Multilingual Computing in Malayalam : Embedding the Original Script of Malayalam in Linux and Development of KDE Applications

Rajeev J S Chitrajakumar R Hussain K H Gangadharan N

Abstract

Indic Language Computing can be fully realized only through embedding vernacular scripts in operating systems. With the advent of OTF (Open Type Font) embedding local scripts in OS compliant with Unicode has become a reality taking computing beyond word processing. Microsoft has already come to this field strongly by embedding Devanagari in MS Windows. Compared to the closedness of Microsoft OS, free and open environment of Linux is ideal for the early accomplishment of multilingual computing. This paper describes initiatives of Rachana team in embedding Malayalam script in GNU/Linux operating system. Modules are added for KDE with its rendering engine QT so that the original exhaustive character set of Malayalam developed by Rachana is embedded fully in compliance with Unicode. For the first time, prospects are open to create DBMS and information systems using Malayalam script. Computing in Malayalam language is being initiated in the true sense only now. The procedures set up by Rachana-GNU/Linux is highly beneficial to the goals of INFLIBNET in fulfilling a total integrated bibliographic control of Indian literature in their native scripts.

Keywords : Multilingual Computing, Localization, Unicode, Desk Top Publishing.

0. Introduction

Language is the foundation of all information systems. Language being the medium of information, there can be no information technology without language. Though IT has successfully assimilated voice and visuals in building up multimedia applications, secondary data indispensable for describing audio-video elements are coded using text. Later, data or information is retrieved and processed using the same text. Words and text are formed using the basic unit of written language called alphabet, character or *lipi*. Lipi in a language is the most systematized and standardized signs used to describe concrete or abstract concepts/ sounds. Without lipi there can be no information systems or information technology.

The computer system to input, render and process text has traditionally been Latin (Roman) based. Support for Indic languages would be implemented using custom rendering engines/shaping engines or using special cases such as Latin font encoding and custom keyboard input systems on top of the Latin based system. This however had several problems – either the custom keyboard input systems wouldn't be applicable to all application programs, or the font encoding would interfere with the correct rendering.

This led to the realization that in order to implement Indic Language solutions it would be necessary to embed the processing code into the Operating System itself, i.e., as first class citizens of the text world just like Latin based languages. Embedding means to allow input, rendering and processing of a language script in the traditional GUI widgets such as Textboxes, Labels and Buttons. Language computing in its truest sense, extending the capability of computing to all spheres of digital application, can only be achieved through this embedding to make the script of the language a 'live' part of the operating system as well as applications.

For the past 15 years word processing and DTP have been smoothly going on in all Indian languages. At the same time none of these languages has achieved a perfect DBMS in local script. We should admit the truth that information technology in India has not yet accomplished information system development in any Indian language! By embedding Indian languages in OS our languages will become as natural as English to the computer and we can make use of our scripts in all the conceivable fields of digital applications. Application programs could utilize operating system facilities for input, rendering and processing of the text and developers need only to provide the text in a suitable form known as encoding. Embedding would also allow more complex programs such as spreadsheets and database management systems to provide support for these scripts, in a uniform manner.

The work done by the authors in embedding Malayalam language falls into following categories:

? Fixing the character set of Malayalam

? Designing fonts

? Choosing an Operating System and GUI

? Coding for Embedding the script

? Adapting applications like text editors, word processors, spread sheets, Graphic utilities, DBMS and DTP to the embedded system.

Accordingly, the paper discusses the following topics:

? Malayalam Lipi and Rachana Language Campaign (Fixing the character set)

? Unicode and Open Type Font (Specifying the character rendering according to an international standard and developing Malayalam OTF fonts)

? Development of Rachana-GNU/Linux Distribution (KDE, OpenOffice, Scribus, etc.)

1. Malayalam Lipi and Rachana Language Campaign

It is from Tamil that Malayalam was born. Tamil is the most important among Dravidian languages. However, it is from the traditions of Sanskrit, the Indo-Aryan language, that Malayalam draws its rich diversity of words and compound alphabets (conjuncts).

It was in 1821 that Benjamin Bailey, a Jesuit priest, designed the first Malayalam metal types for the printing machine. From the basic 56 characters, he forged around 600 conjuncts in beautiful metal type. These letters adopted by Benjamin Bailey were in use for hundreds of years in Malayalam script. Later Herman Gundert designed and added several more conjuncts, and the Malayalam language came to possess 1000+ unique and rich type characters. These two pioneers were also authorities on comparative linguistics of Indian languages, thereby the design of Malayalam characters and types naturally encompassed pan Indian and local specificities. The people of Kerala recognize their language and have become the most literate of communities by learning and using this script. That this character set developed by them have survived and spread extensively during the past one and a half centuries shows their wide acceptance and faithfulness to the original script.

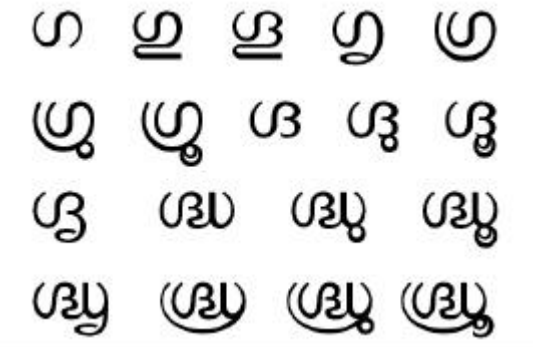
During early 1970s this sophisticated and systematized script language suffered a serious setback. This was the time typewriters started appearing on office tables. The demand for adopting Malayalam as the official language also became strong during this time. Considering the need for typing office files and

correspondence, the nearly 900 characters of Malayalam language was reduced to just 90 to fit into the keyboard of a typewriter. Even some of the fundamental vowel signs were excised. The most aesthetic and functionally superior Malayalam script was trashed without any logic or sensitivity to history. The stable structure attained by Malayalam script suffered cracks and several incongruities developed even in semantic level. This fatal programme was led by a government agency, the Kerala Language Institute and they even succeeded in implementing the truncated alphabets for producing the textbooks of primary standards in 1973.

When computerized typesetting (DTP) became popular in 1980s several software packages and fonts emerged. Several font designers, working in institutions outside Kerala and ignorant of Malayalam language, designed conjuncts casually generating contradictory character mapping which is not found in any other Indian languages. Integrated and stable character set of Malayalam language that survived for centuries became disarrayed and incoherent, and this non-systemization raised the greatest hurdle to attempt areas of digital computing other than word processing.

It was in response to this non-systematization of Malayalam that a language campaign under the banner 'Rachana' (which means 'Graceful Writing') was launched with the following objectives.

- The unique character set developed by a people over centuries transcending class divisions is not just a geometrical sign but the symbol of a culture.
- A language should be revised and modernized when deficiencies are observed in use and communication. And not based on the limitations of a transient historical phenomenon of a typewriter machine.
- The return to the original script is the only way to surmount the disintegration of Malayalam language in learning, comprehension, writing and printing.
- Modern information technology has made it possible to include and manage the exhaustive character set of Malayalam in any application. Rather than cut the alphabets to fit a machine, technology should be tamed to serve the language.
- The original Malayalam alphabets should be made ready for use in the modern language technology. The current information technology is advanced enough to embed the original exhaustive character set of Malayalam in all fields of digital computing.



Conjuncts formed by GA, DHA, DHHA, REPHAM and Consonant-Vowels, showing the exhaustiveness of Rachana character set

With the declaration of Rachana font comprising the exhaustive character set under GNU-GPL (General Public License) in February 2004, the efforts to embed the original Malayalam script in GNU/Linux platform has started.

2. Unicode

The Unicode is a universal encoding format designed to represent the symbols and script elements of the world in a uniform manner. The Unicode is a minimalistic encoding which includes currently all major scripts in use. The basic principle “Encode the characters, not the glyphs” denotes the minimalism of the Unicode encoding. By encoding only abstract characters to code points, the encoding would be able to reflect the semantics of the script rather than represent a mere number. This simplifies higher level processing such as EASCII to Unicode conversions and text stream to visual rendering.

In short the advantages of Unicode are listed below:

? It is a minimalistic encoding designed to represent all other encodings.

? Along with the OTF (Open Type Font) it allows development of languages with complex visual rendering requirements.

? It allows easy migration from an existing encoding scheme to the Unicode.

? The determination of script/code page can be done automatically in the Unicode, since each script is allocated a unique code block.

2.1 Emergence of OTF (Open Type font)

Fonts are the means by which characters in a language can be rendered visually on the screen or in print. It is one of the basic subsystems of text processing in the computer. Initially fonts were bitmap fonts. Soon, for the purposes of digital typography, fonts were designed with Bezier curves, which allowed arbitrary scaling of the font without loss in quality. The abstract curve representation of a character is also known as glyph.

For new languages that entered the computing arena, like Indian languages, the availability of only 256 slots in ASCII based systems made several constraints in the number of glyphs that could be designed in any given font. Combinations of basic characters known as ligatures or conjuncts could be designed and used by allocating a code-point to it. But the space available would remain as low as 256. This forces incomplete and disintegrated implementation of various languages (or families) like Indic, which need a lot more than 256 code-points to represent the entire repertoire. This is what happened in the case of Malayalam language when the attempts were made to accommodate its 1000+ original/ traditional characters.

OpenType Font (OTF) is the new technology with a variety of features that allow complete implementation of Indic languages satisfying all their peculiar characteristics. Microsoft and Adobe introduced it jointly in 1997 to meet the requirements of complex scripts and multi-lingual documents, as well as new techniques in rendering. Although OTF can be used with a variety of encoding, it is best implemented with the Unicode.

For each Unicode encoded character, the font designer can design glyph shapes for that character. Total number of shapes in the encoded and unencoded slots may come around 65,000 (i.e. 2^{16}). The unencoded set contains glyphs for combinations of encoded characters. In this way, an Indic text that contains mostly conjuncts can easily be represented and accordingly a font can be designed accommodating any number of glyphs.

An OTF can only be used advantageously in conjunction with a shaping engine (rendering engine) that is usually implemented in the Operating System (OS) or the windowing toolkit. The shaping engine provides text layout services, which transform a piece of text into glyphs from both the encoded and unencoded sets. This achieves the complex shaping requirements for conjuncts in Indic languages with the use of basic abstract characters of the Unicode.

This process of Shaping/ Rendering occurs as follows:

- The text is analyzed and broken down into segments. Segmentation is done at cluster boundaries.
- Then the string is mapped to a set of glyphs representing the basic characters.
- The basic glyphs are then further transformed on the basis of OTF features. OTF features are special tags associated with the unencoded-glyph shapes, which establishes correspondence between complex shapes and their basic components. The application of features can happen either by substituting the complex shape with a sequence of basic shapes, or by positioning the shapes over the existing shape.
- The final sequence of glyph shapes is laid out according to the requirements of the software.

For speed, interoperability and simplicity of laying out fonts, the system applies several optimization techniques. In general, the position of the OTF layout process is in the OS. For highly specialized applications that require fine use of text visuals, it may also be re-implemented for the particular system. Needless to say, the process should be standard on all systems so that there should not have different input methods and different visuals for the same text.

2.2 OTF in Malayalam

Indic language computing will be greatly benefited from the development of OTF technology. That the limitation of 256 slots is left behind by the 65000 code points alone makes OTF receptive to conjunct-rich Indic scripts.

An example of the application of OTF in Malayalam script will explain conjunct formation and its Unicode representation clearly.

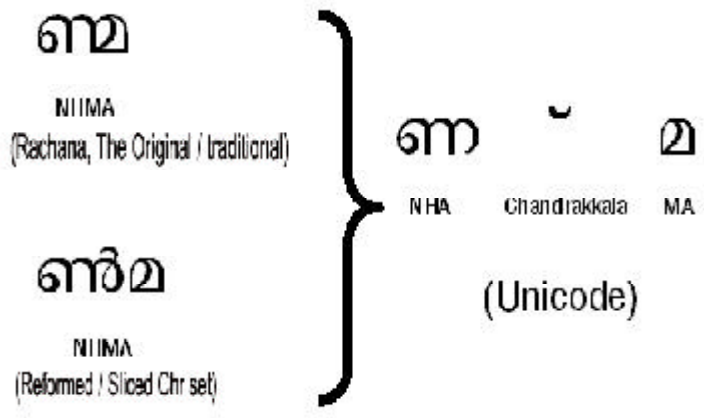
Consider the *NA Chillaksharam* which is represented by a single basic character in the Unicode block. As per the philosophy of the Unicode, glyphs that may be represented by sequences of basic characters will not be allocated a specific code point. So, the *NA Chillaksharam* is not given such a code point. In order to allow rendering of the chill form, the OTF provides a “Halant” feature. According to the Unicode standards, the *Na Chillaksharam* form is represented in the encoding by the sequence

NA + Chandrakkala + ZWJ

All the three characters that constitute this sequence are basic characters in the Unicode. The Halant feature is tagged on the *NA Chillaksharam* glyph that is placed in an unencoded slot and this is applied to *NA + Chandrakkala + ZWJ* glyph sequence. Such a sequence is indeed transformed into the *NA Chillaksharam* form when rendered on screen.

2.3 Unicode and legacy encoding

Considering the different character sets of Malayalam (Original and Reformed), an understanding of the Unicode encoding model can be derived easily from the following example:



In Rachana encoding, there exists a single code-point for the representation of *NHMA*, where *NHMA* is

NHA + Chandrakala + MA

In other encoding for reformed Malayalam such as ISM-Gist and SreeLipi, there does not exist a code-point for this character. Instead the character is rendered as the combination of two characters

(NHA Chillaksharam) (MA)

Unicode solves this problem by encoding only the basic minimal characters from which the compound characters are generated. *NHA, Chandrakalla* and *MA* have individual code-points and the complex shape *NHMA* is encoded as the basic code-point sequence such as

NHA (followed by) *Chandrakkala* (followed by) *MA*

Rachana as well as other encoding can easily be encoded using the Unicode. Along with appropriate OTF fonts and layout tables, it is possible to provide the exact rendering for both encoding schemes of Traditional and Reformed. OTF can reuse glyphs developed for older systems and allow migration to the Unicode system. In time, all documents using either encoding can be converted and the users can be fully migrated from legacy encoding to the Unicode.

3. Development of Rachana-GNU/Linux Distribution

Free operating systems like GNU/Linux and FreeBSD provide source codes and allow modification by independent developers. Due to this, it is possible to implement very good embedding solutions. GNU/Linux provides a plethora of GUI systems and libraries all of which can be modified to provide script embedding. In general, the two most common GUI platforms on GNU/Linux are KDE and GNOME. Embedding the script solution in, say, KDE would allow all KDE based applications to reuse the rendering system. Some of the well-known applications are distributed in either platform. Also, some systems like OpenOffice.org (a replacement for MS Office) have created their own infrastructure for rendering.

3.1 GNU/Linux

GNU/Linux is a total operating system available for a variety of hardware platforms providing a complete spectrum of functionality for users. GNU/Linux runs on platforms ranging from space satellites to home desktops and handheld devices.

A highly simplified view of the architecture of a GNU/Linux computer is shown below:

<table>
<tr><td colspan="2">Environments</td><td>Servers</td></tr>
<tr><td>Command Prompt
Shell (provides command prompt)</td><td>GUI
Desktop Environment (KDE, GNOME, etc)
Graphics and Windowing Toolkits (Qt, GTK, etc)
X (provides basic graphics)</td><td>Various Servers such as RDBMS, Web server, email server, etc</td></tr>
<tr><td colspan="3">System Libraries (glibc and lots more) + system utilities</td></tr>
<tr><td colspan="3">File system</td></tr>
<tr><td colspan="2">Non-hardware specific Kernel parts</td><td>Hardware-specific kernel parts
Computer hardware and peripherals</td></tr>
</table>

Layers of GNU/Linux

The figure shows the layered concept of OS design, where each layer uses the facilities available in the lower layer to provide facilities to the higher layers.

At the very bottom lies the computer hardware and peripherals such as CPU, RAM, CDROM drives, hard disk drives, printers, mouse and display systems, along with the kernel. The popular kernels include Linux and Hurd. It is important to note that Linux is just a kernel and it is useless to an end-user without the rest of the GNU system, which lies in the higher layers.

Above this layer lies the file system that provides facilities for handling files and directories. The Linux files system can be of several types, each with its own features. Some file systems provide an automatic backup capability called journal, which can prevent all kinds of data loss (except for disk drive failure).

System libraries are the functional units, which provide a variety of services (access to the file system, networking capability, password authentication, etc). System utilities are common programs used for listing directories, performing common system tasks, etc.

Above this layer is where the actual end users do their works. There are two environments: the command line environment (Shell or CLI) allows user to type commands and see results. GUI (Graphical User Interface) is the commonly used interface by users, where there are windows, buttons (widgets), the start menu, graphical clock and other applications like office suites and web browser. It is in the GUI that Indic language embedding is most useful to the end user.

Servers are special programs that continually run in the background and provide specific services such as the web server (useful for making websites) and the email server (for storing and transporting email). Developers make applications using servers and end users make use of GUI tools to access these applications.

GUI is split into three layers: the basic X system allows drawing of lines, circles, etc. and takes care of various hardware events. For example, when the mouse is clicked the X system transfers this event to the application along with the position where the mouse is clicked. X is also in charge of display and input/output hardware when running GUI applications.

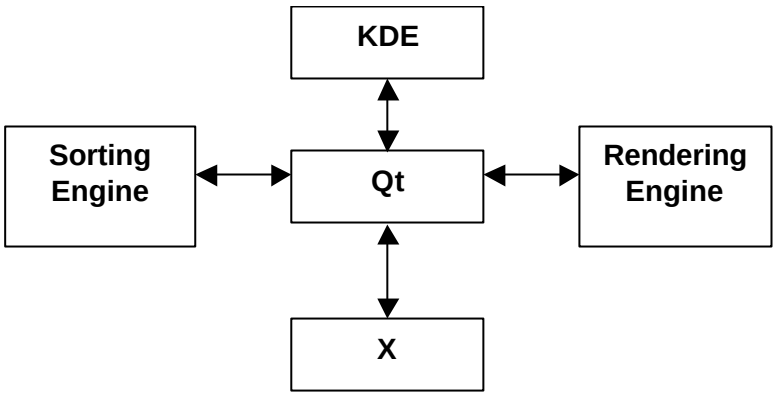
Above the X system lies the graphic toolkits: they provide an easy access to build GUI applications. Unlike the X system, they are more fully featured for developers and provide a unique look and feel to the applications. The graphics and windowing toolkits utilise the services of X to perform its work.

At the top lies the Desktop environment which provides the common concepts such as file manager, desktop, task bar, graphical clock, start menu, common interface for configuration and easy access to the various applications on the system. The Desktop unifies the applications and provides an area with common facilities that can be used by the applications to provide a unique experience to the user. There are several Desktops available on Linux, most popular being KDE and GNOME.

3.1 Development of KDE Applications

Authors of this paper had decided to develop Malayalam embedded system on the KDE platform. It was chosen not just because of the high quality of the existing system, but future development of domain-specific systems such as Library Management systems would be much easier using KDE and its underlying Qt GUI library.

Authors also decided to adopt some non-KDE applications for implementing Malayalam such as OpenOffice.org and Scribus (a professional DTP tool). Although Scribus is a Qt application, it is considered separately, because it has its own text rendering infrastructure due to its need for highly accurate placement and rendering of text to achieve tight typographical control.



The above diagram shows the layered relationship between various parts of the GNU/Linux GUI system. At the bottom is the X system, which takes care of handling the monitor screen and hardware for display. In the middle, Qt provides general services for drawing widgets like Textboxes, Labels and Buttons. At the top is KDE, which provides the common desktop.

Qt is the rendering engine for performing the task of taking an encoded text string and rendering it on the screen. Obviously, Qt is the place for inserting solutions for embedding. The authors proceeded to analyze the requirements of the script rendering engine, along with the script properties, font requirements and encoding formalism. Based on this analysis, the rendering engine component was completed.

Some of the applications that were tested with the new rendering engine are given below:

? Konqueror : Web browser, which allows reading of Malayalam websites

? Kmail : email client like Outlook Express and Eudora

? Kword : Word processor (part of the Koffice suite)

? Kspread: Spreadsheet program (part of the Koffice suite)

? Kpresenter : Presentation software (part of the Koffice suite)

? Kchart: Statistical chart program (part of the Koffice suite)

? Kformula: Formula editing program (part of the Koffice suite)

? Kivio: for flowcharts and other charts (part of the Koffice suite)

? Kolourpaint: a simple yet sophisticated paint program

? Kedit: KDE text editor like Note Pad

? Kgpg: Message encryption program

? Kopete: Instant messenger (with Yahoo, MSN and Jabber protocols)

? Konversation: an IRC client

? Korganizer: Calendaring, scheduling and journal software

? KAB: Address book for storing contact information

? Knoda: Database development system

3.1.1 Knoda

Knoda is particularly useful to build database applications. In the simple usage, it can be used to make forms and connect the forms to databases for data entry and retrieval. In the advanced usage mode, one can write full programs in Python and connect them with the forms. Python is a fully featured language, easy to learn as well as having a very powerful and fast interpreter. As the backend database Knoda may use PostgreSQL server or MySQL server as well as the embedded database SQLite. SQLite allows the development of read-only and embedded databases for CDROM publishing, or for standalone software without the need for servers.

3.2 Office Suites and DTP

The independent software such as OpenOffice.org and Scribus also follow a similar approach to rendering. OpenOffice.org is a full office suite very similar to MS office suite. Scribus is a DTP package in Linux similar to QuarkExpress. KDE architecture can be reused for the implementation of a script under these systems as well. In Scribus such an infrastructure did not exist at the time of the development, and it was up to the Rachana team to provide this system, thus allowing all languages (including non-Indic scripts) to be implemented in the acclaimed DTP tool. When compared to OpenOffice, Koffice (the native office suite of KDE) requires lesser computer resources such as memory and CPU power. Combined with the ease of use and power of the Qt toolkit, Koffice is a much better candidate for further developments.

3.3 Future Directions

Future direction of the Rachana-GNU/Linux distribution with respect to the Library Management community can take place in several simultaneous but interconnected directions. Various tasks to be completed do not allow a linear view, yet we present it as a list.

OS :

- Extension of embedding to GNOME/Pango
- Research on innovative methods for text entry specific to the complexities of Indic languages
- Extending search mechanisms of KDE and GNOME to the Digital Knowledge Archive

Domain-specific Software :

- Support tools for Indicnet
- Content Management system customized for libraries (including support for Multimedia)
- Development of OPAC, either by extending existing ones or developing new software taking into account the diversity of Indic scripts and local requirements
- Storage of data on a Grid
- Support for intelligent queries on structured data in the Digital Knowledge Archives (e.g., structured query on patents and legal codices)
- Development of computer aided teaching learning system for teacher-student community, especially in K-12 environments

Knowledge Initiatives :

? Development of Indicnet which is a Wordnet like knowledge-base of Indic words, meanings, synonyms, antonyms, homonyms etc

? Phased implementation of a national-level Digital Knowledge Archives available to all over the Internet/national network

? Construction of metadata repository

? Construction of Grid infrastructure

3.4 Bibliographic control of Indian Literature and INFLIBNET

When using a comprehensive library automation package, the present practice is transliterating local bibliographic data in to English, which is very often inconsistent and misleading. Implementation of localized KDE/Qt allows the development of local language bibliographic systems that are highly integrated with the existing infrastructure. Library automation and creation of bibliographic databases at the national level can be successfully carried out only if DBMS supports all the Indian languages. Rachana-GNU/ Linux is a step towards this direction.

Some of the avenues for future developments in the Library Management systems on GNU/Linux, as envisaged by INFLIBNET, could include the long wished-for national library integration system. If the SOUL package of the INFLIBNET is enabled to accept and process data in all Indian languages it will open up immense possibilities. This will facilitate inputting data and searching information using native scripts. Such software will be ideal for creating online as well as CD-based Indian National Bibliography.

Other schemes may include development of a national digital library resource with multimedia, which would allow final integration of knowledge resources from all the languages all over the country through a single window. The authors hope that the proposals in the future directions (section 4.3) will be carried out as part of the INFLIBNET programmes to realize the ultimate goal of total bibliographic control of Indian literature.

4. Conclusion

Microsoft has already come to the field strongly by embedding Devanagari in MS Windows in 2004. They have declared their plans to do similar embedding for all other Indian languages. Since the scripts and syntax of each Indian language has got its own peculiarities and complexities Microsoft's task will not be as easy as they expect. On the other hand linguists, typographers and IT experts can be easily assembled in every Indian state and can be effectively mobilized for embedding their scripts in GNU/Linux that offer open source code supported by an international fraternity of developers. Compared to the closedness of Microsoft OS, free environment of Linux is ideal for the early accomplishment of multilingual computing. This is in addition to the excellent facilities already well established in GNU/Linux such as its exceptional networkability, security and robust operations.

5. Acknowledgements

We would like to thank Mr Joseph Sebastian (BTC Engineering, Kuwait) and Dr. Varghese Paul (Cochin University of Science and Technology, Kerala). Sincere gratitude to Dr. Mammen Chundamannil (Kerala Forest Reseach Institute) and Mr. K. Raveendran Asari (former Librarian, Mahatma Gandhi University, Kottayam) in preparing and editing the paper.

6. References

1. K. Desktop Environment : htt://www.kde.org, http://www.koffice.org
2. Qt Toolket : http://www.trolltech.com
3. Unicode Consortium : http://www.unicode.org
4. OpenType Specification 1.4 : http://www.microsoft.com/typography/

About Authors

Mr. Rajeev J. Sebastian is completed his B.Tech from CUSAT.
Email : rajeev_jsv@yahoo.com

Mr. Hussain K H is working as a Documentation Officer in Kerala Forest Research Institute, Peechi, Thrissur, Kerala
Email : hussain@kfri.org

Mr. M. Chitraja Kumar is working in a Kerala University. Thiruvananthapuram, Kerala
Email : chitrajakumar_rachana@yahoo.com