8.2 Localization of Linux Operating System

Localization of Linux Open Source Software for Hindi

Introduction

Many people in India are currently excluded from computer use, the Internet, and the World Wide Web by absence of software in the language, which majority of Indians speak. Availability of local language software will play a crucial role in the process of taking the benefits of Information Revolution to the local community. In this way we can also prevent the restriction of resource usage. The "Technology Development for Indian Languages (TDIL) Programme" of Department of Information Technology, Ministry of Communications & IT has been working to provide Indic Languages Computing.

Developing a local language interface at an operating system level is considered better than developing it at an application level as the former enables us all the applications running on the top of the operating system to inherit the interface.

In view of the above a project titled "Localization of Linux Operating System" was initiated at CDAC, Mumbai (formerly NCST, Mumbai) under the TDIL Programme with the following objectives:

1. To enable Hindi at system level using Unicode encoding and Open Type font technology in X Window system -a dominant GUI on Linux.
2. To enable Hindi in various existing applications which are distributed in standard Linux distributions.
3. To create Hindi locale in X Window system with Unicode support.
4. To enable Hindi in standard utilities like editors and browsers.
5. To develop an Indic Script Shaping library to enable script shaping for Hindi.

The main objective of this project is to design a localized "user friendly" interface at the system level, which look more natural to the local user and also to localize suitable components within Linux OS (like GNOME desktop environment) to enable applications to create, edit and display contents in Indian Languages.

Characteristics of Indic Scripts

The Indic scripts are very complex in nature however they have a common phonetic nature. The various characteristics of Indic scripts and complexities involved in providing Indian Language support in computing are described below.

Display & Editing

Display and editing of text in linearly composed scripts like the Roman script is relatively simple to handle. But Indian language encoding and display requires computational processes hence enabling Indian language can not be achieved by simple font level substitution. Sophisticated processing of the encoded text is needed for determining the final appearance on the display.

Character Input

Indian language text input differs from that in English. The most significant difference of these is that in English, each keystroke maps directly to a letter. Each letter has a unique code. A *"Syllable"* - the Indian language equivalent unit of writing letter, however is composed of one or more characters i.e. vowels, consonants and modifiers, entered through the keyboards or any other input mechanism. The user types in a sequence of vowels, consonants, and modifiers. The machine then composes syllables at run time, based on language dependent rules. Every syllable is thus represented in the machine as a unique sequence of vowels, consonants and modifiers.

Glyph Display

In Indic scripts each syllable has a unique visual representation. As it is not possible to have glyph for each syllable, the font file normally contains certain component glyphs from which a syllable is composed. The onscreen representation of a syllable is then a composition of glyphs from the Indian language font.

Caret positioning

In a roman editor, carets are positioned in between alphabets. In an Indian language editor, carets are positioned in between syllables. Syllables are a sequence of bytes in memory and a sequence of glyphs on screen. Moving over a syllable means moving over appropriate number of bytes in memory and over combined advanced width of glyphs on the screen.

Syllable Composition

As the user types in a sequence of vowels, consonants and modifiers, syllables start forming. These syllables are formed progressively. For example a sequence of one consonant and one vowel may constitute a syllable. Adding a modifier to this combination changes it to a different syllable. The states keep changing. A new keystroke that does not form part of the current syllable, marks the beginning of a new syllable.

As the syllables keep changing, their representations on the screen change. The number of bytes in a syllable is not linearly related to the number of glyphs required to represent it on screen. This is radically different from the simple roman script model. What it means is that even when a character is added to the edit buffer the line extent may decrease, or even when a character is deleted from the edit buffer the line extent may increase. The system/application has to be aware of this.

Since the syllable's onscreen representation is a composition of different glyphs, there has to be support for such composition at runtime. Ideally this support must come from the operating system. The complete syllable, in its current state, is required for this composition. This means that the system/application has to maintain the syllable break up information amongst the characters in its buffer.

Thus the editor in an Indian language context needs to maintain a count of the number of bytes for each syllable, and if it has to handle display at individual glyph level then the number of glyphs forming a syllable.

Backspace and Delete

Backspacing removes the last appended consonant, vowel or modifier from the syllable, like a stack's pop operation. When there are no more items to be removed, the syllable is deleted. The syllable and its on-screen representation changes after every backspace. The delete operation however removes the entire syllable.

An editor must provide these required features for enabling Indian language text editing operations.

Architecture of Indic support on Linux OS

X Window system is the dominant graphical user interface of Linux, so this has been modified to Indian languages. Basic changes to the X Window system infrastructure have been done to localize all modern packages by locale setting. The architecture of the system is shown in figure 1.

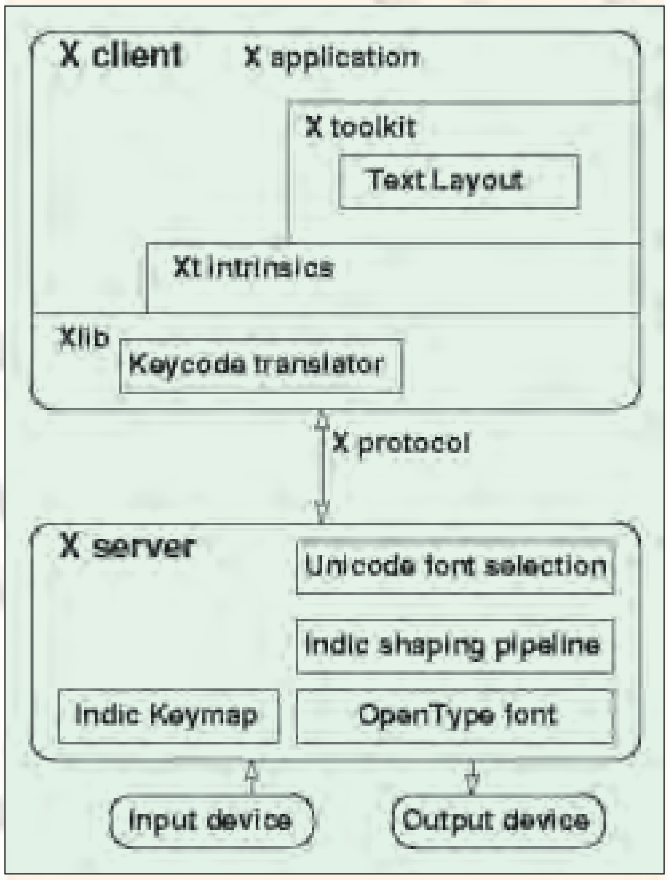


Figure 1- The Architecture

Unicode encoding has been selected to display multilingual text. The X Server has been modified to treat all 8 bit strings as UTF-8 encoding of Unicode characters. The XFree86 4.0.3 supports only one font in its graphic context. It has been modified in such a way so that at startup it is loaded with a selected set of default fonts, for example, Latin and Devanagari. At run time depending upon the Unicode, the server selects the correct font for display. A default system font can be selected for Indian script at X Window startup time.

The fixed fonts (like BDF) are not suitable for Indian scripts. Scalable True Type Fonts solve only scaling. It does not solve the problem of placing the vowel symbols. Because of the phonetic nature of the Indian scripts characters may change their position and appearance as the new characters are typed in. For example, क ् र should be displayed as क्र and not as क् र . Open type Fonts contains tables which give substitution and positioning information that is needed by Indian scripts. Hence Rendering Engine for Open Type font has been incorporated into the XServer. The characters to be displayed will first be classified according to the language. Then it will be split into syllables. For all Indian scripts, the syllable breaking is similar. After this each syllable is reordered. Reordering is script dependent. After reordering, the syllable undergoes substitution with by one or more glyphs. Finally the glyphs are positioned properly. As an example, suppose text sequence क ् र म is typed by user. It is broken into two syllables क ् र and म . The first syllable may be reordered to क र ्. After reordering substitution is done to get क्र . Positioning these two glyphs finally gives क्रम . The block diagram of the shaping pipeline is as shown in Fig. 2.

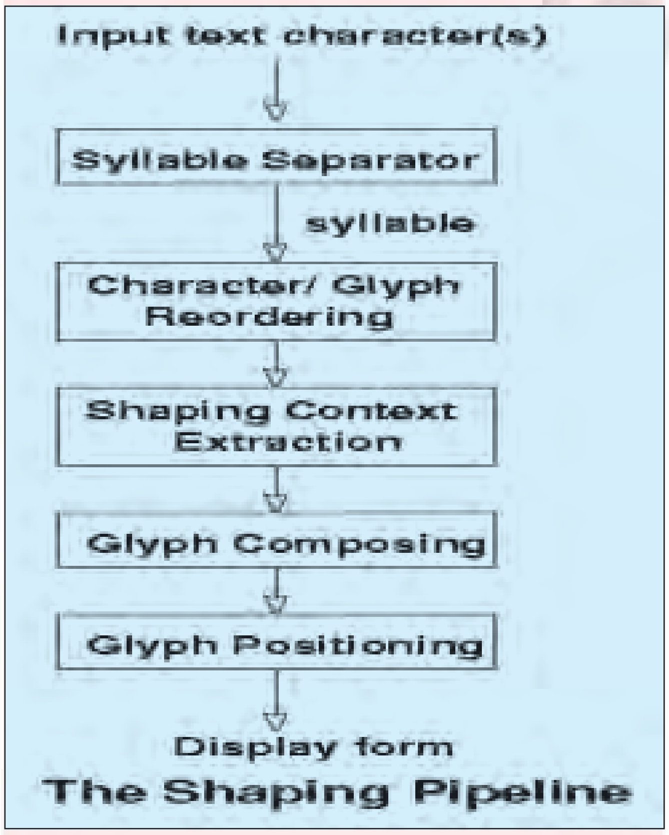
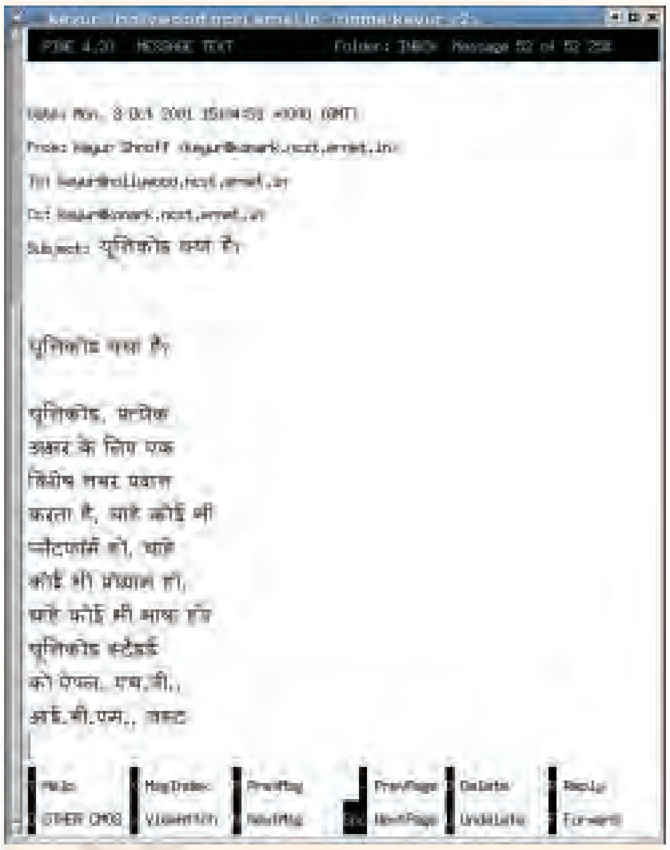
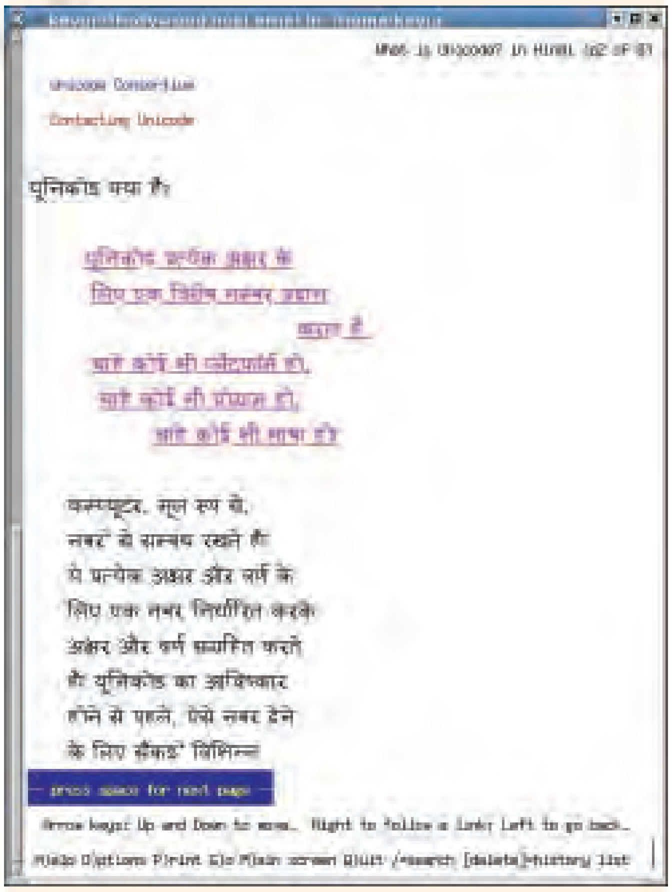


Figure 2- The shaping pipeline



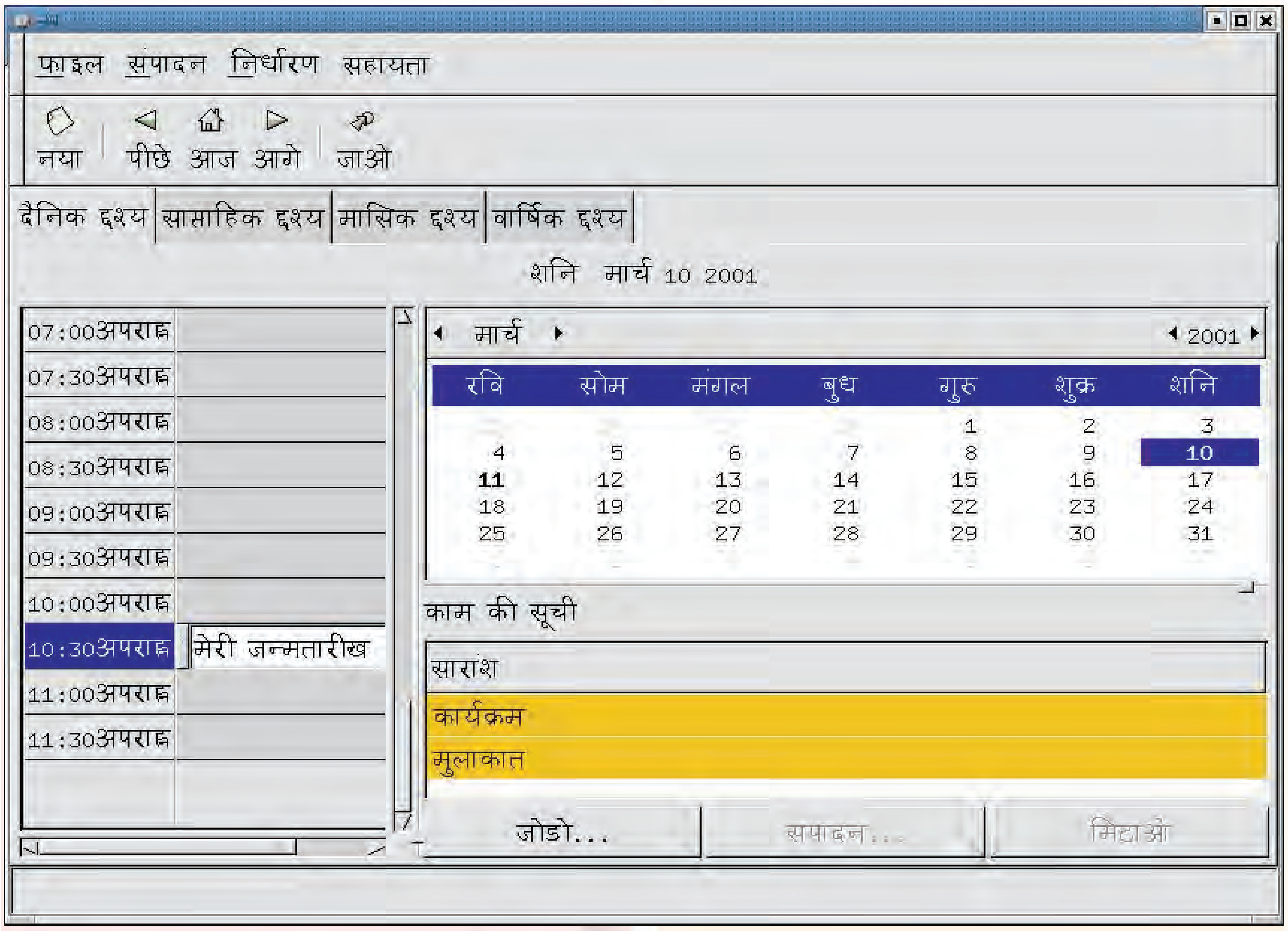
```
kayur@hollywood.ncst.ernet.in: /home/kayur/...
PINE 4.33   MESSAGE TEXT            Folder: INBOX  Message 52 of 52 ALL

Date: Mon, 8 Oct 2001 15:14:52 +0000 (GMT)
From: Kayur Shroff <kayur@konark.ncst.ernet.in>
To: kayur@hollywood.ncst.ernet.in
Cc: kayur@konark.ncst.ernet.in
Subject: यूनिकोड क्या है?


यूनिकोड क्या है?

यूनिकोड, प्रत्येक
अक्षर के लिए एक
विशेष नम्बर प्रदान
करता है, चाहे कोई भी
प्लैटफॉर्म हो, चाहे
कोई भी प्रोग्राम हो,
चाहे कोई भी भाषा हो।
यूनिकोड स्टैंडर्ड
को ऐपल, एच.पी.,
आई.बी.एम., जस्ट

? Help       < MsgIndex    P PrevMsg        - PrevPage   D Delete      R Reply
O OTHER CMDS > ViewAttch   N NextMsg      Spc NextPage   U Undelete    F Forward
```

Figure 3 - Screenshot 1 - pine - a utility for Internet News and Email



```
kayur@hollywood.ncst.ernet.in: /home/kayur
                                        What is Unicode? in Hindi (p2 of 8)
   Unicode Consortium

   Contacting Unicode

यूनिकोड क्या है?

      यूनिकोड प्रत्येक अक्षर के
       लिए एक विशेष नम्बर प्रदान
                        करता है
       चाहे कोई भी प्लैटफॉर्म हो,
        चाहे कोई भी प्रोग्राम हो,
            चाहे कोई भी भाषा हो।

   कम्प्यूटर, मूल रूप से,
   नंबरों से सम्बन्ध रखते हैं।
   ये प्रत्येक अक्षर और वर्ण के
   लिए एक नंबर निर्धारित करके
   अक्षर और वर्ण संग्रहित करते
   हैं। यूनिकोड का आविष्कार
   होने से पहले, ऐसे नंबर देने
   के लिए सैंकड़ों विभिन्न
-- press space for next page --
  Arrow keys: Up and Down to move.  Right to follow a link; Left to go back.
 H)elp O)ptions P)rint G)o M)ain screen Q)uit /=search [delete]=history list
```

Figure 4 - Screenshot 2 - lynx - a text web browser



रवि	सोम	मंगल	बुध	गुरु	शुक्र	शनि
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Figure 5 - Screenshot 3 - gnomecal - a calendar application

Support for Other Indic Scripts

The work has been started to support other Indic Script on Linux i.e. Assamese, Bengali, Gujarati, Kannada, Malayalam, Marathi, Oriya, Punjabi, Sanskrit, Tamil & Telugu. Also the Open Type Font development for these languages initiated. A screenshot of the stress testing of Tamil and Malayalam shaping engine is shown as in Fig. 6.

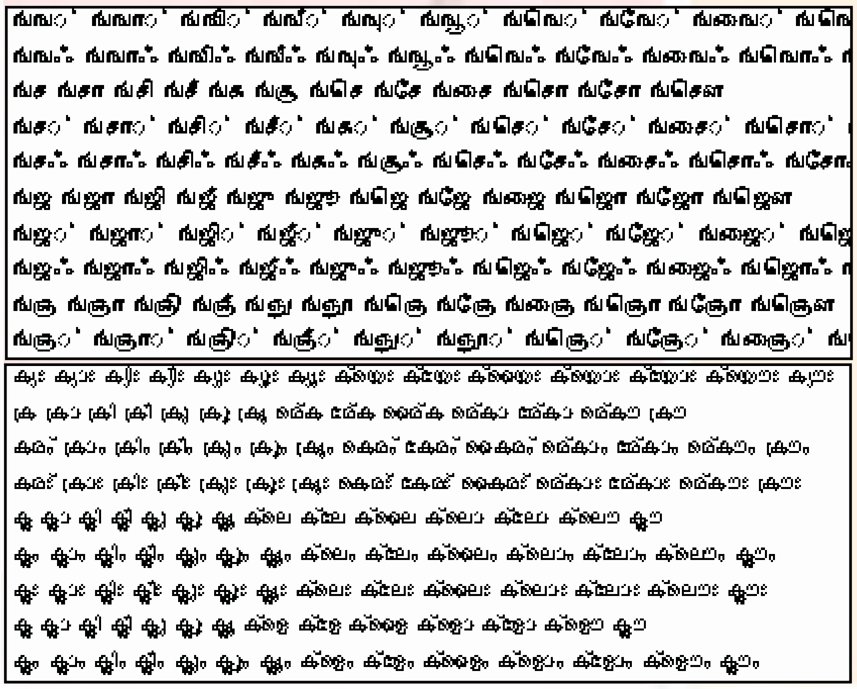


Figure 6 - Screenshot 4 Stress testing of Tamil & Malayalam shaping engine

Courtesy: Dr. Alka Irani
Senior Research Scientist
Shri Vinod Kumar
Software Specialist C-DAC
(Formerly National Centre for Software Technology)
Gulmohar Cross Road No. 9,
Juhu, Mumbai 400 049
Tel. : 91-22-26201606
Email: indix@ncst.ernet.in