-Unicode days, to display a character correctly, a system had to know not only the character's number, but what character set or code page the number was based in. Often, systems did not strongly distinguish between the concepts of character set and font, or even language. If two different character sets reused the same numbers, then different fonts might show dramatically different glyphs for the same character, or simply not know what glyph to show.

There are many legacy documents which have serious issues being displayed, printed, or used in any way because of issues surrounding fonts. When dealing with legacy documents, it is the responsibility of the end-user to track down and install the correct fonts.

**The Bad New Days**

When dealing with new documents, it is the responsibility of the end-user to track down and install the correct fonts. (Only joking. Well, not exactly. Sort of. It's still not as straightforward as it could be.)

Most fonts commonly used today are designed for Unicode. They may (or may not) be a font intended for a specific language, but at least most include Latin even if it isn't the primary intended use, simply so that common western characters show up intelligibly. But many fonts also include glyphs for Unicode characters from many other languages.

There are two approaches to using fonts with many languages: A system can include many different fonts, one per language. This keeps fonts small and has the advantage of making crystal clear in the name of the font which font is intended to cover glyphs for which languages in Unicode. “Droid Arabic Naskh” is intended for use with Arabic while “Droid Sans Ethiopic” is intended for use with Ethiopian. The downsides are that each font is not interchangeable with the other because they probably don't have glyphs for each other's languages, and that the user is presented with a potentially huge (hundreds) list of fonts to choose between.

Or, a system can include only a handful of fonts, but have each font include many thousands of glyphs for all the Unicode characters. The downside of this is that it is a lot of work to create such a large font, and it is difficult to show any kind of a reference which helps the user figure out the different visual styles of print for a specific language. *Arial Unicode MS*, for example, contains nearly all Unicode characters, but is 22Mb in size, orders of magnitude larger than many smaller fonts. As even low-end phones have exploded in capabilities, this has become more of an issue of managing human complexity than of technological complexity.
It would be nice if there were one Sans-serif (plain; non-decorated), one serif (decorative), and one monospace font available as a standard which was guaranteed to be present on all systems and display all Unicode characters.

Unfortunately, there is still no standard – or even de facto standard – for what fonts are available on all systems. Microsoft has included a wide variety of different fonts at different times and versions of its Windows and Office products.

For a while, Microsoft freely distributed the “Core Fonts for the Web”. This included Andale Mono, Arial, Arial Black, Comic Sans MS, Courier New, Georgia, Impact, Times New Roman, Trebuchet MS, Verdana and Webdings. Any document using these fonts will almost certainly be readable on nearly any modern computer system. Microsoft has distributed the fonts widely in a variety of methods and with different distribution terms over the years, but is not currently distributing them (though a number of updated versions are shipped with various MS products). The free fonts have not been updated regularly and support for Unicode is mixed.

Google, too, has its “Google Fonts” (previously “Google Web Fonts”), which is a little different spin on the problem. This is an API and website that make it easy for websites to automatically display the correct fonts. All fonts included in Google Fonts are licensed under the SIL Open Font License. Google Fonts includes a number of SIL fonts.

**Common Font Families**

There are thousands of different fonts, many of which appear quite similar visually. Unfortunately, many of the fonts have extremely similar names. It is vitally important to recognize that even minor differences in names indicate completely different fonts which may (or may not) look and act similarly. To keep font names clear in this text, they will be italicized. Names of font families will be in normal print.

Another dark secret of the font world is that some of the most common fonts from direct competitors are actually licensed variations on the same font. Virtually every major computer vendor has some font that looks like Arial but are nominally independent of Microsoft's font from an intellectual property standpoint. In reality, most of these (including Microsoft's) were developed by the same company (Ascender/Monotype Imaging/Linotype). The good news is that this means there is a high degree of graphical overlap between them and most computer systems can map them automatically with few issues. The bad news is that different vendors licensed them at different times, so the precise glyph support for newer Unicode characters varies by vendor and version.

There are several families of fonts that are of particular interest to international users.

**GNU Unifont**

*GNU Unifont* – Roman Czyborra. GNU Unifont is a single font rather than a family. It contains more glyphs than any other font, even *Arial Unicode MS*. It has complete coverage (~57000 glyphs) of the Unicode Basic Multilingual Plane, including essentially all currently assigned Unicode characters relevant to translation. It is licensed under the GPL, and anyone can modify or redistribute it. Unfortunately, it is a bitmapped font, which prevents it from being appropriate for most publishing (ie
this font is ugly at anything except 8x8 or 16x16 resolution).

**Arial Family**

Arial is one of the most common fonts in the world. Since 1992, the basic version has shipped with Microsoft Windows, and recent versions of Apple OS X. It is a plain sans-serif font with very simple characters. It is based on a pre-computer font, Monotype Grotesque, and is similar metrically to Helvetica. It looks good but not great in a wide variety of uses, formats, and media. There are many different variants on basic Arial.

**Arial** – Microsoft (by Ascender Corp. principals). The plain version of Arial has support for most Latin Unicode characters, Hebrew, Arabic, IPA, and Greek, but not extensive support for other languages (~4000 glyphs).

**Arial Unicode MS** – Microsoft (licensed from Monotype). This font extends Arial for as many Unicode characters as possible. It appears to be the most complete set of Unicode glyphs in any one non-bitmapped font (> 50,000). Arial Unicode MS has been widely distributed over the years, including with Microsoft Office. It is also bundled with Mac OS X and is available for purchase from Ascender Corp. It is not, however, distributed with Microsoft Windows. It is available for free download from [http://fontsgeek.com/fonts/Arial-Unicode-MS-Regular](http://fontsgeek.com/fonts/Arial-Unicode-MS-Regular).

**Liberation Sans** – Red Hat (licensed from Ascender Corp.); Google (licensed from Ascender Corp.), released under GPL and SIL Open Font License. Has support for most Latin Unicode characters, but not extensive support for non-Roman scripts. Some versions of this font are also referred to as Arimo. License confusion has made Liberation Sans more difficult to extend than anticipated, and it is gradually being replaced in popularity by Droid Sans.

**Droid Sans** – Android (Open Handset Alliance; Google, licensed from Ascender Corp.), released under Apache License. Similar to Liberation Sans. Droid Sans is easy and free for anyone to add glyphs to, so it boasts extensive Unicode support. Droid Sans Fallback supports over 43,000 Unicode glyphs, nearly as many as Arial Unicode MS. Originally intended specifically for mobile devices, it is quickly becoming a popular font because of its generous licensing terms.

**Roboto** – Android (Google), released under Apache License. Because Droid Sans wasn't ugly enough for Google. Roboto is functionally similar to Droid Sans, but reportedly has issues with accented glyphs.

**Ubuntu** – Ubuntu (Canonical, licensed from Dalton Maag), released under the Ubuntu Font License. Excellent Latin Unicode support, and a growing number of non-Roman languages.

There are many other less well known variants on Arial from many different companies and organizations. Most, but not all, are descended from Ascender Corp. They are often downloadable under permissive licenses of various kinds.

**SIL Font Family**

SIL has been at the forefront of internationalized font development for decades. It has published many excellent and well-known fonts, for a number of different linguistic families. All are available for free.
Prerelease Courtesy Copy

January, 2014


SIL Encore IPA and SIL IPA93

SIL Doulos

SIL Sophia

SIL Manuscript

These were the first international fonts widely available and they were extremely popular in their day. There are many important legacy documents which were created using these fonts. These fonts appear visually superficially similar to Doulos SIL and Charis SIL (notice the similar names, as well), but THEY ARE NOT COMPATIBLE. They do not provide Unicode support and characters are mapped to different numbers. In fact, they date to 1993. SIL says,

IMPORTANT: The SIL Encore IPA and SIL IPA93 fonts are obsolete, symbol-encoded fonts. Their use is discouraged. If you decide to download and use these fonts, please note there is no user support for these fonts. If your university or organization requires the use of these fonts, please request they change their requirement to Doulos SIL, a Unicode-encoded font which contains the complete IPA repertoire. (http://scripts.sil.org/cms/scripts/page.php?item_id=encore-ipa)

SIL cannot stop providing downloads of these fonts because they are required for any legacy documents that use them. However, never use these fonts for new documents. Despite SIL's website plea, as recently as 2013 there were still college courses using them. If your instructor requires these fonts, the course material is extremely (20 years!) out of date.

Modern SIL Fonts

The modern SIL Unicode fonts have focused on Latin and the IPA. They are of exceptional quality graphically and provide good support for the state of the art in rendering technology.

Charis SIL – SIL, released under the SIL Open Font License. A serif (decorated) font, it is designed to provide good readability under less than ideal field conditions. Charis SIL’s goal is to “provide a single Unicode-based font family that would contain a comprehensive inventory of glyphs needed for almost any Roman- or Cyrillic-based writing system, whether used for phonetic or orthographic needs.” It contains > 4500 high quality glyphs for all Latin Unicode characters. Charis SIL is the recommended font of the International Phonetic Alphabet. (Note, however, that many other fonts also provide good IPA support.)

Doulos SIL - SIL, released under the SIL Open Font License. A serif (decorated) font visually similar to Times New Roman, Doulos SIL's goal is to “provide a single Unicode-based font family that would contain a comprehensive inventory of glyphs needed for almost any Roman- or Cyrillic-based writing system, whether used for phonetic or orthographic needs.” It contains > 4500 high quality glyphs for all Latin Unicode characters.

Gentium Plus – SIL, released under the SIL Open Font License. A serif (decorated) font, it is designed
to be easy for newly literate readers to learn. It has also won awards as a beautiful font. It contains > 5500 high quality glyphs for all Latin, Greek (including archaic), Cyrillic, and IPA Unicode characters.

The State of Non-English Language Support on Major Platforms

The most recent versions of all major operating systems provide good support for internationalization in languages of wider use. Each of them (with the notable exception of Microsoft Windows 7 & 8) ship with at least one font providing strong Unicode support. Most include support out of the box, or with easy installation, for fully localizing menus, dates, currency, etc. into a limited number (typically < 50) of commercially important languages. And each provides keyboards of various kinds in a somewhat larger (typically < 200) but still limited number of additional languages.

Details are below, in alphabetic order.

Android

Android is Google's operating system for smartphones and tablets. In *Emerging ICT Trends Affecting Progressive Publication of Newly Translated Scripture*, Fierbaugh points out, “Android is expected to solidify at between 80-85% of all smartphone sales in the coming year.” (p15) In the developing world where most minority languages are located, this is much higher, with nearly all smartphone and tablet sales (> 95%) being Android.

Android is a newer operating system and it supports Unicode at even the lowest levels. Additionally, its built-in font, Droid Sans Fallback, provides exceptional glyph support for nearly all language orthographies. In short, everything in Android occurs in a fully internationalized UTF-8 Unicode-based format unless explicitly requested otherwise.

One of Android's strengths is its customization, and Android has many easily installable keyboards for every conceivable need. Google has an “official” keyboard for Pinyin entry of Chinese characters. Swype, by Nuance Communications, is also popular and has advanced the state of the art in keyboarding significantly.

Multiling Omni Keyboard

For international keyboards (both Latin and non-Roman), however, the defacto standard on Android is Multiling Omni Keyboard, by Honso. It is used by millions of active users daily. Multiling supports hundreds of different scripts and languages, second only to Keyman.

Multiling is free, runs well on low-end hardware, and supports swype-style gestures. It is easy to install by simply searching for “Multiling” in the Google Play store. It has user-friendly tutorials and documentation. Adding additional supported languages is a snap, and it has strong built-in crowdsourcing to help create keyboards for unsupported languages.

Apple

Note: In *Emerging ICT Trends Affecting Progressive Publication of Newly Translated Scripture*,
pp36-39, Fierbaugh describes four non-linguistic reasons iOS is often inappropriate for developing world projects:

1. In the developing world, iPhones are used only by the rich.
2. iPhones do not allow downloading of media through the browser.
3. iPhones do not support SD cards.
4. iPhones do not allow sharing of media via Bluetooth.

There is no equivalent to KeyMan, MultiLing, or Swype on Apple products. KeyMan has released a special version for iOS, but it is not a general purpose keyboard replacement; text is entered into the KeyMan text editing area and then copy & pasted to other applications.

**OS X (Mac)**

OS X has excellent Unicode font support because it ships with Arial Unicode MS standard and pre-installed. For languages which Apple supports, OS X has easy switching between keyboards. For minority languages, keyboards may be added, but it is more work.

**iOS (iPhone & iPad)**

iOS has good Unicode font support. For languages which Apple supports, there are keyboards provided. Apple strictly controls iOS and forbids alternative keyboards. This has hamstrung any attempts to add data entry for minority languages. Because of these restrictions, KeyMan's iOS application provides a way to enter text into a special text editing area, but then requires copy & paste to move it to other applications.

Apple's latest iOS keyboards for entering non-Roman scripts have received mixed reviews (Willis, *Apple confirms 10-key Pinyin keyboard only for iPhone in HK and China*).

**Linux**

Linux has a broad international development base, so it provides excellent internationalization. Both the core operating system and its included software such as LibreOffice and Firefox tend to have more localized languages than other systems, and make it easy to add additional ones.

Unicode support has been standard for some time. Modern distros come with Droid Sans for displaying nearly all Unicode glyphs within a single font, but also ship with a dizzying array of hundreds of other fonts for specific languages and scripts. Most distros include Charis SIL, Doulos SIL, and Gentium Plus or make it trivial to install them through Yum or Synaptic (the free Linux "app stores").

While there are many legacy options, iBus (Intelligent Input Bus) is the modern way for Linux systems to provide international keyboards. iBus supports many different keyboard input methods for non-Roman scripts. The ibus-m17n extension supports extremely complex scripts which require knowledge of states and surrounding text.
KMFL, which interfaces to iBus, is an option which allows KeyMan format keyboards to be used directly with Linux. This is legal, and initial development was by Tavultesoft (KeyMan), using funds provided by SIL.

**Balsa**

[http://balsa.kkoncepts.net/](http://balsa.kkoncepts.net/)

Balsa is a special purpose Linux distro optimized for linguistic and translation work. It is designed to be easy to learn and use, including by default just the features which translators need to do their job. Balsa comes "out of the box" with ParaTExt, the SIL Unicode fonts, Pidgin, Thunderbird, and a typing tutor. It includes everything necessary for international fonts, keyboards, etc. pre-installed.

The major advantages of Balsa are:

1. No viruses! Great security. Under the covers, Balsa is a custom version of Ubuntu.
2. Easier training. Balsa is designed for users who may have minimal education. There aren't a lot of cluttered icons, and all images are large and easy to read.
3. Easier tech support. If it "breaks", throw away the SD card or USB drive and pop in a new one. All important data resides on both the hard drive and the SD card/USB drive.

Balsa is ready for production use on translation projects. It is undergoing active ongoing development.

**Wasta**

[http://itconnect.info/group/linux-desktop/page/wasta-linux](http://itconnect.info/group/linux-desktop/page/wasta-linux)

Wasta is similar to Balsa in that it is a complete operating environment for translators. It shares the malware security advantages of Balsa, but is closer to a Windows "look & feel". Wasta is based on Mint and has the SIL repos (app store), but does not ship with ParaTExt, FieldWorks, Adapt It, or KMFL preinstalled. Installing any of them from the "app store" is not difficult, however.

**Balsa or Wasta?**

Balsa and Wasta are both intended for Mother Tongue Translators (MTTs), but at different levels:

- Wasta looks "Windows-ish" and is geared towards typical African MTTs with at least a passing familiarity with MS Windows and perhaps a high school education.
- Balsa looks "Unity-ish" and is geared towards typical Pacific MTTs with little or no familiarity with computers and perhaps a middle school education.

Balsa is based on Ubuntu while Wasta is based on Mint, so apart from the user interface look & feel, they are similar under the covers. Both Balsa and Wasta are impervious to the malware which plagues translation projects, and are good choices for projects where there will be little technical support available.
Microsoft Windows

Microsoft Windows support has gotten better and better over the years. Windows 7 and Windows 8 both have good Unicode support. The primary problem has been licensing. Microsoft has had confusing and varied licensing for which specific versions of Windows were able to use different international language features. This has been resolved in Windows 8, with all versions of Windows 8 supporting Language Packs.

For some reason, the Microsoft font Arial Unicode MS, which has outstanding Unicode support (and ships with Apple OS X), does not ship with any Windows operating system. It does, however, ship with Microsoft Office and a variety of other misc. software products. This leaves font support unnecessarily spotty on Microsoft Windows platforms. Arial Unicode MS is often present, but not consistently or reliably so.

The Microsoft Keyboard Layout Creator (MSKLC) provides the ability to create custom keyboards not already supported by Microsoft. It is an older 32-bit application distributed by Microsoft and comes with no support. It provides simple 1:1 key remapping (essentially just remapping any key to another Unicode symbol), and does not support any kind of complex scripting. It does support basic dead-key combinations. It could, for example, recreate the IPA keyboard, but would struggle with complex non-Roman scripts.

Windows, Office, and a few other Microsoft products use Language Packs to completely localize the user interface in various languages. Language Packs are polished and professional renditions of these products' dialogs, menus, etc. into other languages. Language Packs are exclusively for languages with large commercially important populations. The number of Language Packs has gradually gone up with each release of Windows.

Not to be confused with Language Packs, Language Interface Packs, or LIPs, are provided through the Microsoft Local Language Program in cooperation with local governments and businesses. LIPs tend to be less polished, but have all the basic words and phrases to use Windows or Office in a language. While Microsoft's marketing implies that these are tiny languages in danger of going extinct, almost none of them are what linguists think of as minority languages. Most are Languages of Wider Communication.

Microsoft's Language Portal also offers terminology word lists for technology key terms and technical writing style guides for languages for which Language Packs or LIPs exist.

XP

Microsoft Windows XP was not originally released with non-Roman script support, and with only limited support for non-English European languages. It has been added to some versions in some geographies in a variety of different ways and with different license terms. Because of XP's age and security issues, projects should immediately upgrade to newer systems. Existing legacy XP hardware can often use Balsa, Wasta, or Mint effectively, with a huge corresponding increase in both security and productivity.
Win 7

Windows 7 supports 35 languages fully through Language Packs and about 55 more at least partially through the addition of LIPs. Unfortunately, licensing is frustrating, as follows:

- No language is available as both a Language Pack and a LIP.
- Only Ultimate and Enterprise editions of Windows 7 may use Language Packs (ie top-35 languages).
- Any Genuine copy of Windows 7 Starter, Home Basic, Home Premium, Professional, Enterprise, or Ultimate may use a LIP (ie a secondary language of wider communication).

(Sources:  
http://msdn.microsoft.com/en-us/goglobal/ee461121.aspx#Windows7editionsmultilingualsupportsummary and  

One common problem is users of U.S. English Windows 7 Professional who wish to use a Language Pack such as Arabic, Chinese, French, or German. There is no solution except purchasing an otherwise unnecessary Ultimate or Enterprise edition license.

Win 8

Windows 8 brings some significant changes to how Microsoft licenses, ships, and supports alternative languages on Windows. It centralizes language settings in the Control Panel and allows any Windows 8 version to download additional Language Packs. Windows 8 supports 109 languages on a per-user basis (different users may have different languages).

Keyman

http://www.tavultesoft.com/keyman/

Keyman is a commercial software program for Microsoft Windows published by Tavultesoft. It supports over 1000 languages with virtual keyboards, fonts, and enhancements to Windows' support for other languages. Keyman makes life much easier for people using Microsoft Windows with non-Roman scripts.

Keyman is the slickest and easiest to use solution for international computing in general. It has a well-done installer and plentiful help. The only software on any platform that even begins to come close to Keyman is Multiling Omni Keyboard for Android (though Multiling has vastly more daily users than Keyman).

For some time, Keyman was free for non-commercial use. However, this policy has changed and now Keyman is $24 for Keyman Desktop Lite, or $70 for Keyman Desktop Professional. Substantial (huge!) discounts are available for translation-related non-profit use.

Charging even a tiny amount has proven to be a stumbling block to widespread use by the newly
computer literate, and there seems to be a move away from Keyman for non-professionals as built-in Unicode support has gotten better in the various operating systems.

**InKey**

InKey was an open source attempt at a free KeyMan competitor. While a few people love it, it appears to be abandonware.

InKey is currently at version 0.301. The InKey Tutorial states, "InKey 1.9 is alpha software, not considered stable."

**Ekaya**

Ekaya is a backport of Linux-based KMFL to Microsoft Windows. Its primary purpose is to allow KeyMan format keyboards to be used on Windows without requiring an actual purchase of KeyMan. Ekaya is legal open source software. It is not under active development.

For a number of years, there was an agreement between Tavultesoft (KeyMan), which created Linux-based KMFL, and SIL, which funded it, not to backport KMFL to Windows, so SIL did not forbid Ekaya's use on its projects, but also did not encourage or endorse it. Marc Durdin, of Tavultesoft, lifted this restriction in an email to the Linguistics & Translation Software mailing list dated 2 December 2013. Keith Stribley, who was not a member of SIL, was the primary Ekaya developer, and passed away around 2009.

**Appendix: Simple Solutions to Five Common Use Cases**

[INSERT CommonNonEnglishICTUseCasesFlowcharts.odg HERE.]

*I need to read a non-English document.*

*I need to create a non-English document.*

*What OS and font should my new project use?*

*I need to read an old document containing garbled text.*
I need to edit an old document containing garbled text.
Bibliography


Microsoft. (2013, December 19). *LLP.Lang_Offerings_List.xlsx*


Microsoft. (n.d.). *Local Language Program*

Microsoft. (n.d.). *Local Language Program - Frequently Asked Questions*


