8.3 Bricks and Mortar for Digital Resources in Indian Languages

Gora Mohanty

Abstract—**The creation of resources for building Indian language content in the online world needs various kinds of support technology. Over the years, free software tools have matured enough to provide a stable and easily usable base for supporting Indian languages in a standardized, platform- neutral manner. I will present a complete set of such tools that can immediately meet needs for content generation in most major Indian languages.The talk will cover text input methods, spellcheckers, sorting and locale issues, font converters, transliterator, databases, and text-to-speech, and focus on a live demonstration of such tools. I will also emphasize the importance of free software, and of open document formats in this area.**

I. INTRODUCTION

TILL recently, computers have by and large been easily usable only in English, and, to some extent, other European languages. What Indian language software has been available has usually operated in a non-standardised manner, so that severe issues have arisen with the portable use of content created with such tools. Things have changed over the last few years, with improved support for at least the major Indian languages, and with the advent of user interfaces in native Indian languages.

Open-source operating systems, and applications, have traditionally held a lead in supporting multi-lingual features, and support for international standards like Unicode [29], and for the complex text handling required by Indian languages has also matured. This article discusses the open-source tools available for content creation and management in Indian languages, covering both minimal requirements like fonts, input methods, and rendering, as well as advanced tools like spell-checkers, and text-to-speech. It also explains the importance of free software in this context, and outlines possible ways of spurring the creation of online content.

II. FREE SOFTWARE IN THIS CONTEXT

Open-source software provides four basic rights to the end-user, namely the right to use, study, modify, and redistribute the modifications, whereas closed-source software provides only the right of use. These rights have some far-reaching consequences. The ability to study the software lets new developers learn programming techniques from the world's top experts. The right to modify, and redistribute the software ensures that one is not left at the mercy of any single organisation. The net result is to commoditise software, ensuring that it remains economical. As it is not feasible in an open-source environment to continuously milk consumers by charging an arm and a leg for each new version of an application, the commercial focus of companies changes to providing high-quality support.

In the context of localisation, open-source software complies better with standards, and, in particular, there is system-wide compliance with Unicode. There is also a built-in internationalisation framework that allows easy translation to local languages, so that almost every open-source application nowadays permits localisation. These advantages have been recognised even by hard-nosed economists, e.g., the conservative Economist magazine [13] noted back in 2003 that open-source desktop interfaces were available in more than twice as many languages as Windows XP, a lead that has since widened. Licensing costs for open-source software are also typically zero, which is a huge benefit for disadvantaged portions of society, as well as for non-profit institutions such as educational organisations, NGOs, and government bodies.

Finally, one of the interesting aspects of open-source software development is that it encourages a community-based effort, as the end-user now can, and in fact is actively encouraged to participate in the development process, rather than being a passive consumer. This spirit of sharing, and joint effort, is important if one seeks a bottoms-up, rather than a top-down approach. It is not enough to simply hand out free-of-cost copies of software in the name of bridging the digital divide. One has to ensure that the linguistic community that is supposedly being served by such software becomes an integral part of the process, and actually make use of the basic tools to create things of lasting value.

III. ESSENTIAL SOFTWARE

There are four indispensable items for an Indian language to be supported by an operating system:

- System-level support for Unicode, and rendering: Unicode support has been available on Linux for a while, and extends to all levels of the support from the base C libraries, glibc, to X and various rendering engines. The complex features required for the rendering of Indian language text are usually provided through font technologies like OpenType [2, 10, 9], developed jointly by Adobe and Microsoft, and also supported on newer Apple computers. OpenType for Indic languages is supported in Linux through various renderers like ICU [8], IndiX [3], Pango [23], and QT [25], though it should really be integrated into the base X GUI layer.

Gora Mohanty is with Sarai, CSDS, 29 Rajpur Road, New Delhi (e-mail: gora@sarai.net)

- Fonts: An OpenType font, with Unicode encoding, is required for rendering, printing, etc. A variety of such fonts, covering most major Indian languages, are available under an open-source licence, e.g., such as the one listed under the IndLinux project [1].
- Input methods: Here, we confine our attention to keyboard input methods, though in the long run, other forms of input, such as handwriting recognition, or speech-to-text might be more useful in an Indian context. Rather than making custom keyboards for each Indian language, it makes more sense to simply map characters in existing English keyboards to Indian language equivalents through software. Under Linux, this mapping can be done through various applications, but we will focus on only two of these: (a) xkb [5, 31, 7] provides a very low-level mapping of the keyboard, interfacing directly with the base X GUI layer, and (b) SCIM [26] that provides many more possibilities than the simple one-to-one mapping scheme of xkb. SCIM is probably the future of open-source keyboard input methods.

  There are two broad classes of keyboard maps in use for Indian languages: (a) pseudo-phonetic ones like ITRANS [4], Bolnagri (a xkb-based phonetic map), etc., and (b) non-phonetic ones like Inscript, and Remington (typewriter) keymaps, which aim at efficiency in typing.
- Locales: A locale defies culturally-sensitive information, including things like names of months, days of the week, currency symbols, sorting order, etc. Locales are usually system wide, such as the locales for various Indian languages that come bundled with glibc, though some applications, like OpenOffice, prefer to use their own.

The support for these items need to be available at a very low-level in the system, so that applications share a common code-base, making it easier to resolve any bugs and deficiencies in the software. The old practice of each application rolling its own support for Indian language features leads to a myriad issues with non-standardisation, and incompatibility with other software.

Position	Hindi	Default Eng.	Best Eng.
Not found	5%	6%	2%
1	71%	59%	60%
1-5	91%	86%	83%
1-10	94%	91%	90%
Any	95%	94%	98%

Table 1: Performance of aspell in spell-checking Hindi, and English.

In addition to these, the following items are also important for a modicum of comfort in working in Indian languages on the computer:

- Spell-checking: The GNU spell-checking engine, aspell [18] now includes support for documents in UTF-8 (Unicode). Dictionaries for many Indian languages are also available [6], though they are far from comprehensive at the moment. aspell has many nice features, such as phonetic rules and affix specifications, that make it well-suited for spell-checking in Indian languages.

  We have recently gone through the exercise of customising aspell for Hindi, with the aim of also using this as a case study for other Indian languages. Table 1 compares the performance of aspell in Hindi against the default English method, and also against the best procedure for English (this is not used by default as it is more time-consuming). A test list of deliberately mis-spelled words was run through aspell for each language, and the performance was measured by (a) counting the percentage of words for which the correct replacement was suggested, and (b) the position of the correct replacement in the list of suggestions returned by aspell. It can be seen from the table that the performance for Hindi typically exceeds that for English. This is not as surprising as it might be at first glance, as Hindi is spelled phonetically.
- Sorting: A default sorting order for Indian languages has been defined in the glibc locales, but has been tested only for Hindi, and Oriya. A more rigorous testing should be undertaken, and incorporated into the Unicode Common Locale Data Repository [28].
- Printing: Printing in Indian languages now works from GNOME/KDE applications, as well as from browsers like Mozilla, and Firefox.
- Desktop software: Almost any desktop application required for homes, and businesses can also be used in Indian languages, under Linux. This includes office suites like OpenOffice [11], browsers like Firefox [17], mail clients like Zimbra [30] or Mozilla Thunderbird [21], databases like PostgreSQL [24] or Mysql [22] with front-ends like oobase, the OpenOffice database component, or rekall [12].
- Font converters, for content in legacy 8-bit fonts: Before the advent of standardised encodings like ISCII or Unicode, software for creating Indian language content used non-standard, 8-bit font encodings, so that there is a large body of existing content in these formats, making it useful to have a means of conversion to Unicode. There is a Firefox plugin, Padma [20], that does on-the-fly conversion of online content from these legacy encodings to Unicode, and work is under way to have a general-purpose library, and utilities, to handle such conversion tasks.

- Dictionaries / glossaries: aspell includes dictionaries for several Indian languages, but these are far from being large enough, and comprehensive enough. Likewise, glossaries of technical terms have been prepared for use in translations of GNOME, KDE, OpenOffice, etc., but a lot more work is needed here.

IV. ADVANCED SOFTWARE

Besides the basic tools listed in the previous section, work is at various stages of progress on advanced software that can help in Indian language content creation, and dissemination:

- Text-to-speech: Open-source software is now available that can convert electronic text into speech. This is useful, for example, to allow visually handicapped, or illiterate, people access to online content. There are two main open-source applications that can synthesise speech in Hindi, and a few other Indian languages: (a) Festival [16] is the better-known system, which has modules for Hindi, Marathi, and Telugu, and (b) eSpeak [15] which uses a different technique to synthesise speech. For both applications, there remains a fair amount of work in getting the Hindi speech synthesis to work accurately.
- Speech-to-text: The opposite problem, of the recognition of speech, and its conversion into electronic text, is a significantly more difficult. Among the various open-source speech recognition programs, perhaps the best-known is CMU Sphinx [14].
- Optical character recognition (OCR): This is another area which needs a lot of work, especially in the open-source domain. Of the open-source OCR engines, perhaps the most promising ones are GOCR/JOCR [19], and Tesseract [27].

V. PROCESS OF CONTENT DEVELOPMENT

The applications described here constitute only the essential tools, based upon which the real work of content development in Indian languages can be taken up. The modalities of how the development of meaningful content in Indian language will come out remain to be seen, and, in an open-source world, will eventually be driven by the users of the system themselves, but here are some ideas on how to spur the growth of online content in Indian languages:

- Change focus from technological development to deployment of existing solutions, and promotion of usage: The current set of tools in the open-source world are already good enough for creating Indian language content. The ongoing work of translation of the user interface of the applications into the local language will render them suitable for use even by non-English speakers. Thus, it is time now to put these tools in the hands of actual users, thereby empowering them to create content on their own.
- In the immediate term, the quickest way to build a community of local language users is by building sites that cater to specialised interests. This could include items like blogs, news and search engines that work in Indian languages, and provide facilities like email to users. A good example of this is http://sampada.net which started out as a Kannada localisation site, but has slowly transformed into a focal point for literary writers, and other people interested in Kannada literature.
- The other immediate area that can be targeted is the provision of these tools to vernacular educational institutions, and establishments that publish in Indian languages; people that already have a need for software in Indian languages, and who are already in the business of content creation.
- Wikipedia (http://en.wikipedia.org) is a very interesting project to create a free encyclopedia based solely on contributions from users. It is founded on principles similar to those of the free software movement, and has been a resounding success, being comparable in quality to top-of-the-line commercial encyclopedias like the Encyclopedia Britannica. Indian language versions of Wikipedia also exist, for example, the Hindi one (hi.wikipedia.org). Thus, the building of an active community of Indian language contributors to such an enterprise would be highly beneficial.

REFERENCES

[1] A list of Indic fonts, with Unicode encoding. http://indlinux.org/wiki/index.php/IndicFontsList.

[2] Adobe Open Type fontpage http://www.adobe.com/type/opentype/main.html

[3] An Indian language compilation of GNU/Linux software. http://www.cdacmumbai.in/projects/indix/.

[4] An Indian language transliteration package. http://www.aczoom.com/itrans/.

[5] Description of the X keyboard extension. http://netadmin1.ic.tsu.ru/en/xkb/.

[6] Dictionaries for GNU Aspell. http://aspell.net/XXX.

[7] Enhancements to xkb configuration. http://www.xfree86.org/current/XKB-Enhancing.html

[8] ICU is a widely used set of C/C++ and Java libraries for Unicode support, software internationalization and globalization. http://www.306.ibm.com/software/globalization/icu/index.jsp. The Sourceforge ICU project is hosted at http://icu.sourceforge.net/.

[9] Microsoft FAQ on OpenType. http://www.microsoft.com/typography/faq/faq9.htm.

[10] Microsoft OpenType font specifications. http://www.microsoft.com/typography/specs/default.htm

[11] OpenOffice.org is an open-source, multi-platform and multi-lingual office suite, compatible with all other major office suites, and is free to download,use, and distribute.http://www.openoffice.org.

[12] Rekall, the database management system. http://www.thekompany.com/products/rekall/.

[13] Open source's local heroes. Multi-lingual support in open source software. *The Economist*, Dec. 4 2003. Online at http://www.economist.com/science/tq/displaystory.cfm?story_id=2246308. Subscription might be required.

[14] The CMU Sphinx Group Open Source Speech Recognition Engines. http://cmusphinx.sourceforge.net/html/cmusphinx.php.

[15] The eSpeak open source software speech synthesizer. http://espeak.sourceforge.net. eSpeak uses a different approach to synthesising speech than other open-source text-to-speech engines. A Hindi module is in the process of refinement.

[16] The Festival speech synthesis system. http://www.cstr.ed.ac.uk/projects/festival/. A Hindi module for Festival can be obtained from http://janabhaaratii.org.in:9673/indicbhaaratii/Members/Priti_Patil/festival-hi-0-1-tar.gz.

[17] The FireFox browser homepage. http://www.mozilla.org/products/firefox/.

[18] The GNU Aspell homepage. http://aspell.net.

[19] The GOCR program, also known as JOCR. http://jocr.sourceforge.net/.

[20] The homepage of the Padma Firefox extension. http://padma.mozdev.org. Padma does on-the-fly conversion from 8-bit Indian language fonts to Unicode.

[21] The Mozilla Thunderbird mail client. http://www.mozilla.com/en-US/thunderbird/.

[22] The Mysql database. http://mysql.org.

[23] The Pango project. http://www.pango.org.

[24] The PostgreSQL database. http://www.postgresql.org.

[25] The Qt cross-platform application development framework. http://www.trolltech.com/products/qt.

[26] The Smart Common Input Method (SCIM) platform project. http://www.scim-im.org/. Provides a user-friendly input method interface for POSIX-style operating systems, including a platform to make input method development easier.

[27] The Tesseract optical character recognition engine. http://code.google.com/p/tesseract-ocr/. This is based on code that was developed at HP Labs between 1985 and 1995, and has now been open-sourced. Most of the current work on Tesseract is sponsored by Google.

[28] The Unicode Common Locale Data Repository (CLDR). http://www.unicode.org/XXX.

[29] The Unicode Consortium. http://www.unicode.org.

[30] The Zimbra mail server. http://www.zimbra.com.

[31] K. Toman and I.U. Pascal. The xkb configuration guide. http://www.xfree86.org/current/XKB-Config.html

This paper was presented at LRIL-2007: National Seminar on Creation of Lexical Resources for Indian Language Computing and Processing at C-DAC Mumbai (26th to 28th March 2007), jointly organized by the Commission for Scientific and Technical Terminology (CSTT), New Delhi, MHRD, Govt. of India and the Centre for Development of Advanced Computing (C-DAC), Mumbai, Department of Information Technology, MC&IT, Govt. of India.