HINDI EDITOR
WITH SPELL CHECKER

A DISSERTATION SUBMITTED TO
VINAYAKA MISSION UNIVERSITY,SALEM

By

RITU AGGRAWAL
Registration No:

in partial fulfillment for the award of the degree of

MASTER OF PHILOSOPHY IN COMPUTER SCIENCE

VINAYAKA MISSION UNIVERSITY
SALEM -636305

SEPTEMBER, 2007

December 17,2007

UNDERTAKING

Certified that this Dissertation titled " **HINDI EDITOR WITH SPELL CHECKER** " is my bonafide research work carried out under the supervision of **Dr. Grish Nath Jha**. Certified further that the work reported herein does not form part of any other dissertation or dissertation on the basis of which a degree or award is conferred on an earlier occasion .

Ritu Aggrawal

VINAYAKA MISSIONS UNIVERSITY, SALEM

BONAFIDE CERTIFICATE

Certified that this Dissertation titled " **HINDI EDITOR WITH SPELL CHECKER** " is the bonafide work of **Ritu Aggrawal** who carried out the research under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other dissertation or dissertation on the basis of which a degree or award was conferred on an earlier occasion of this or any other candidate.

Ritu Aggrawal **Dr. Girish Nath Jha**

[SUPERVISOR]

ACKNOWLEDGEMENT

I acknowledge with a deep sense of gratitude, continuous help and encouragement from my supervisor Dr. Grish Nath Jha, which he has provided to me despite his busy schedule. Without his able direction, active supervision, positive attitude and motivation, it would have been extremely difficult for the present research and development to complete. He helped me from time to time in programming and dissertation writing and solved several difficulties.

I extend my gratitude to the Special centre for Sanskrit studies (SCSS), Jawaharlal Nehru University, New Delhi for allowing to use the Language Technology research facilities for this work. .

I express thanks and gratitude for the registrar Dr. Manju Singh and other colleague faculty member Mr. Sumit Chauhan of the Management Education & Research Institute , New Delhi for helping me from time to time during this work.

I convey my earnest credit to Mummy , Papa , Husband, and younger sisters whose positive views, love and extra care have made easy to complete this work.. Lastly, I express my sincere thanks to all those who directly or indirectly helped me in completing this work.

RITU AGGRAWAL

"The language is not just a means of communication. Instead, it is the only means that can save the culture of its people."

LIST OF ABBREVIATIONS

IT – Information Technology

NLP – Natural Language Processing

MT – Machine Translation

MAT – Machine Assisted Translation

VB- Visual Basic

ASP – Active Server Pages

JSP – Java Server Pages

CD – Compact Disk

NOL – National Official Language

DEVANAGRI CODE CHART (In Hexadecimal)

09	00	10	20	30	40	50	60	70
0		ऐ	ठ	र	ी	ॐ	ॠ	०
1	ँ	ऑ	ड	ऱ	ु	॑	ॡ	Res
2	ं	ऒ	ढ	ल	ू	॒	ॢ	Res
3	ः	ओ	ण	ळ	ृ	॓	ॣ	Res
4	ऄ	औ	त	ऴ	ॄ	॔	।	Res
5	अ	क	थ	व	ॅ	Res	॥	Res
6	आ	ख	द	श	ॆ	Res	०	Res
7	इ	ग	ध	ष	े	Res	१	Res
8	ई	घ	न	स	ै	क़	२	Res
9	उ	ङ	ऩ	ह	ॉ	ख़	३	Res
A	ऊ	च	प	Res	ॊ	ग़	४	Res
B	ऋ	छ	फ	Res	ो	ज़	५	Res
C	ऌ	ज	ब	़	ौ	ड़	६	Res
D	ऍ	झ	भ	ऽ	्	ढ़	७	ॽ
E	ऎ	ञ	म	ा	Res	फ़	८	Res
F	ए	ट	य	ि	Res	य़	९	Res

TRANSLITERATION KEYS

अ	a
आ	A
इ	i
ई	I
उ	u
ऊ	uu/U
ऋ	RRi
ऌ	LLi
ए	e
ऐ	ai
ओ	o
औ	au
क्	k
ख्	kh
ग्	g

ध्	dh
ङ	~N
च्	c
छ्	ch
ज्	j
झ्	jh
ञ	~n
ट्	T
ठ्	Th
ड्	D
ढ्	Dh
ण्	N
त्	t
थ्	th

द्	d
ध्	dh
न्	n
त्	t
प्	p
फ्	ph
ब्	b
भ्	bh
म	m
य्	y
र्	r
ल्	l
व्	v
श्	sh

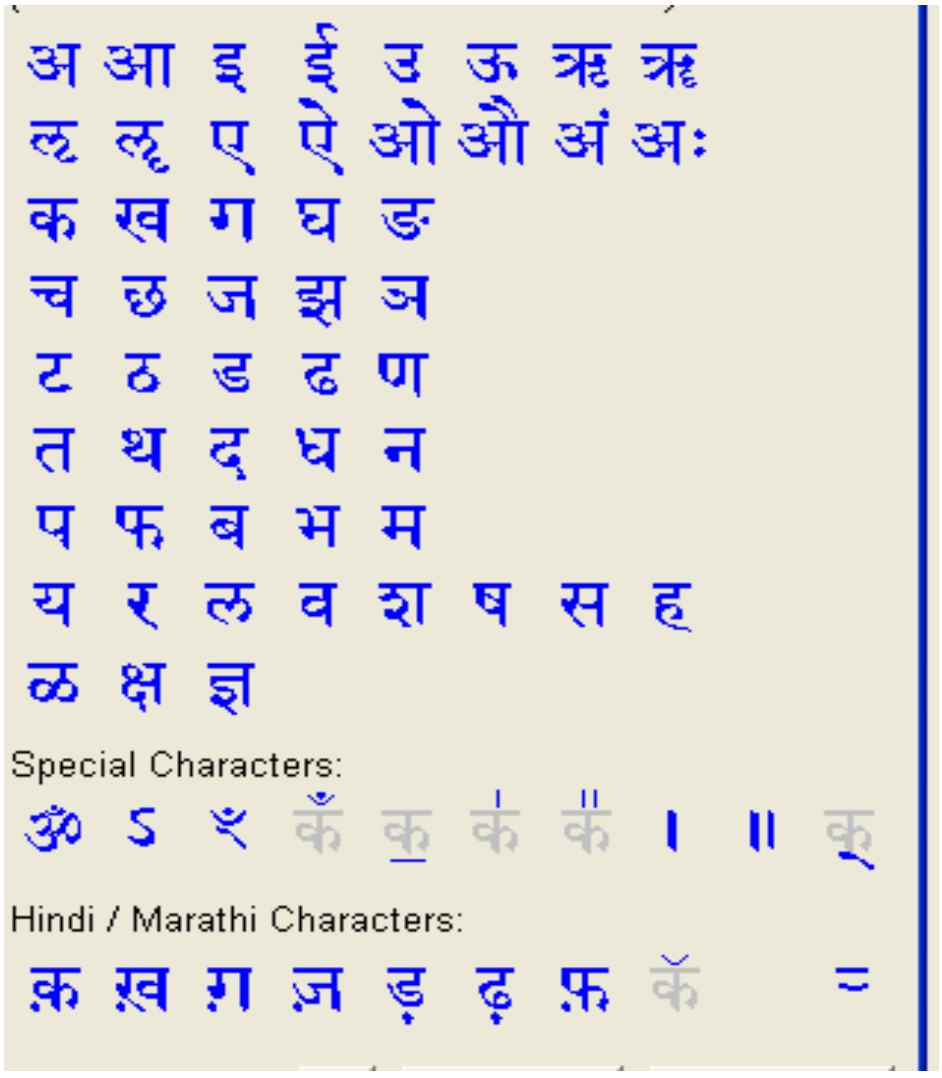
ष्	S
स्	s
ह्	h
ॐ	OM
ॠ	RRI
ऌ	LLI
०	0
१	1
२	2
३	3
४	4
५	5
६	6
७	7
८	8
९	9

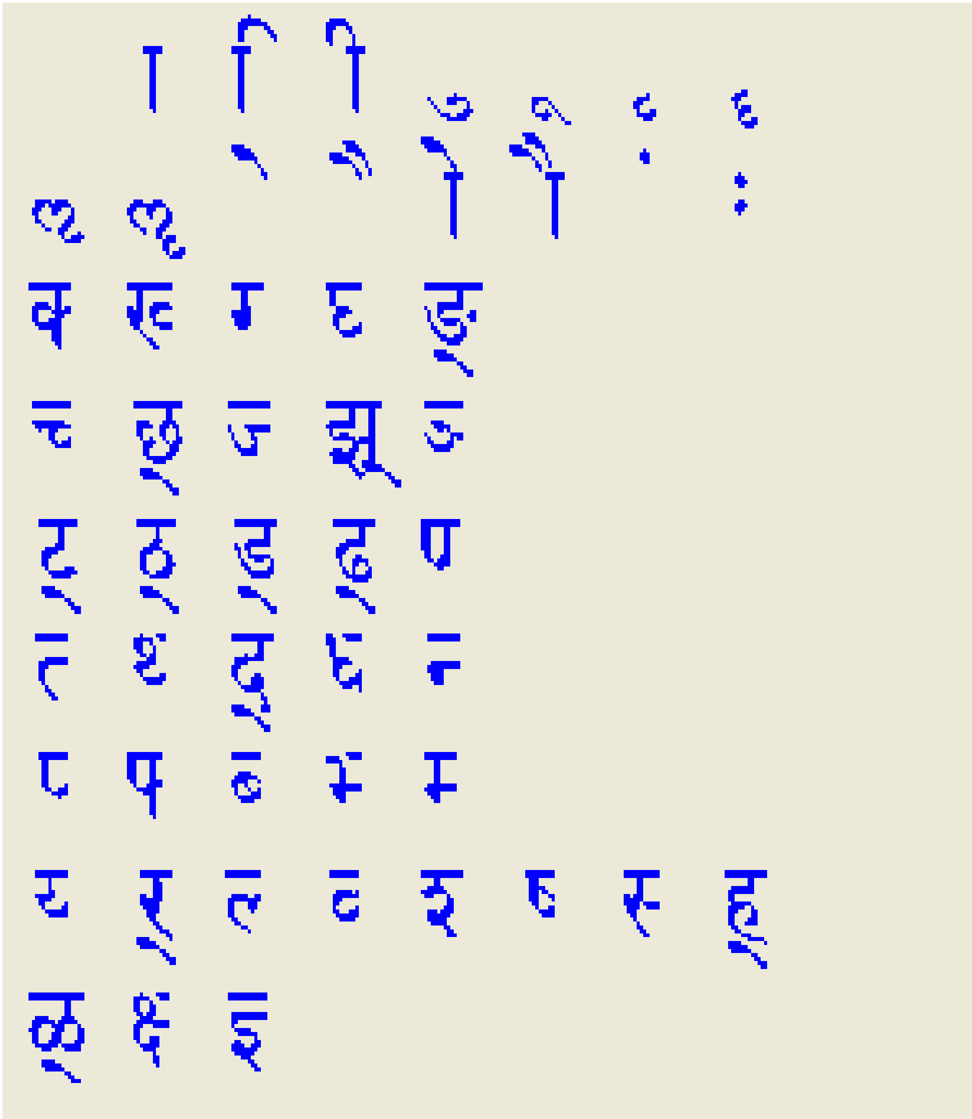
TABLE OF CONTENTS

Chapter 1 INTRODUCTION

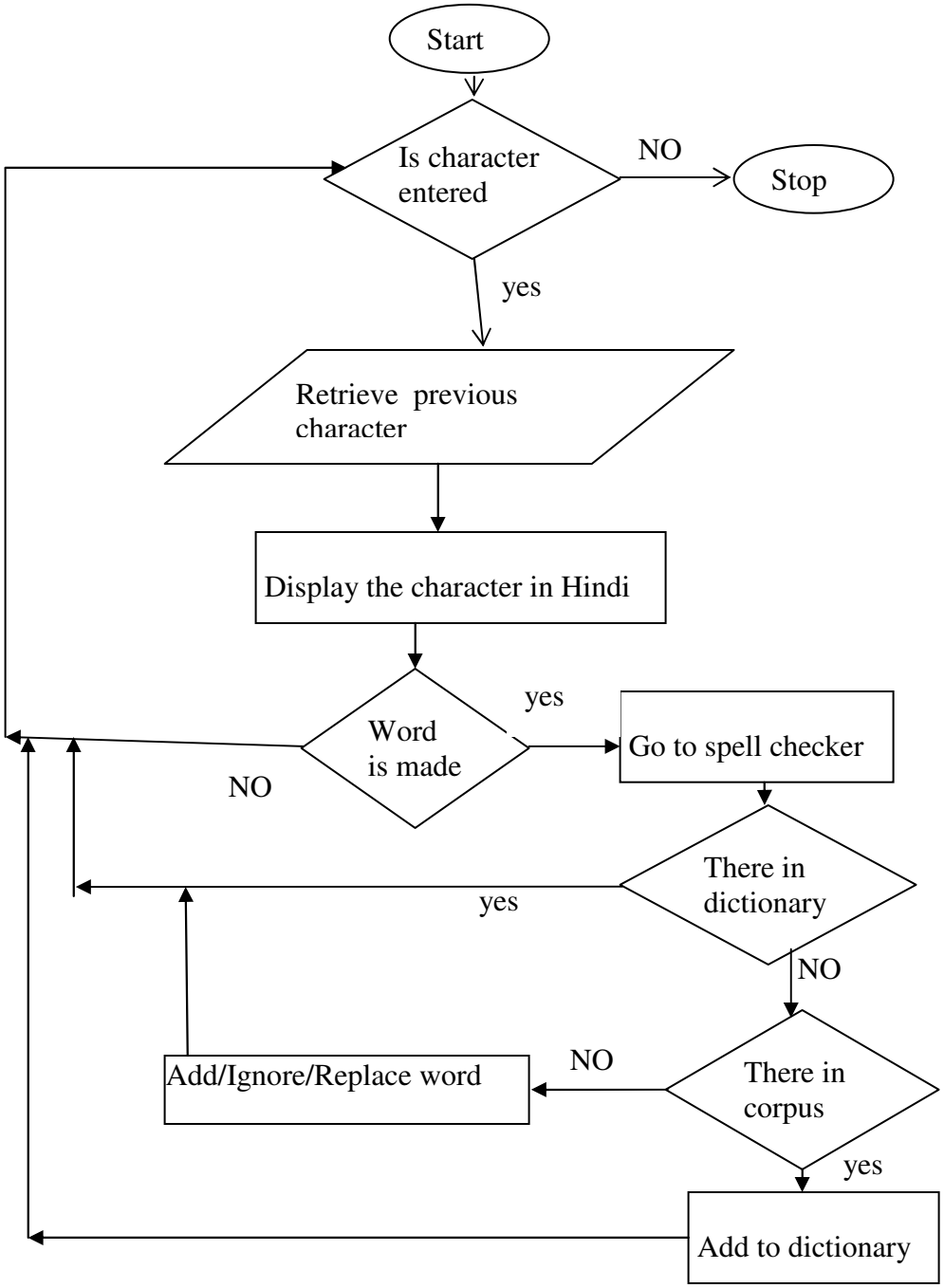
The dissertation is an effort at the M.Phil. level for developing a Hindi editor with spell checker with ITRANS as input mechanism

Itrans is Freeware and has been developed by Omkarananda Ashram Himalayas , Rishikesh , India 2003. It works for windows 95/98/ME/NT 4/2000/XP. It works for Hindi and Marathi Characters. The various characters which are available in it are :





Flowchart of the software is as follows:



The work consists of the following –

- Comprehensive study on the Hindi alphabet and writing system
- Formalizing the rules for editor
- Creating a testing dictionary
- Creating a testing corpus
- Studying the feasibility of unicode support in VB and evaluating the tools and techniques
- Creating a editor for typing in Hindi
- Doing the spell check of each word first in dictionary and if not found then in corpus

Organizations working for Hindi language technology and current status

The **Department of Information Technology (DIT), Ministry of Communications and Information Technology, Government of India**'s mission is to proliferate the use of Indian languages on computers, to overcome language barriers that restrict the nation's path to knowledge and development. Keeping this in mind and need for building software systems for Indian languages, the government started IT initiatives in Indian languages and knowledge based systems in many technology centres, universities and institutes funded by the **Technology Development for Indian Language (TDIL),** Programme of Govt. of India. IIT has released a **CD for Hindi Software Tools** (for free distribution) where organizations C-DAC, Prologix, Modular Infotech, Softview, Priya Informatics, Cyberscape Multimedia Limited, Chennai Kavigal, Cadgraf Digital, IIIT (Hyderabad) have contributed various tools for Hindi. It has a Hindi

language translation tool , Open Office which converts English sentences to Hindi sentences in the given domain. It also contains a Hindi spell checker, but that works only with MS office , and there are difficulties in using it properly.

Centre for Development of Advanced Computing C-DAC is a premier national institute of the Department of Information Technology (DIT), Ministry of Communications and Information Technology, Government of India. **MANTRA -** (**MA**chiNe assisted **TRA**nslation tool) for translation of English text to Hindi is one of its major achievements, which opens large gates of knowledge to the vast non-English speaking Indian population. It was launched in year 1999 by Government & Non-Profit Organizations , which is nominated by Intel Corporation company. This software follows a structural and lexical transfer approach, using semantic information, when required. It converts a complete English sentence into Hindi .

Indian Institute of Technology, Kanpur started its work on Computer Processing of Indian Languages and scripts in early seventies. The dominant role that the computer could play in solving complex problems posed by the plurality of languages and scripts in the country, was very well visualized by the researchers at IITK and some path-breaking work done in this area provided the foundations for the present R & D efforts.

Prof. R.M.K Sinha's researched on comparison of different possible coding schemes, keyboarding schemes, pros & cons of phonetic keyboarding and internal representation, its inherent transliteration capability, schema for Machine Translation using Interlingua, etc. He for the first time, gave strategy for English to Hindi and Hindi to English Transliteration (R.M.K.

Sinha, B. Srinivasan), Spell-checker (R.M.K. Sinha and K.S. Singh), segment display (R.M.K. Sinha). In 1990-92 Professor Sinha conceptualized the design of a Machine Aided Translation system for translation from English to Indian Languages. This system was named **ANGLABHARTI** and the underlying methodology was named as ANGLABHARTI Technology or ANGLABHARTI Approach. Currently, AnglaHindi, the English to Hindi MAT based on Anglabharti methodology, which accepts unconstrained text, has already been made available to the users and has been very well received.

AnglaBharti is a pattern directed rule based system with context free grammar like structure for English (source language). It generates a 'pseudo-target' applicable to a group of Indian languages. A set of rules obtained through corpus analysis is used to identify plausible constituents with respect to which movement rules for the 'pseudo-target' are constructed. A language specific text generator converts the 'pseudo-target' code into target language text.

International Institute of Information Technology, Hyderabad has a language technologies research centre (LTRC) , which is a centre of excellence in information technology education, research and training . The centre conducts basic and applied research on various aspects of natural language technology. In the area of MT, **Shakti** system is being developed from English to Indian languages. It combines rule-based approach with statistical approach and its modular design makes it possible to produce machine translation systems for new target languages rapidly. It can be tried for three target languages: Hindi, Telugu and Marathi.

Microsoft product **Hindi MS office 2003** has three dimensions of computing . First is Input ,second is Storage and third is Output. It can take input in any given language. The input goes in unicode and is saved unicode. Because of Unicode, the problem and compulsion of having special fonts to read a special Hindi text is rooted-out by Hindi MS office 2003 and standardization was established. It has MS Word having menus, spell check, translation etc. Spell-check has a dictionary containing 81,000 words and can be updated by new words. Outlook which can be customized in desired language. The words in menu which have easy equivalents in Hindi were translated in hindi, others were When we select a language to work in, the interface also goes to change automatically. MS Excel: we can install any font and it works without any problem. All the necessary languages can be input in a cell. We can give a name to files and folders in Hindi and search them also by the Hindi- names. Powerpoint: You can prepare an entire presentation in Hindi. Access: can store data in devanagari and can also search. Queries can also be run with devanagari key values. Multilingual charts. MS Publisher: Can publish Devnagari web content with ease. We can go directly in ms word and edit. Infopath: can create interactive form with help of this component and set a validation criterion.

Itranslator 99 developed by Omkarananda Ashram Himalayas , is a famous free utility which converts encoded text or text files (according to phonemes) into Devanagari or Transliteration script (True Type), which can be saved in Rich Text Format (.rtf), as web page (.htm), as ANSI text (.txt), or pasted via clipboard in any other application.

HindiWriter - The Phonetic Hindi Writer with auto word lookup is a software written (and freely distributed) by Devendra Parakh. It includes Hindi Spellcheck dictionary (with over 60,000 words) that works with Microsoft Word and open office.org 2.0

Baraha Software – It is a transliteration based word processing application which converts 'Indian language text written in English' to their own respective scripts. It is developed with an intention to provide a free software to enable and encourage Indians use their native languages on the computers. The first public release of Baraha software was made on January 1998 for Kannada language . Baraha can be effectively used for creating documents, sending emails and publishing web pages. This is a good software for typing but it does not have a spellchecking mechanism.

How is this work different

Itranslator 99 is using phonemes for conversion, but itrans is doing post transliteration i.e first the complete text is written in English and then we have to press convert, so that it is converted into Hindi. IIt does have an auto convert feature but that is good only for small size data. Moreover the screen flickers every time if something is pressed. It also does not have a spell checker. Some words like koI can not be written in it. The Phonetic Hindi Writer software written by Devendra Parakh needs support for Indic languages in your computer which roughly takes 10 MB of your space. Without this it can not be used.

The current work is different from existing research in the following ways -

1. The present work is intended as a web application which will be used by anybody anytime for Hindi editing and spell-checking. In its current version (1.0), it is a VB standalone system. Version 2.0 will be a web-application using a server language like ASP/JSP which will allow it to be used on internet. The user can than directly type in Hindi on internet using their normal keyboard, and get spell checking done, and save the file.
2. In this you are typing using a phonetic keyboard according to ITRANS transliteration scheme.
3. With this you can make files in Hindi itself which can be saved using the save option it has.
4. If some already existing Hindi file is there then that can be opened and more data can be appended to it.
5. It is of great use when you want to see how characters combine to make a word in Hindi.
6. It has a spell checker that checks each word automatically while you are typing. This checking can be done in a dictionary and if not found in it then in a corpus.
7. It also has the facility of adding words in your dictionary (from corpus or new words).
8. No separate fonts have to be downloaded or installed. The program uses the popular ITRANS scheme for transliteration.
9. Allows you to type in Hindi without having to remember a new keyboard layout

Chapter 2 LANGUAGE TECHNOLOGY IN HINDI

In India, computers that demand English as a working language are leaving hundreds of millions of potential users in the sidelines[1]. With over 500 million speakers, Hindi is the second-most spoken language after Chinese. As per a 1991 census report, Hindi was proclaimed by over 77 % of the Indian population as the "one language across the nation". Hindi has undergone considerable revolutions before it was adopted as the "Rajyabhasha" of India. However these hundreds of millions of people who prefer Hindi language need to take advantage of the IT revolution. India has been working on developing Language Technology, which researches computer systems, which understand and/or synthesize spoken and written human languages. Included in this area are speech processing (recognition, understanding, and synthesis), information extraction, handwriting recognition, machine translation, text summarization, and language generation[2]

Brief history of Hindi

Hindi's origins as a distinct identity can be traced back to the period in history commonly referred to as the Middle Indo-Aryan Period. This period, spanning between 600 BC and 1000 AD, represents a vital period of development in most present-day Indian languages. This period is divided into stages, demarcated along the lines such as the Emperor Asoka's Era, the age of the Prakrits and the birth of the Apabhramsa dialects in the 6th century AD. It represents a key period of linguistic development in history, tracing the paths taken by emergent languages between the Old Indo-Aryan

Era of Sanskrit and the Modern Indo-Aryan Era of Hindi and other Indian languages. [3]

Hindi, is based on the Khariboli dialect of the Delhi region. A more scholarly, Sankritized form of Hindi was believed to have developed primarily in Varanasi, the Hindu holy city and is believed to have been based on the Eastern Hindi dialect of that region. The term "Hindi" can thus be an ambiguous term at times.

The standard Hindi is heavily influenced by the Sankritization approach endorsed by the Indian Government after the Partition of 1947. Hindi, in its original form prior to Independence, shared a considerable degree of verbal similarity with Urdu. Hindi and Urdu were often referred together as a single entity called "Hindustani", along with several other languages like Awadhi, Bagheli, Bihari (and its dialects), Rajasthani (and its dialects) and Chhatisgarhi. However, the approach advocated in virtually every educational institution and medium of public information, employs a Sanskrit-oriented language developed by Indian scholars along the lines of the Varanasi dialect.

The effort to promote Hindi as the NOL has been partially successful, though Hindi has received a lot of popular support from the film and electronic media. The race for evolving and promoting a NOL, after independence is still going on. The Government of India worked on standardizing Hindi, instituting the following changes:

- standardization of Hindi grammar: In 1954, the Government of India set up a committee to prepare a grammar of Hindi; The committee's report was released in 1958 as "A Basic Grammar of Modern Hindi"
- NLP Lab : In computing of Hindi or other Indian languages

- standardization of Hindi spelling
- standardization of the Devanagari script by the Central Hindi Directorate of the Ministry of Education and Culture to bring about uniformity in writing and to improve the shape of some Devanagari characters.
- scientific mode of transcribing the Devanagari alphabet
- incorporation of diacritics to express sounds from other languages.

- Lekhika : a platform of independent word processor with tools for Hindi:
    - Dictonary
    - Spell checker in local languages
    - Desktop utilities like calendar, calculator etc.
    - ISCII to Unicode converts
    - Translation

During the annual "Hindi Divas" ceremony on 14 September, 2005, the Government launched a chip version of Hindi translation software, on mobile phones and the internet which made Hindi learning easy and high-tech. It also launched MANTRA (Machine assisted Translation Tool) software that facilitate translation of Government documents from English to Hindi, and is put to use in the fields of administration, health and agriculture. The LILA (Learn Indian Languages through Artificial Intelligence) software also came that helped non Hindi speakers to learn Hindi on mobile phones and the internet through Tamil, Telugu, Kannada, Bengali and English. [3]

2.1 **Issues in Hindi computing**

The Hindi language consists of 11 vowels and 35 consonants and is written in Devanagari script. Hindi is equipped with a rich consonant system, with about 38 distinct consonant units of sounds. However, the number of phonemes, as these units of sound are called, cannot be accurately determined, owing to the large number of dialects that exist, which employ many derivatives of the consonant repertoire. However, the traditional core of the consonant system is directly inherited from Sanskrit with an additional seven sounds conjectured to have originated from Persian and Arabic.

Hindi also employs a large number of modifying sounds. The "anuswara" is employed as a dot above the vowel to produce a deprecated sound. The "visarga" (denoted by a ":") is also exclusively employed with vowels. However, though used in conjunction with vowels, the "visarga" and "anuswara" are both considered to be consonants. Nasalized sounds are represented by a "chandra bindu" sign above a vowel. The "ardha chandra bindu" sign is also employed with vowels to indicate the ‘o' sound in English. The "khutma" is a sound used with great frequency with consonants. Usually represented as a dot above a consonant, it is believed that the "khutma" was introduced due to the influence of the Urdu, Persian and Arabic languages in predominantly Hindi-speaking regions.

However, despite its leading status as the second-most spoken language in the world. Hindi is not employed at concurrent levels over the internet and in basic home computing utilities. There are different standards, file formats, fonts which are not interchangeable in ordinary circumstances**.** But, emphasis in this regard has returned with an increased vigor in recent

times. With the proper guidance, support and governance, Hindi may soon ascend to become a true language of the world in every respect.

In order to achieve national unity and integration in the face of the linguistic and cultural diversity, the founding fathers of our constitution had identified Hindi as the Official Language of the Indian Union. According to the Official Language Act, all Central Government communications have to be made simultaneously available both in Hindi and English, as English continues to be the associate official language. Accordingly the bulk of official business is initiated and conducted in English. Presently, the translation work is executed manually by a large network of translators positioned in all Government Departments and Public Sector Undertakings. However, the translators find it difficult to cope with the massive translation requirement leading to inordinate delays.

Difficulties faced are:

- Keyboard, this is an aspect where there is a yawning difference between the characteristics of English alphabet and the characteristic of the alphabet in Hindi language. Different models of adapted keyboards have been proposed and used with varying degrees of success. Lack of global standards hindered the progress. The imperative need for upward compatibility has slowed down the pace of defining standards.
- the *phonetic mechanism of some sounds peculiar to Hindi (e.g. rda, dha etc)* The distinction between aspirated[*1] and unaspirated consonants will

*1 Aspiration is the strong burst of air that accompanies either the release or, in the case of preaspiration, the closure of some obstruents. In English, the *t* should be aspirated in *tore* and unaspirated in *store*.

be difficult for English speakers. In addition, the distinction between dental[*2] and retroflex[*3] consonants will also pose problems. English speakers will find that they need to carefully distinguish between four different d-sounds and four different t-sounds.

- *pronunciation of vowels*: In English, unstressed vowels tend to have a schwa[*4] quality. The pronunciation of such vowels in English is changed to an "uh" sound; this is called reducing a vowel sound. The second syllable of "unify" is pronounced /ə/, not "ee." The same for the unstressed second syllable of "person" which is also pronounced /ə/ rather than "oh." In Hindi, English-speakers must constantly be careful not to reduce these vowels.
- In this respect, probably the most important mistake would be for English speakers to reduce final "ah" sounds to "uh." This can be especially important because an English pronunciation will lead to misunderstandings about grammar and gender. In Hindi, "vo bolta hai" is "he talks" whereas "vo bolti hai" is "she talks." A typical English pronunciation in the first sentence would be "vo boltuh hai," which will be understood as "she talks" by most Hindi-native speakers.
- The 'a' ending of many Sanskrit and Sanskrit borrowed masculine-gender words, due to Romanization[*5], is highly confused by non-native speakers, because the short 'a' is dropped in Hindi. There are exceptions, of course, if the devanagari script itself dictates the

---

*2 Dental is a consonant that is articulated with the tongue against the upper teeth, such as. (त् थ् द् ध् न)

*3 Retroflex is a consonant where the tongue is placed behind the alveolar ridge, and may even be curled back to touch the palate region of the mouth.such as (ट् ठ् ड् ढ् ण)

*4 schwa mean an unstressed and toneless neutral vowel sound .Such vowels are often transcribed with the symbol <ə>, regardless of their actual phonetic value.

*5 romanization is the representation of a word or language with the Roman (Latin) alphabet, or a system for doing so, where the original word or language uses a different writing system

diacritical mark for the vowel "long ā" at the end of certain masculine words, like BrahmA (äüa).

- The variation between the phonetic and graphic representation in Devanagari when computed through the regular KB (ASCII).
- Market is fragmented having different Companies with different segment of Indian Language Computing
- Market sluggish have proprietary Font, Encoding Scheme and Non-Standard Key-Board
- Lack of Integrated Computing Environment which no single Indian Company is offering
- Lack of Market growth having non-English speaking Literate people .Also cost of hardware and Internet Access is very high
- Lack of financial muscle where none of the Indian IT giant in Indian language computing
- Lack of Availability of venture fund having large number of users but hardly any customers

2.2 Usefulness of Editor/Spell Checkers

Editors and Spell checkers in Indian languages are the basic tools that need to be developed urgently. A spell checker is a software tool that identifies and corrects any spelling mistakes in a text. Spell checkers can be combined with other applications or they can be distributed individually. Whatever be the case, a spell checker should handle a word and identify the spelling error

in it and should help in correcting the errors. Spell checkers are mostly bundled as a component of word processors. Currently spell checkers are available for English, and most major foreign languages. Spell checking initiatives in Indian languages are having very low success rates. It is also believed that complete spell checking is not possible in any language. However, it is a notable fact in Indian languages there is not even a basic spell checking tools[2] .MS Office has done work in this field. It has hindi MS Word having hindi interface which is have spell checker, which underlines the wroung word in the file. It also has auto correct along with treasures giving synonyms , antonyms etc. The size of dictionary is nearly 80000 words.

Constraints in developing spell checkers:

There are many basic difficulties in developing spell checkers in Indian languages. First, we must develop corpus and dictionary for every language. Spell checkers, dictionaries and other linguistic tools are only developed based on the corpora only. Therefore, when we use successfully the corpus in our languages, we will be able to develop other linguistic tools. So, we must try to get words from corpora in our languages first. The time spent and work done towards using corpora will lead to other possible developments. So, every need of the language will be achieved when words from corpora are checked and added to dictionary if wished.

Chapter 3 HINDI LANGUAGE AND UNICODE

3.1 ISCII and UNICODE

Hindi written in Devanagari script is the official language of the Indian Union according to the Article 343 of the constitution of India.

Indian Script Code for Information Interchange (ISCII)

ISCII was proposed in the eighties and a suitable standard was evolved by 1991. Here are the salient aspects of the ISCII representation.

- It uses 8 bit code which is an extension of the 7 bit ASCII code
- It is a single representation for all the Indian Scripts.
- codes have been assigned in the upper ASCII region (160 - 255) for the aksharas of the language.
- The scheme also assigns codes for the Matras (vowel extensions).
- Special characters have been included to specify how a consonant in a syllable should be rendered. Rendering of Devanagari has been kept in mind.
- A special Attribute character has been included to identify the script to be used in rendering specific sections of the text.

It must be stated here that ISCII represents the very first attempt at syllable level coding of Indian Language aksharas. Unfortunately, outside of CDAC

which promoted ISCII through their GIST technology, very few seem to use ISCII.[4]

Unicode for Indian Languages

Unicode was the first attempt at producing a standard for multilingual documents. Unicode owes its origin to the concept of the ASCII code extended to accommodate International Languages and scripts. It is increasing being accepted as a standard for Information Interchange worldwide as most of the major IT Companies have declared their support for it. Unicode for Indian Languages use ISCII-88 and not ISCII-91 which is the latest official standard. It was felt necessary that Indian Government should represent UNICODE Consortium for necessary modification in the code pertaining to Indian languages script and hence Department of Information Technology became full member of Unicode Consortium with voting right. The implementation of Indic scripts by Unicode has been done in such a way that phonetic transliteration between Indic scripts will be easy as code points for corresponding characters are well defined.

Unicode uses a 16 bit encoding that provides code point for more than 65000 characters (65536). Unicode Standards assigns each character a unique numeric value (no matter what the platform, no matter what the program, no matter what the language) and name. The Unicode standard and ISO10646 Standard provide an extension mechanism called UTF-16 that allows for encoding as many as a million. Presently Unicode Standard provide codes for 49194 characters.

Unicode policy for character encoding

Unicode consortium has laid down certain policy regarding character encoding stability by which no character deletion or change in character name is possible only annotation update is possible

1. Once a character is encoded, it will not be moved or removed.
2. Once a character is encoded, its character name will not be changed.
3. Once a character is encoded, its canonical combining class and decomposition (either canonical or compatibility) will not be changed in a way that would affect normalization.

4. Once a character is encoded, its properties may still be changed, but not in such a way as to change the fundamental identity of the character.
5. The structure of certain property values in the Unicode character database will not be changed.

Issues when dealing with Unicode for Indian scripts [4]

- There is no allocation for half characters or subjoined forms or for conjoined forms. However, those are implemented as functions (by use of Halant). Devanagari is also implemented as a function.
- Rendering text in a manner that is uniform across applications is quite difficult. Windowing applications with cut, copy/paste features suffer due to problems in correctly identifying the width of each syllable on the screen. Also, applications have to worry about specific rendering issues when modifier codes are present. How applications run into difficulties in rendering even simple strings is illustrated with examples in a separate page.

- Interpreting the syllabic content involves context dependent processing, that too with a variable number of codes for each syllable.
- A complete set of symbols used in standard printed text has not been included in Unicode for almost all the Indian scripts.
- Displaying text in scripts other that what Unicode supports is not possible. For instance, many of the scripts used in the past such as the Grantha Script, Modi, Sharada etc., cannot be used to display Sanskrit text. This will be a fairly serious limitation in practice when thousands of manuscripts written over the centuries have to be preserved and interpreted.
- Transliteration across Indian scripts will not be easy to implement since appropriate symbols currently recommended for transliteration are not part of the Unicode set.
- The unicode assignments bear little resemblance to the linguistic base on which the aksharas of Indian scripts are founded. While this is not a critical issue, it is desirable to have codes whose values are based on some linguistic properties assigned to the vowels and consonants, as has been the practice in India.
- It does not really blend with the syllabic writing systems used in india, much less provide the means to express linguistic content without ambiguity and in a manner that ties in well with our own understanding of languages.

3.2 Technical Characteristics

Hindi Alphabet Characteristics

The Hindi language, in common with Sanskrit, Marathi, Konkani, Nepali, Maithili, (of late Dogri, Santhali, Bodo) and many other north Indian dialects, is written in the Devanagari script which is also the accepted all-India script for Sanskrit.

The alphabets consist of 11+1 = 12 vowels and 35 consonants, as given below:

Vowels (स्वर)

अ आ इ ई उ ऊ ऋ ए ऐ ओ औ ऑ

a A i ii u U RRI e ai o au O

(i) ऑ (O) indicates English O in words like office (ऑफिस), Coffee (कॉफी), Sauce(सॉस)

(ii) The vowel ऋ occurs only in Sanskrit words borrowed into Hindi. In Hindi ऋ is pronounced as 'r + i = ri' whereas it is pronounced as 'r + u =ru' in some south Indian languages.

(iii) ं (anuswara) and ः (visarga) are often included in the list of vowel letter and are usually written as अं and अः, but so as Hindi is concerned, they are mostly used with consonant.

(iv) For all practical purposes, अ–आ , इ–ई and उ–ऊ may be regarded as pairs of short and long vowels. ए–ऐ and ओ–औ are all long vowels.

Dependent Vowel Signs (Matras)

To indicate a vowel sound other than the implicit one, a vowel sign (matra) is attached to the consonant. Thus, there are equivalent matra for all the vowels. Explicit appearance of a matra in a syllable overrides the inherent vowel. These matra can exist alone below, to the right or to the left of the consonant to which it is applied to. The 'matra' mostly come after the consonant letters as below:-

आ ⇒ ा , इ ⇒ ि , ई ⇒ ी , उ ⇒ ु , ऊ⇒ ू , ऋ ⇒ ृ ,

ए ⇒ े , ऐ⇒ ै , ओ ⇒ ो , औ ⇒ ौ

अ (a) has no matra . The matra ा (आ), ि (इ), ो (ओ), ौ (औ)are written after the consonant whereas ी (ई) is written before, ु (उ), ू (ऊ) and ृ (ऋ)are written below and े (ए) and ै (ऐ) are written above. Thus

क् + आ = का

क् + इ = कि

क् + ई = की

क् + उ = कु

क् + ऊ = कू

क् + ए = के

क् + ऐ = कै

क् + ओ = को

क् + औ = कौ

क् + ऋ = कृ

With र् (r) उ and ऊ matra are written in an exceptional form i.e. र् + उ = रु

and र् + ऊ = रू

It may be noted that the matra is tagged on to the consonant letter and is never written in full. Thus, क् + इ (k + i) will not be written as कइ but as कि

is the correct form of matra . In कई (kaI) forms, इ is a vowel in full form, not the matra.

(b) Consonant Letters व्यंजन

क	ख	ग	घ	ङ	च	छ	ज	झ	ञ
ka	kha	ga	gha	~Na	ca	cha	ja	jha	~na
ट	ठ	ड	ढ	ण	त	थ	द्	ध	न
Ta	Tha	Da	Dha	Na	ta	tha	da	dha	na
प	फ	ब	भ	म	य	र	ल	व	श
pa	pha	ba	bha	ma	ya	ra	la	va	sha

ष	स	ह
Sa	sa	ha

The following points are to be noted :

(i) अ (a) is a inherent in each consonant letter.

(ii) ष (Sh) occurs only in Sanskrit words borrowed into Hindi.

(iii) ङ (~N), ञ (~n), ड़ (D) and ढ़ (Dh) never occur in the beginning of the word; ङ and ञ and never occur independently themselves. They are always combined with a following consonant.

The first twenty-five consonants i.e. (ka) to (ma) are divided into five categories (Varga):

ka category (क वर्ग) – क ख ग घ ङ

ca category (च वर्ग) – च छ ज झ ञ

Ta category (ट वर्ग) – ट ठ ड ढ ण

ta category (त वर्ग) – त थ द ध न

pa category (प वर्ग) – प फ ब भ म

The fifth letters of each category, ङ ञ ण न and म are nasals.

Rest of the consonants can be placed in an unmarked category.

Consonant Conjuncts

The device of conjoining consonant letters was used in writing Sanskrit to indicate the pronunciation of consonants without an intervening inherent a.

Traditional conjunct consonant letters are क्ष (ksha), ज्ञ (jna), त्र (tra), श्र (shra) and द्य (dya). It is to be noted that in Hindi ज्ञ is pronounced as ग्य (gya).

Traditional conjunct consonant letters are very common in Sanskrit loanwords.

The common conjuncts are listed as given below:

क्क (kka), क्ख (kkha), क्त (kta), क्य (kya), क्ल (kla), क्व (kva), ख्य (khya), ग्द (gda), च्च (cca), ण्ठ(ëöha), ण्य (ëya), त्त (tta), त्त्व (ttva), त्थ (ttha), न्न (nna), न्म (nma), न्य (nya), प्त (pta), प्य (pya), प्र (pra), ब्ज (bja),

ब्द (bda), ब्ब (bba), म्म (mma), म्य (mya), य्य (yya), ल्ह (lha), व्य (vya), श्क (shka), ष्क (ñka), ष्त (ñta), ष्ण (ñëa), ट्ट (ööa), ड्ड (òòa), द्द (dda) द्ध (ddha), द्व (dva), ह्न (hna), ह्म (hma), ह्र (hra), ह्ल (hla), ह्व (hva)

It is to be noted that conjuncts involving initial r are written with a special superscript form for 'r': Thus र् + क = र्क (rka), र् + म = र्म (rma), र् + ष = र्ष (rña) etc. written at the end of its syllable such as र् + ष + ा =र्षा (rñä),

र् + थ +ी = र्थी (rthé), र् + य + ो = र्यो (ryo).

But when 'र' (r) follows a consonant letter having vertical stroke below and to the left of the stroke: क् + र = क्र (kra), म् + र = म्र (mra), द् + र = द्र (dra).

When preceeded by ट थ ड ढ and छ it is written as given:

ट् + र = ट्र (Tra) , ड् + र = ड्र (Dra)

Halant (्)

The Halant is the vowel अ (a) omission sign. It serves to cancel the inherent vowel of the consonant to which it is applied. When the simple consonant

without the inherent 'अ' is specifically be expressed, (a sign right slanting stroke) called Hal or Halant, is put below the letter. But it should be noted that halant specifically is not used below the consonant
letters having vertical stroke. In these letters, only vertical stroke is removed. E.g. ख ⇒ ख्, ग ⇒ ग्, च ⇒ च्, त ⇒ त्, प ⇒ प्, ल ⇒ ल्. Even the vertical stroke in between the consonant letters like क and फ are not removed, only a part of the right portion of the letter is removed.

E.g. क ⇒ क्, फ ⇒ फ्

Halant is used on those consonant letters, which have no vertical stroke in them. E.g. छ् (ch), ट् (ö), त् (t), ढ् (åh), ठ् (öh), र् (r), ह् (h) When the inherent अ occurring at the end of the some of the words is silent, such as अर्थात् (arthät), परिषद् (pariñad), halant can be used below the consonant.

Nukta (.)

A subscript dot (nukta) is sometimes used with certain Devanagari letters to denote sounds of non-Indian origin in loan words. Fine of the consonant letters in Devanagari with nuktas (diacritic mark) represents some of persian, Arabic and English sounds. This usage in common, but not obligatory, the more so since the great majority of Hindi speakers tend to

replace these sounds with sounds of Indian origin. E.g . क़ (qa) , ख़ (ûa) , ग़ (úa) , ज़ (za) , फ़ (fa). In the letters क़ (qa) , ख़ (ûa) , ग़ (úa), nukta or dot is not mostly used as majority of Hindi speakers replace them with क (ka), ख (kha), ग(ga).

Anusw**a**ra ं

Anusw**a**ra indicates a nasal sound. It is a 'homorganic' nasal representing ङ ञ ण न् and म् belonging to any of the five ka, ca, öa , ta and pa categories. When the nasal consonant precedes the consonant of the same category, it will be taken as anuswara. For example, अङ्क = अंक (aMka) मण्डन = मंडन (maMDana) हिन्दी =हिंदी (hindI) लम्बा =लंबा (laMbA). Here 'N' is the sign of anuswara. It is to be noted that in Sanskrit anuswara ं is not usually used as a homorganic nasal for nasal consonants ङ् ञ् ण् न् म् . These nasal consonants are used themselves.

It is to be mentioned that before य (ya), व (va), र (ra), ल (la), श (sha), स (sa), anuswara (ं) is used preceding them. Such as संयम (saMyama) ,

संवाद (saMvAda) **संरक्षण** (saMrakSaNa) , **संलाप** (saMlApa) , **संसार** (saMsAra).

Here 'स' represents prefix **सम** But in reduplicated form of the nasal letters such as **अन्न** (anna) , **सम्मान** (sammAna) , **तुम्हारा** (tumhArA) anuswara is not used. Similarly in **य** (ya) , **व** (va) , **ह** (ha), anuswara is not used and the nasal letter will come in its original form. E.g. **अन्य** (anya) , **साम्य** (sAmya) , **सम्न्वय** (samnvaya) and **कान्हा** (kAnhA).

Anunasik ँ

The superscript sign 'chandrabindu' (ँ) represents anunasik sound. It is placed above a vowel denoting vowel nasality through the nose. E.g.

हाँ **आँख** (äðkha)

हुँ **पुँछ** (puðcha)

when a consonant has a vowel sign i.e. matra above its headline, the chandrabindu (ँ) will be used as 'anuswara' although this anuswara represents anunasik sound i.e. chandrabindu. E.g. **सिंचाई** (siïcäé) ,

खिंचना (khiïcanä) , भेंट (bheNTa) , भैंस (bhaIàsa) , पोंछना (poïchanä) , रौंदना (raundanä).

Visarga ः

The sign (ः) called visarga is written in a linear way. It has the sound of a voiced ह (ha) in Hindi. It occurs almost exclusively in Sanskrit words borrowed into Hindi. E.g. अतः (ataù) , प्रायः (präyaù) , सामान्यतः (sämänyataù)

Numerals

In Devanagari, numerals are used as

० १ २ ३ ४ ५ ६ ७ ८ ९

But at All India level, Roman numerals are used :

0 1 2 3 4 5 6 7 8 9

also marks चौथा (1/4) cauthä , आधा (1/2) ädhä , पौना (3/4) paunä, सवा (1¼) savä, डेढ़ (1½) DeRha , and ढाई (2½) òhäé

Punctuation Marks (Virama)

In Hindi, Sentences are concluded with the vertical mark (।) called purn virama or danda.

The vertical stroke is also used for marking the end of the first hemistich i.e. half verse. For marking the end of the verse itself two vertical strokes called dergh viram may be used. E.g.

पोथी पढ़ पढ़ जग मुआ पंडित भया न कोय।

ढ़ाई आखर प्रेम का पढ़े सो पंडित होय॥

pothé paåha paåha jaga muA paëòita bhayA na koya,
åhäé äkhara prema kä paåhe so paëòita hoya.

Some modern writers in Hindi prefer to use the English punctuation mark full stop to the vertical stroke.

The rest of the punctuation marks, viz. Comma [,] (alpviram), semi colon [;] (ardh virämm), colon [:] (virämcinha), hyphen [-] (yojak), dash [-] (nirdeshak), single and double inverted commas [‘ ’ and “ ”] (uddharan cinha), question mark [?] (prashna sücak), bracket [( )] (koñöak) etc. have been borrowed from English. However, the colon [:] is usually avoided, lest it should be confused with the ‘Visarga’ sign.

Ancient Signs

ॐ (Om), श्री (çré) are mostly used in Hindi also as in Sanskrit.

Character Set Consideration

• Hindi Characteristics

Devanagari script is syllabic and is written from left to right. The characters of the script are given below in their traditional order called ‘Varnmala’ accompanied by Roman characters.

Vowels
Syllabic form:

अ आ इ ई उ ऊ ऋ ए ऐ ओ औ ऑ

a ä i é u ü å e ai

o au ô

Intra-Syllabic form:

ा, ी, ि, ु, ू, ृ, े, ै, ो, ौ

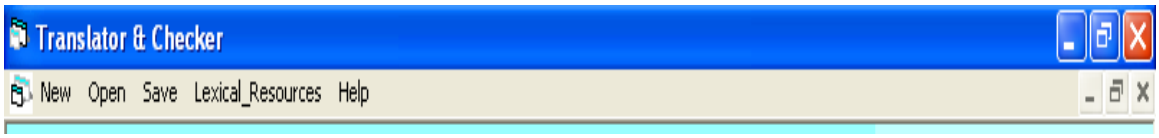
Consonants

Voiceless unasp.	**Voiceless asp.**	**Voiced unasp**	**voiced asp.**	**Nasal**
क ka	ख kha	ग ga	घ gha	ङ ìa
च ca	छ cha	ज ja	झ jha	ञ ïa
ट Ta	ठ Tha	ड Da	ढ Dha	ण Na
त ta	थ tha	द da	ध dha	न na
प pa	फ pha	ब ba	भ bha	म ma
य ya	र ra	ल la	व va	श ça

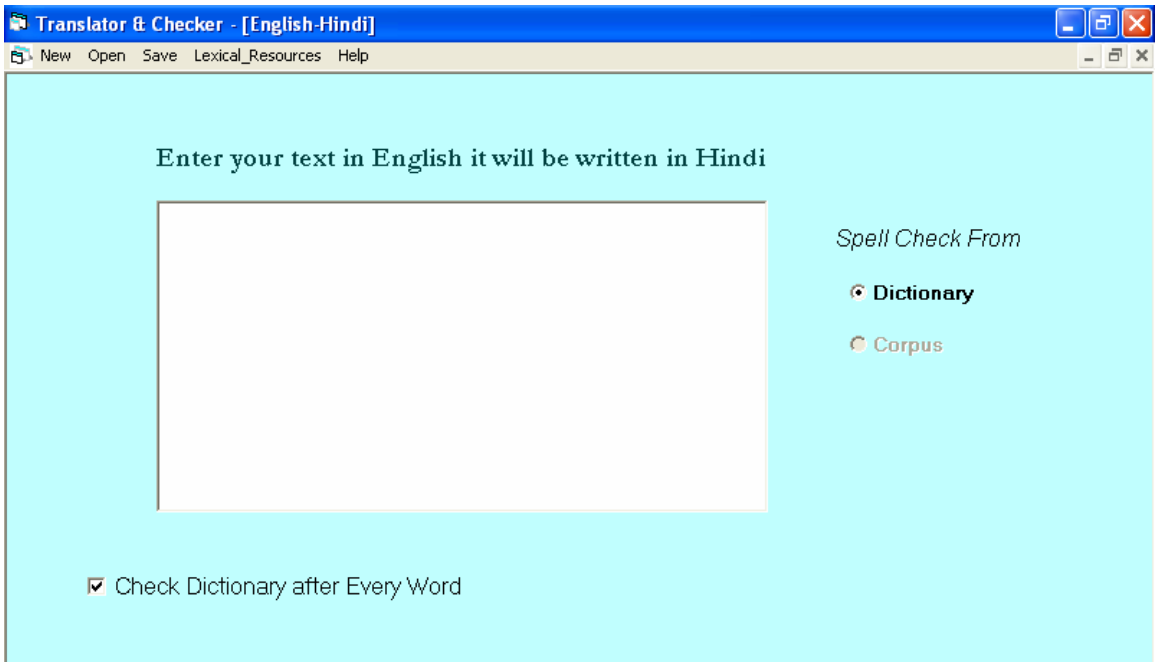
ष ña स sa ह ha ड़ åa ढ़ åha

Chapter 4 HINDI EDITOR WITH SPELL CHECKER

It has a MDI (Multiple document interface) form which have many children forms. The menu has the following options.



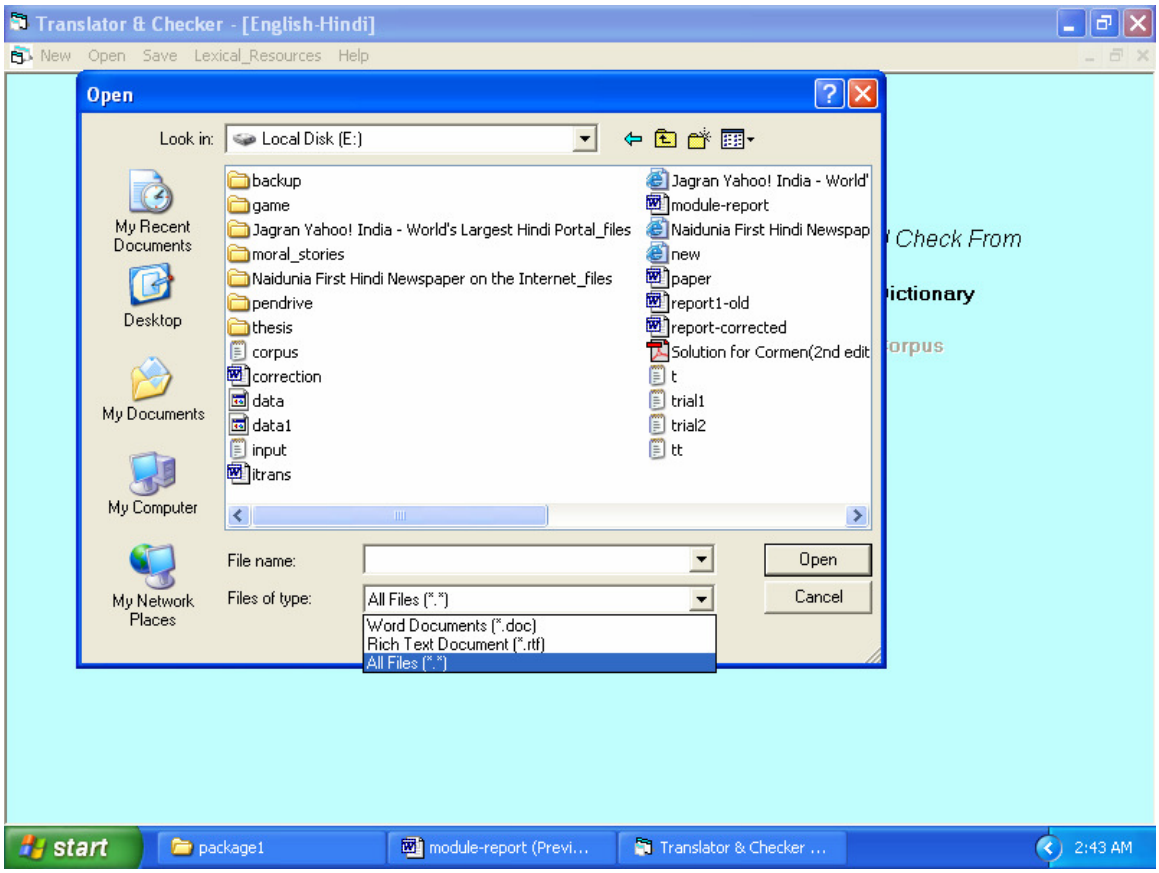
New – you can open a window which has a blank writing space.



code snippet

```
Private Sub newd_Click()
Unload convert1
convert1.Show
End Sub
```

Open – If you already have a file written in Hindi (using unicode) then you can give the path of that file (open the file) and the contents of the file will be displayed in the writing space where you can append more data to it.



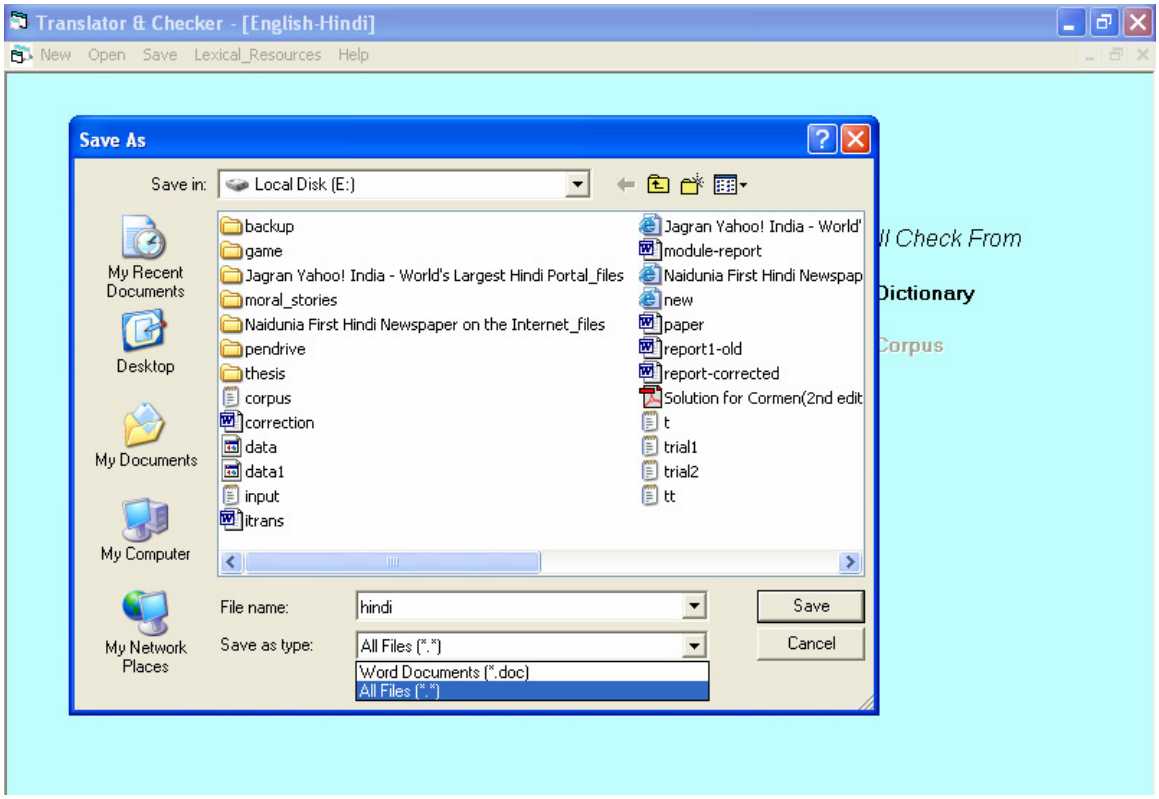
code snippet

```
Private Sub opend_Click()
Dim data As String
Dim FF As Long
Dim FilePath As String
Dim fsys As New FileSystemObject
Dim tstream As TextStream

cdlopend.Filter = "Word Documents (*.doc)|*.doc|Rich Text Document (*.rtf)|*.rtf|All Files (*.*)|*.*"
cdlopend.ShowOpen
FF = FreeFile()
FilePath = cdlopend.FileName
If FilePath <> "" Then
```

```
Set tstream = fsys.OpenTextFile(FilePath, ForReading, False, True)
Do While tstream.AtEndOfStream = False
  data = tstream.ReadLine
  convert1.txtinput.Text = data
Loop
tstream.Close
End If
End Sub
```

Save – whatever is written in the writing space that is copied in the specified file.



code snippet

```
Private Sub saved_Click()
Dim fsys As New FileSystemObject
Dim tstream As TextStream
Dim FF As Long
```

```
Dim FilePath As String
cdlsaved.Filter = "Word Documents (*.doc)|*.doc|All Files (*.*)|*.*"
cdlsaved.ShowSave
FF = FreeFile()
FilePath = cdlsaved.FileName '"C:\Temp\Output.txt"
If FilePath <> "" Then
Set tstream = fsys.CreateTextFile(FilePath, True, True)
tstream.WriteLine (convert1.txtinput.Text)
MsgBox "data added"
tstream.Close
End If
End Sub
```

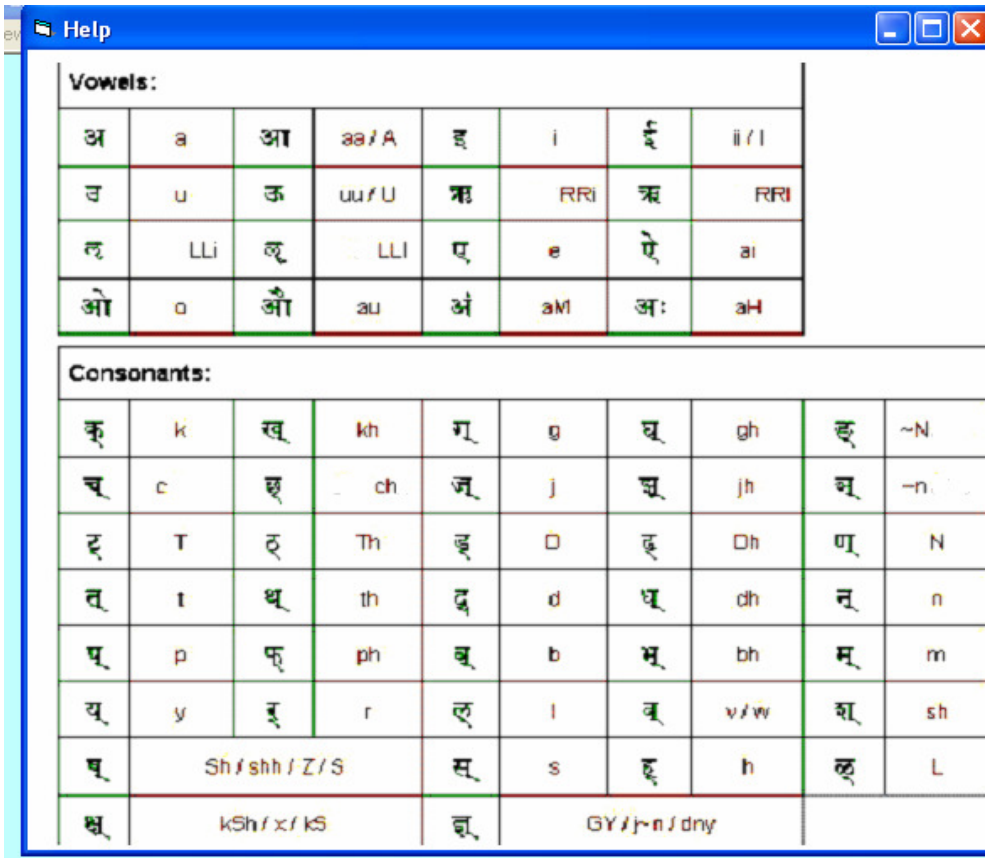
Lexical Resources – it is used to specify the path of dictionary and the corpus.

code snippet

```
Private Sub corpus_Click()
Dim fsys As New FileSystemObject
Dim tstream As TextStream
Dim FF As Long
   'MsgBox "Please give the path of the dictionary", , "Dictionary Path"
   cdldic.Filter = "Word Documents (*.doc)|*.doc|Rich Text Document (*.rtf)|*.rtf|All Files (*.*)|*.*"
   cdldic.ShowOpen
   FF = FreeFile()
   corpusfile = cdldic.FileName
End Sub
```

```
Private Sub dict_Click()
Dim fsys As New FileSystemObject
Dim tstream As TextStream
Dim FF As Long
   'MsgBox "Please give the path of the dictionary", , "Dictionary Path"
   cdldic.Filter = "Word Documents (*.doc)|*.doc|Rich Text Document
(*.rtf)|*.rtf|All Files (*.*)|*.*"
   cdldic.ShowOpen
   FF = FreeFile()
   dictfile = cdldic.FileName
End Sub
```

Help – It tells you about the convention followed in the project.



code snippet

```
Private Sub help_Click()
```

```
frmhelp.Show 1
End Sub
```

4.2 Module Description

Various modules are written to check what the typed character is

1. vowel

```
Public Function vowel(ByVal ch As Integer) As Boolean
If ((ch = 97) Or (ch = 101) Or (ch = 105) Or (ch = 111) Or (ch = 117))
Then
   vowel = True
Else
   vowel = False
End If
End Function
```

2. vayanjan

```
Public Function vayanjan(ByVal ch As Integer) As Boolean
If ((ch > 2324) And (ch < 2362)) Then
   vayanjan = True
Else
   vayanjan = False
End If
End Function
```

3 consonant

```
Public Function consonant(ByVal ch As Integer) As Boolean
If (((ch >= Asc("a")) And (ch <= Asc("z"))) And (vowel(ch) = False))
Then
   consonant = True
Else
  consonant = False
End If
```

```
End Function
```

4 cap_vowel

```
Public Function cap_vowel(ByVal ch As Integer) As Boolean
If ((ch = Asc("A")) Or (ch = Asc("I")) Or (ch = Asc("U"))) Then
   cap_vowel = True
Else
   cap_vowel = False
End If
End Function
```

5 cap_consonant

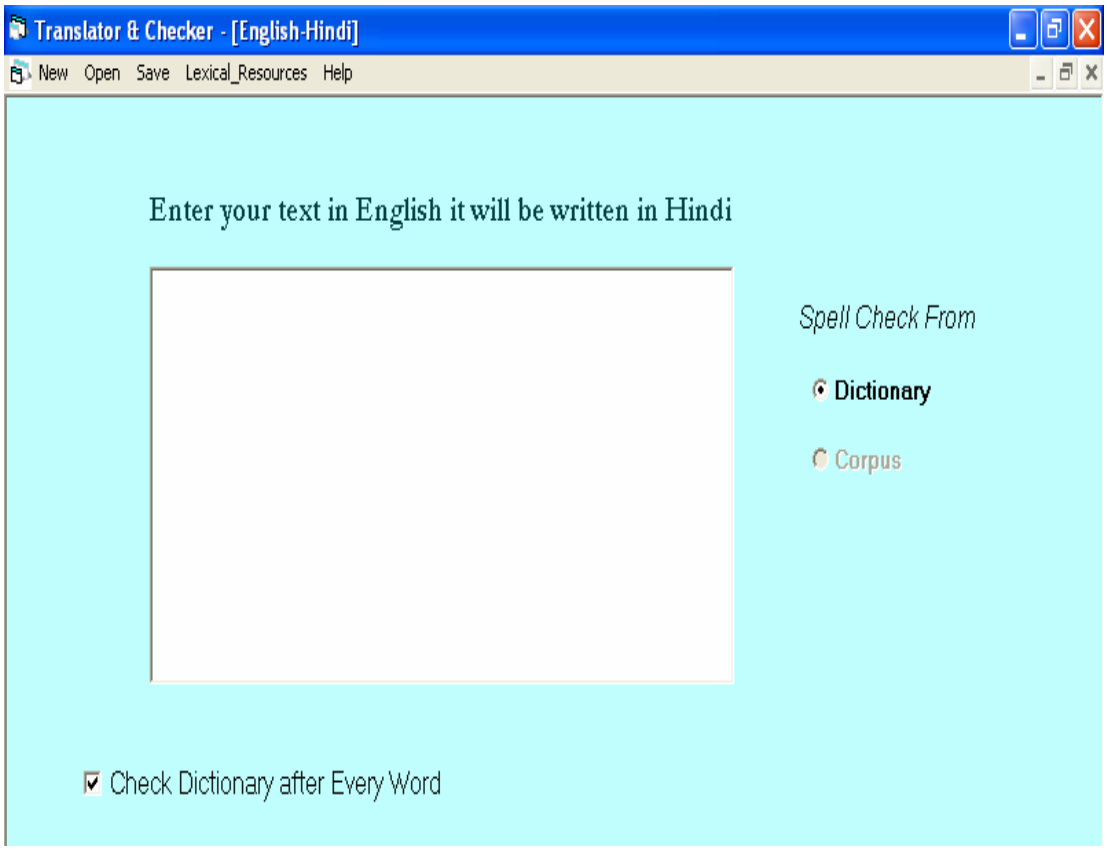
```
Public Function cap_consonant(ByVal ch As Integer) As Boolean
If ((ch = Asc("T")) Or (ch = Asc("D")) Or (ch = Asc("N")) Or (ch =
Asc("S")) Or (ch = Asc("Z")) Or (ch = Asc("L"))) Then
   cap_consonant = True
Else
   cap_consonant = False
End If
End Function
```

6 number

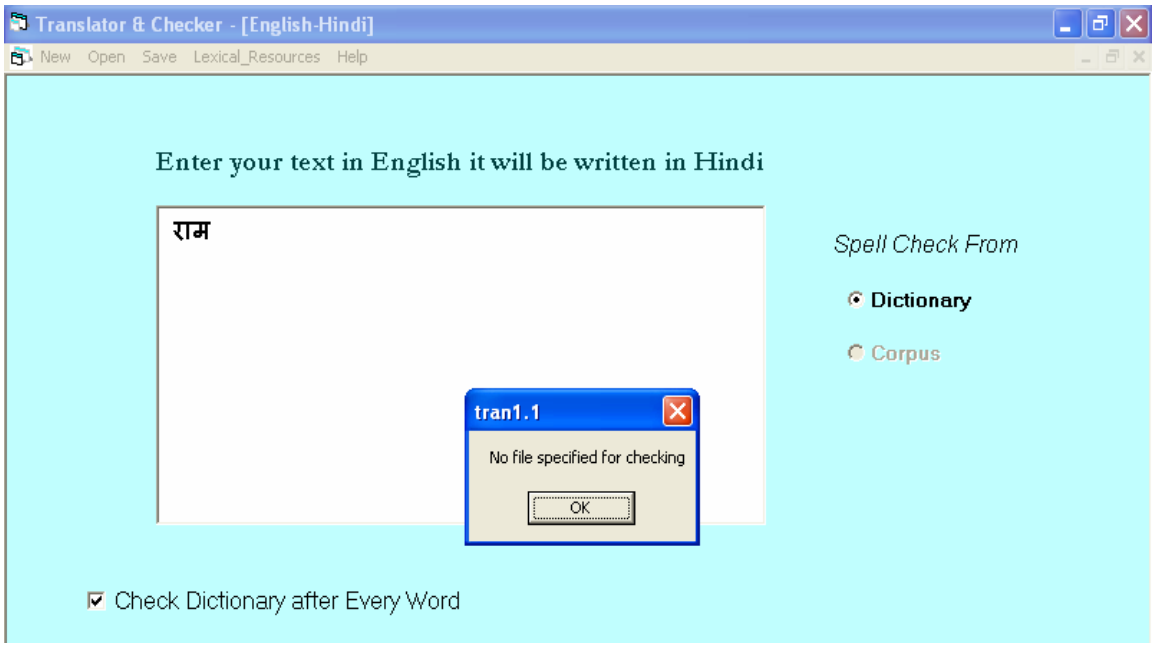
```
Public Function number(ByVal ch As Integer) As Boolean
If ((ch >= 48) And (ch <= 57)) Then
   number = True
Else
   number = False
End If
End Function
```

4.3 Test dictionary

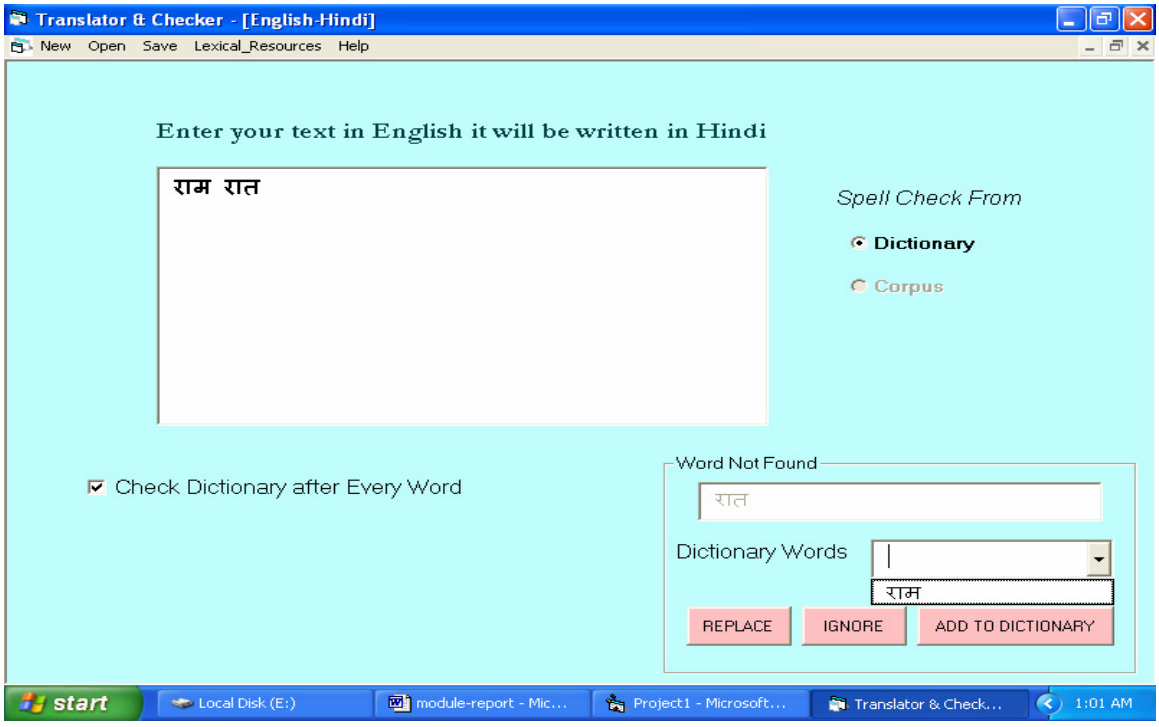
You have a check Box “check dictionary after every word “ which ask you whether spell checking should be done or not. By default spell checking will be done but if you do not select checkbox then it will behave like as a simple Editor without spell checker.



When spell checker is on , then soon as the typing is started the checking process in the dictionary also starts. i.e when we complete a word (by giving a space) the word is automatically checked in the dictionary. Therefore before writing anything we should give the path of a dictionary by opening the dictionary. If we don’t give path before writing then as soon as word is completed a message is shown to add the dictionary like this.



Once the dictionary is present each word is checked. If the word is available then nothing happens , otherwise the word is shown like this.



code snippet if word is not found

```
If found = False Then 'not in dictionary not in corus
    txtword.Text = lastword
```

```
        cmbdata.Clear
        Set tstream = fsys.OpenTextFile(dictfile, ForReading, False, True)
        Do While (tstream.AtEndOfStream = False)
           ch = tstream.Read(1)
           If (Mid(lastword, 1, 1) = ch) Then
              ch = ch + tstream.ReadLine
              cmbdata.AddItem ch
           End If
        Loop
        tstream.Close
        Frame1.Visible = True
    End If
```

We have then three operations, which we can perform these are

1. Ignore – nothing will happen we will move back to our writing space

```
Private Sub cmdignore_Click()
Frame1.Visible = False
End Sub
```

2. Add to Dictionary – the word will be added to the dictionary

```
Private Sub cmdadd_Click()
Dim fsys As New FileSystemObject
Dim tstream As TextStream
If bdict.Value = True Then
   ofile = dictfile
Else
   ofile = corpusfile
End If
Set tstream = fsys.OpenTextFile(ofile, ForAppending, False, True)
```

```
tstream.WriteLine (txtword.Text)
cmbdata.AddItem txtword.Text
tstream.Close
Frame1.Visible = False
End Sub
```

3. Replace – may be the word you have written is incorrect and you want to change it (replace it) with some word which is already in the dictionary, then you can use this option.

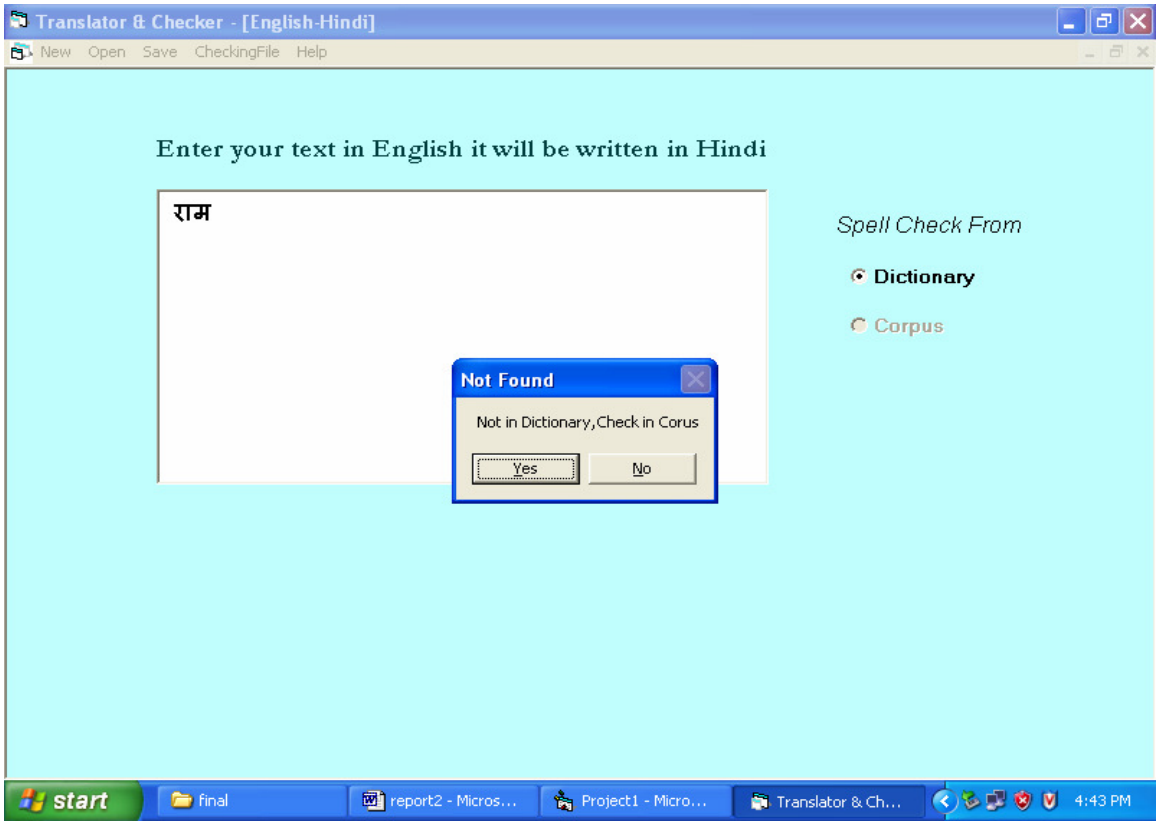
```
Private Sub cmdreplace_Click()

If cmbdata.ListIndex >= 0 Then
   txtinput.Text = Mid(txtinput.Text, 1, lwordstart - 1)
   txtinput.Text = txtinput.Text + cmbdata.List(cmbdata.ListIndex)
   Frame1.Visible = False
Else
   MsgBox "No word selected for replacement", , "No Selection"
End If

End Sub
```

4.4 Test Corpora

Each time when the word is not present in the dictionary, it ask for a option of searching in the corpus (corpus should be already added to the project)



Code snippet if word is not there in the dictionary

```
If found = False Then
    vbans = MsgBox("Not in Dictionary,Check in Corus", vbYesNo, "Not Found")
    If (vbans = vbYes) Then
      bcorpus.Value = True
      wrd = ""
       If corpusfile = "" Then
          MsgBox "No file specified for checking"
       Else
```

```
            Set tstream = fsys.OpenTextFile(corpusfile, ForReading, False,
True)
            found = False
            Do While ((tstream.AtEndOfStream = False) And (found =
False))
               ch = tstream.Read(1)
               If ((AscW(ch) > 2304) And (AscW(ch) < 2416)) Then
                  wrd = wrd + ch
               Else
                  If (lastword = wrd) Then found = True
                  wrd = ""
               End If
               'If tstream.AtEndOfLine = True Then wrd = ""
            Loop
            tstream.Close

            If found = True Then
               vbans = MsgBox("You want to add to Dictionary", vbYesNo,
"Found")
               If (vbans = vbYes) Then
                  Set tstream = fsys.OpenTextFile(dictfile, ForAppending,
False, True)
                  tstream.WriteLine (txtword.Text)
                  cmbdata.AddItem txtword.Text
                  tstream.Close
               End If
            Else
               MsgBox "Not in Corus ", , "Not found"
            End If
         End If ' end of corus file name is there
         End If 'end of check in corus ' not check in corus
```

```
End If 'word not found
```

4.5 How it works

Each time a character is pressed, the keypress event of textbox is called which has the ASCII value of the character pressed. Depending on the character pressed the respective action is taken and the corresponding character of Hindi (Unicode Devanagari) is displayed instead of original pressed character.

Code snippet

```
Private Sub txtinput_KeyPress(KeyAscii As MSForms.ReturnInteger)
Dim pascii As Integer
Dim i As Integer
Dim flag As Boolean
length = Len(txtinput.Text)


If om = True Then
   If KeyAscii = Asc("M") Then
      txtinput.Text = txtinput.Text + ChrW(2384)
      KeyAscii = 0
   Else
      om = False
   End If
End If

If tilt = True Then
   If KeyAscii = Asc("N") Then
```

```
        txtinput.Text = txtinput.Text + ChrW(2329)
        KeyAscii = 0
    ElseIf KeyAscii = Asc("n") Then
        txtinput.Text = txtinput.Text + ChrW(2334)
        KeyAscii = 0
    Else
        tilt = False
    End If
End If

If ((sym2 = "") And (sym1 = "R") And (KeyAscii <> 82)) Then
    sym1 = ""
    sign = False
End If

If ((sym2 = "") And (sym1 = "L") And (KeyAscii <> 76)) Then
    sym1 = ""
    sign = False
End If

Select Case (KeyAscii)
Case 32
    wlength = 0 'space entered
  Call check
Case 44 ' , for puranviram
    txtinput.Text = txtinput.Text + ChrW(2404)
Case 46 ' . for doubleviram
    txtinput.Text = txtinput.Text + ChrW(2405)
Case 79 ' O for om
    om = True
Case 126 ' ~ N,n
```

```
    tilt = True
Case Asc("R")
    If sym1 = "" Then
        If length = 0 Then
            sign = False
        Else
            If (vayanjan(AscW(Mid(txtinput.Text, length - 1, 1))) = True) Then sign =
True
        End If
        sym1 = "R"
    Else
        sym2 = "R"
    End If

Case Asc("L")
    If sym1 = "" Then
        If length = 0 Then
            sign = False
        Else
            If (vayanjan(AscW(Mid(txtinput.Text, length - 1, 1))) = True) Then sign =
True
        End If
        sym1 = "L"
    Else
        sym2 = "L"
    End If

Case Asc("i")
    If ((sym1 = sym2) And (sym1 <> "")) Then
        If ((sym1 = "R") And (sign = True)) Then
            txtinput.Text = Mid(txtinput.Text, 1, length - 1)
```

```
            txtinput.Text = txtinput.Text + ChrW(2371)
            KeyAscii = 0
            sign = False
            sym1 = ""
            sym2 = ""
        ElseIf ((sym1 = "R") And (sign = False)) Then
            txtinput.Text = txtinput.Text + ChrW(2315)
            KeyAscii = 0
            sym1 = ""
            sym2 = ""
        ElseIf ((sym1 = "L") And (sign = True)) Then
            txtinput.Text = Mid(txtinput.Text, 1, length - 5)
            txtinput.Text = txtinput.Text + ChrW(2402)
            KeyAscii = 0
            sign = False
            sym1 = ""
            sym2 = ""
        ElseIf ((sym1 = "L") And (sign = False)) Then
            txtinput.Text = Mid(txtinput.Text, 1, length - 4)
            txtinput.Text = txtinput.Text + ChrW(2316)
            KeyAscii = 0
            sym1 = ""
            sym2 = ""
        End If
    End If

Case Asc("I")
    If ((sym1 = sym2) And (sym1 <> "")) Then
        If ((sym1 = "R") And (sign = True)) Then
            txtinput.Text = Mid(txtinput.Text, 1, length - 1)
            txtinput.Text = txtinput.Text + ChrW(2372)
```

```
            KeyAscii = 0
            sign = False
            sym1 = ""
            sym2 = ""
        ElseIf ((sym1 = "R") And (sign = False)) Then
            txtinput.Text = txtinput.Text + ChrW(2400)
            KeyAscii = 0
            sym1 = ""
            sym2 = ""
        ElseIf ((sym1 = "L") And (sign = True)) Then
            txtinput.Text = Mid(txtinput.Text, 1, length - 5)
            txtinput.Text = txtinput.Text + ChrW(2403)
            KeyAscii = 0
            sign = False
            sym1 = ""
            sym2 = ""
        ElseIf ((sym1 = "L") And (sign = False)) Then
            txtinput.Text = Mid(txtinput.Text, 1, length - 4)
            txtinput.Text = txtinput.Text + ChrW(2401)
            KeyAscii = 0
            sym1 = ""
            sym2 = ""
        End If
    End If

Case 72  ' H FOR :
    txtinput.Text = txtinput.Text + ChrW(matra_code(10))
Case 77    ' M for .
    txtinput.Text = txtinput.Text + ChrW(matra_code(9))
Case 78 '.N for chandrabindu
  If length > 0 Then
```

```
   If (AscW(Mid(txtinput.Text, length, 1)) = 2405) Then
     txtinput.Text = Mid(txtinput.Text, 1, length - 1)
     txtinput.Text = txtinput.Text + ChrW(2305)
     KeyAscii = 0
   End If
   End If

Case Asc("D") '.D for dha
   If length > 0 Then
    If (AscW(Mid(txtinput.Text, length, 1)) = 2405) Then
     txtinput.Text = Mid(txtinput.Text, 1, length - 1)
     txtinput.Text = txtinput.Text + ChrW(2396)
     KeyAscii = 0
   End If
   End If

Case Asc("c") '.c  for ardhachandra
   If length > 0 Then
    If (AscW(Mid(txtinput.Text, length, 1)) = 2405) Then
     txtinput.Text = Mid(txtinput.Text, 1, length - 1)
     txtinput.Text = txtinput.Text + ChrW(2373)
     KeyAscii = 0
   End If
   End If

Case Asc("a") '.a  for avagraha
   If length > 0 Then
    If (AscW(Mid(txtinput.Text, length, 1)) = 2405) Then
     txtinput.Text = Mid(txtinput.Text, 1, length - 1)
     txtinput.Text = txtinput.Text + ChrW(2365)
     KeyAscii = 0
```

```
 End If
End If

Case Asc("h") '.Dh for dhhakan
 If length > 0 Then
  If (AscW(Mid(txtinput.Text, length, 1)) = 2396) Then
    txtinput.Text = Mid(txtinput.Text, 1, length - 1)
    txtinput.Text = txtinput.Text + ChrW(2397)
    KeyAscii = 0
   ElseIf (AscW(Mid(txtinput.Text, length, 1)) = 2405) Then
    txtinput.Text = Mid(txtinput.Text, 1, length - 1)
    txtinput.Text = txtinput.Text + ChrW(2381)
    KeyAscii = 0
 End If
End If
End Select

If number(KeyAscii) = True Then
   i = KeyAscii - 48
   txtinput.Text = txtinput.Text + ChrW(2406 + i)
End If

If vowel(KeyAscii) = True Then
 wlength = 0
 If length = 0 Then
   txtinput.Text = txtinput.Text + ChrW(code(KeyAscii - 97))
 ElseIf (Mid(txtinput.Text, length, 1)) = " " Then
    txtinput.Text = txtinput.Text + ChrW(code(KeyAscii - 97))
 Else
  Select Case (KeyAscii)
   Case 97 'a
```

```
    If (AscW(Mid(txtinput.Text, length, 1)) = code(50)) Then
        txtinput.Text = Mid(txtinput.Text, 1, length - 1)
    ElseIf (vayanjan(AscW(Mid(txtinput.Text, length, 1))) = True) Then
        txtinput.Text = txtinput.Text + ChrW(matra_code(0))
    ElseIf (AscW(Mid(txtinput.Text, length, 1)) = code(0)) Then 'aa
       txtinput.Text = Mid(txtinput.Text, 1, length - 1)
       txtinput.Text = txtinput.Text + ChrW(code(26))
    End If
Case 101 'e
     If (AscW(Mid(txtinput.Text, length, 1)) = code(50)) Then
       txtinput.Text = Mid(txtinput.Text, 1, length - 1)
       txtinput.Text = txtinput.Text + ChrW(matra_code(5))
    End If
Case 105 'i
    If (AscW(Mid(txtinput.Text, length, 1)) = code(50)) Then 'ki
       txtinput.Text = Mid(txtinput.Text, 1, length - 1)
       txtinput.Text = txtinput.Text + ChrW(matra_code(1))
    ElseIf (AscW(Mid(txtinput.Text, length, 1)) = matra_code(1)) Then 'ii
       txtinput.Text = Mid(txtinput.Text, 1, length - 1)
       txtinput.Text = txtinput.Text + ChrW(matra_code(2))
    ElseIf (AscW(Mid(txtinput.Text, length, 1)) = code(0)) Then 'ai
        txtinput.Text = Mid(txtinput.Text, 1, length - 1)
        txtinput.Text = txtinput.Text + ChrW(2320)
    ElseIf (vayanjan(AscW(Mid(txtinput.Text, length, 1))) = True) Then 'kai
              txtinput.Text = txtinput.Text + ChrW(matra_code(6))
    End If
Case 111 'o
    If (AscW(Mid(txtinput.Text, length, 1)) = code(50)) Then
       txtinput.Text = Mid(txtinput.Text, 1, length - 1)
       txtinput.Text = txtinput.Text + ChrW(matra_code(7))
    End If
```

```
    Case 117 'u
        If (AscW(Mid(txtinput.Text, length, 1)) = code(50)) Then 'ku
          txtinput.Text = Mid(txtinput.Text, 1, length - 1)
          txtinput.Text = txtinput.Text + ChrW(matra_code(3))
        ElseIf (AscW(Mid(txtinput.Text, length, 1)) = matra_code(3)) Then 'kuu
           txtinput.Text = Mid(txtinput.Text, 1, length - 1)
           txtinput.Text = txtinput.Text + ChrW(matra_code(4))
        ElseIf (AscW(Mid(txtinput.Text, length, 1)) = code(0)) Then 'au
           txtinput.Text = Mid(txtinput.Text, 1, length - 1)
           txtinput.Text = txtinput.Text + ChrW(2324)
        ElseIf (vayanjan(AscW(Mid(txtinput.Text, length, 1))) = True) Then 'kau
                 txtinput.Text = txtinput.Text + ChrW(matra_code(8))
        ElseIf (AscW(Mid(txtinput.Text, length, 1)) = code(20)) Then 'uu
          txtinput.Text = Mid(txtinput.Text, 1, length - 1)
          txtinput.Text = txtinput.Text + ChrW(2314)


        End If
    End Select
  End If
End If

 If (cap_vowel(KeyAscii) = True) Then
  wlength = 0
  If (length = 0) Then
     txtinput.Text = txtinput.Text + ChrW(code(KeyAscii - 39))
  ElseIf (Mid(txtinput.Text, length, 1)) = " " Then
    txtinput.Text = txtinput.Text + ChrW(code(KeyAscii - 39))
  Else
    Select Case (KeyAscii)
    Case 65 'A
      If (AscW(Mid(txtinput.Text, length, 1)) = code(50)) Then
```

```
            txtinput.Text = Mid(txtinput.Text, 1, length - 1)
            txtinput.Text = txtinput.Text + ChrW(matra_code(0))
        Else
            txtinput.Text = txtinput.Text + ChrW(code(KeyAscii - 39))
        End If

      Case Asc("I")
       If (AscW(Mid(txtinput.Text, length, 1)) = code(50)) Then
            txtinput.Text = Mid(txtinput.Text, 1, length - 1)
            txtinput.Text = txtinput.Text + ChrW(matra_code(2))

        End If

      Case Asc("U")
        If (AscW(Mid(txtinput.Text, length, 1)) = code(50)) Then
            txtinput.Text = Mid(txtinput.Text, 1, length - 1)
            txtinput.Text = txtinput.Text + ChrW(matra_code(4))

        End If
      End Select
    End If
End If

 If (cap_consonant(KeyAscii) = True) Then
      txtinput.Text = txtinput.Text + ChrW(code(KeyAscii - 39))
      txtinput.Text = txtinput.Text + ChrW(code(50)) ' adding halant
End If

If (consonant(KeyAscii) = True) Then
  wlength = wlength + 1
  If (wlength >= 2) Then
```

```
      If ((KeyAscii = Asc("h")) And (AscW(Mid(txtinput.Text, length - 1, 1)) =
code(Asc("T") - 39))) Then
         txtinput.Text = Mid(txtinput.Text, 1, length - 2)
         txtinput.Text = txtinput.Text + ChrW(2336) + ChrW(code(50))
      ElseIf ((KeyAscii = Asc("h")) And (AscW(Mid(txtinput.Text, length - 1, 1))
= code(Asc("D") - 39))) Then
          txtinput.Text = Mid(txtinput.Text, 1, length - 2)
         txtinput.Text = txtinput.Text + ChrW(2338) + ChrW(code(50))
      ElseIf (KeyAscii = Asc("h")) Then
         pascii = AscW(Mid(txtinput.Text, length - 1, 1))
         i = 0
         flag = True
         Do While i <= 25 And flag = True
          If code(i) = pascii Then
            flag = False
            pascii = i + 97
          Else
            i = i + 1
          End If
         Loop
         txtinput.Text = Mid(txtinput.Text, 1, length - 2)
         txtinput.Text = txtinput.Text + ChrW(code(pascii - 42)) + ChrW(code(50))
      Else
          txtinput.Text = txtinput.Text + ChrW(code(KeyAscii - 97)) +
ChrW(code(50))
      End If
  Else
   txtinput.Text = txtinput.Text + ChrW(code(KeyAscii - 97)) + ChrW(code(50))
  End If
End If
If KeyAscii <> 32 Then KeyAscii = 0
```

End Sub

4.6 Result Analysis

The editor is capable of writing all valid words of hindi. The spell checker is giving a correct result on a toy dictionary of 500 words and a corpus of 10,788 words.

Chapter 5 CONCLUSION

Limitations of the system

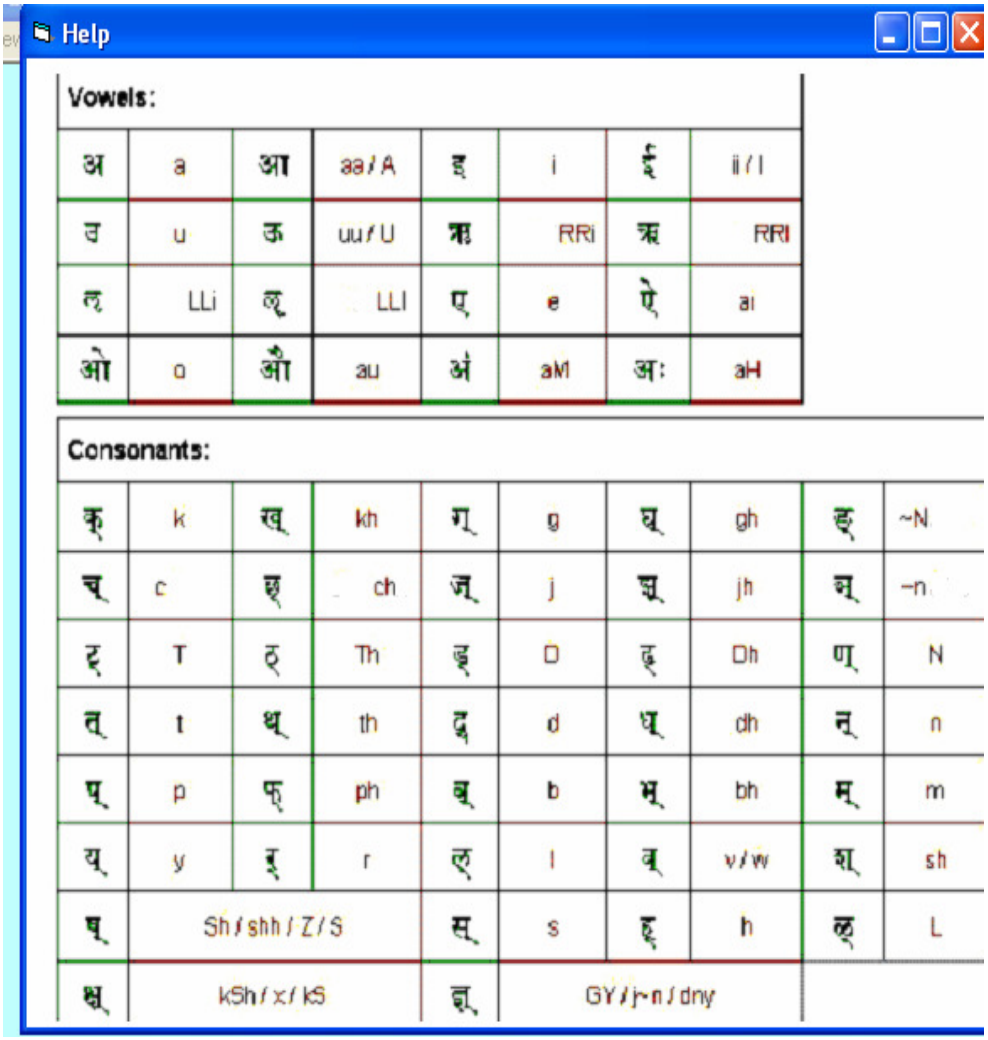
- Cut , copy and paste facility is not available with mouse click.
- Text can only be appended at the end.
- Currently it is a VB standalone system.
- User should be familiar with phonetics of Hindi.
- Dictionary should have words written in separate lines.

Further Release

- It will be a web-application using a server language like ASP/JSP which will allow it to be used on internet
- Mouse could be used more properly.
- Data could be written at any place in the file.
- The size of dictionary will be increased.
- More corpus will be added to it.

APPENDICES

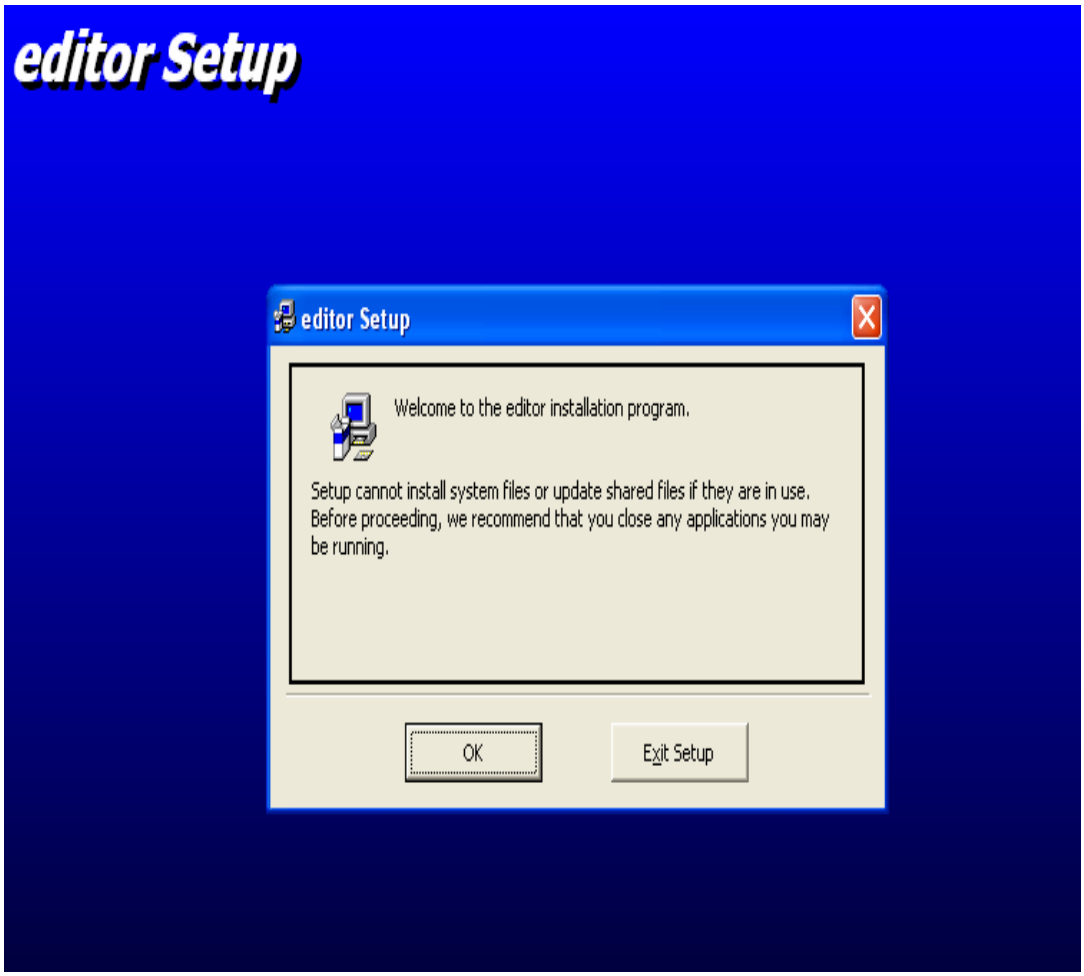
Encoding scheme for the project



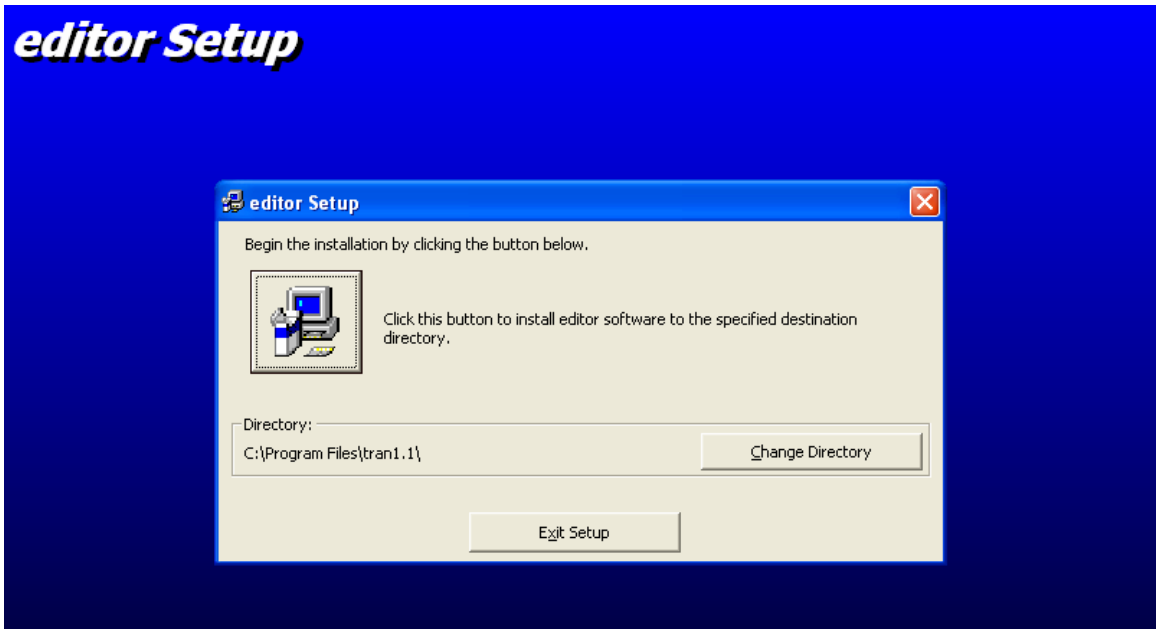
Installation Steps

Double Click on setup icon

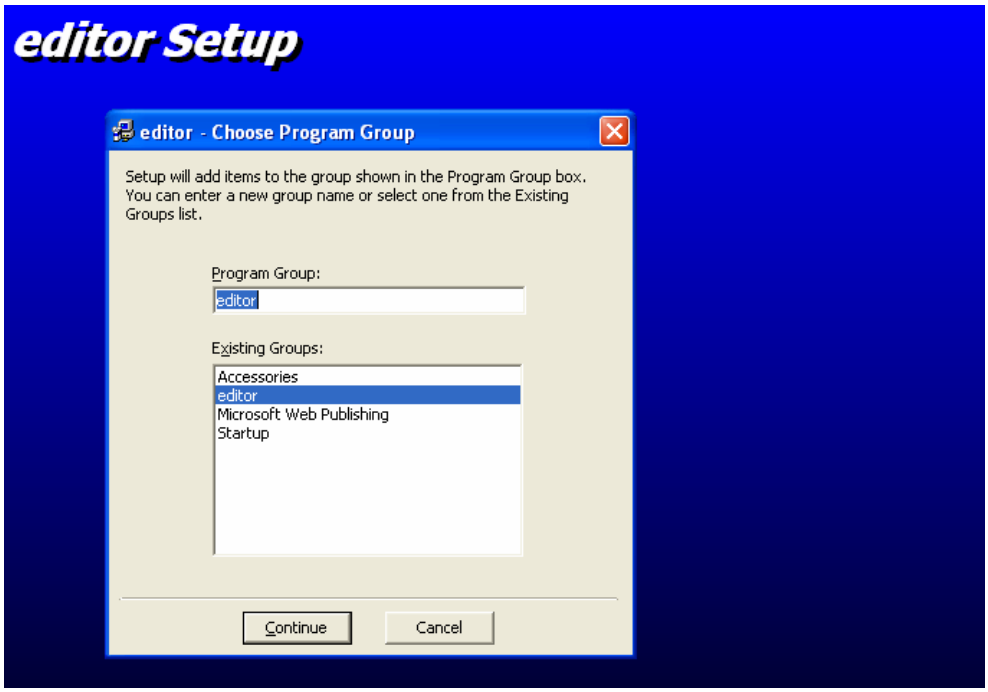
It will ask to close other applications running. Then press OK



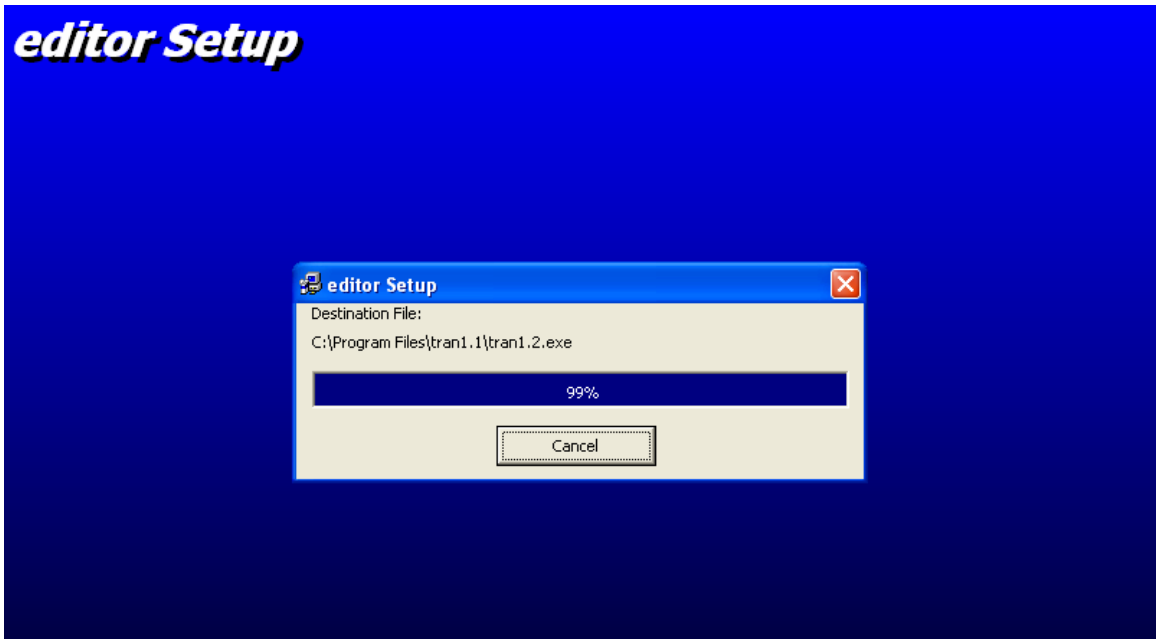
On the next screen give the directory where it has to be installed and click on the button.



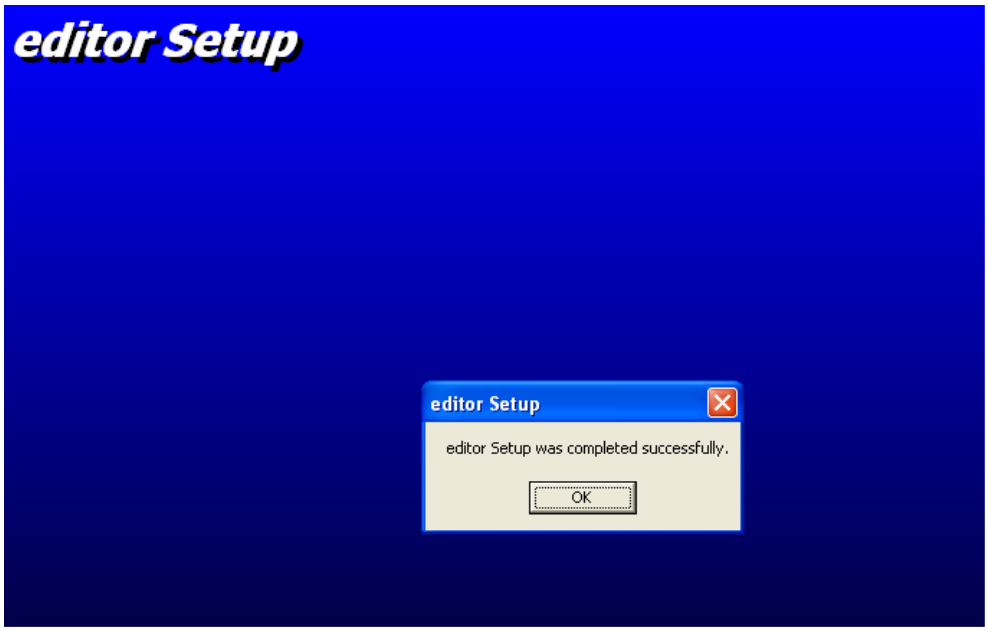
It will ask the name will you like to display in your start menu.



Setup will start and the files will be copied on the specified directory



Finally it will give a message if it has been installed properly.



Now you can find the editor installed in your start ->program files.

The main screen of the project.


Translator & Checker
New Open Save Lexical_Resources Help
Welcome
To the
English to Hindi Editor with Spell Checker
It has been developed as a supplement to a dissertation submitted by Ritu Aggrawal, in partial fulfillment of the requirements for the award of the degree of Master of Philosophy. This work has been carried out under the supervision of Dr. Grish Nath Jha, special centre for Sanskrit studies, J.N.U, New Delhi-110067
start
package1
module-report - Micro...
Translator & Checker
2:25 AM

BIBLIOGRAPHY

1. CSI communication november2005
2. http://bhashaindia.com
3. vishvabharat , jaunary2002
4. http://acharya.iitm.ac.in/multi_sys/exist_codes.php
5. www.hindiwriter.org.htm
6. http://www.techtree.com/India/News/Govt_Launches_Hindi_Software
7. http://www.baraha.com
8. http://www.microsoft.com/india/officehindi/
9. http://www.microsoft.com/india/officehindi/office2003.aspx
10. Workshop on Hindi Support and compatibility for Computers from 31st March to 1st April, 2005, JNU organized by Dr. G.N Jha.
11. Dr. V Sankaranarayanan , "IT in Indian Languages" ,November2005
12. Preamble "Linking people through Indian language technologies" Indian language technology products exposition and release , souvenir dated 15 April 2005 published by the government of India.
13. Dr.Om Vikas , "collaborative Development of Indian language technologies"
14. Dr. Grish Nath Jha , "Language Technology in India : Survey" , November 2005
15. Ananthakrishnan R.,Kavitha M, Jayprasad J Hegde, Chandra Shekhar, Ritesh Shah, Sawani Bade and Sasikumar M , "Matra : A practical approach to fully automatic indicative English-Hindi machine translation" , first national symposium on modeling and shallow parsing of Indian languages from 2-4 April 2006.