Abdel-Malek Boualem, Stéphane Harié and Jean Véronis

Laboratoire Parole et Langage
CNRS & Université de Provence
29, Avenue Robert Schuman, 13621 Aix-en-Provence Cedex-1, France
e-mail: multext@lpl.univ-aix.fr

MtScript.1.1 is freely available via WWW. It can be downloaded at the URL:
http://www.lpl.univ-aix.fr/projects/multext/MtScript/

Abstract

This paper describes the multilingual text editor **MtScript** developed in the framework of the MULTEXT project. **MtScript** enables the use of many different writing systems in the same document (Latin, Arabic, Cyrillic, Hebrew, Chinese, Japanese, Korean, etc.). Editing functions enable the insertion or deletion of text zones even if they have opposite writing directions. In addition, the languages in the text can be marked, customized input rules can be associated with each language and different kinds of character coding systems (one byte or several bytes) can be combined. **MtScript** is based on a portable environment (UNIX, X-Window, C, Tcl/Tk).

Keywords

Multilingual text, document, character, input, coding standards, editing, textual data interchange.

Résumé

Cet article présente 1'éditeur de textes multilingues **MtScript** développé dans le contexte du projet MULTEXT. **MtScript** permet de mixer nombreux types d'écritures dans un même document (latin, arabe, cyrillique, grec, hébreu, chinois, japonais, coréen, etc.). Ses fonctions d'édition permettent d'insérer ou de supprimer des zones de texte même en écritures en sens opposés. De plus, **MtScript** permet de marquer les langues utilisées dans un texte multilingue et de leur associer des règles de saisie au clavier, et de traiter différents types de codage des caractères (sur un ou plusieurs octets). Enfin, **MtScript** a été développé dans un environnement portable (UNIX, X-Window, C, Tcl/Tk) et est basé sur les standards de codage internationaux.

Mots-clés

Texte multilingue, document, caractère, saisie, norme de codage, édition, échange de données textuelles.

1. Introduction

In a previous paper [BOUA95a] we outlined difficulties associated with the design of multilingual text editors. We mentioned that if solutions for European languages were to be derived, the processing of other language is still at a conceptual stage. We presented the prototype of the "TE" multilingual text editor [BOUA90] which we integrated to a machine translation system from French to Arabic [BOUA93]. In doing this the editor showed-up weaknesses in the coding of characters and documents, incompatibilities in exchanging textual data and problems concerning non portable environments. This paper presents the **MtScript** multilingual text editor developed within the context of the MULTEXT[1] project. **MtScript** allows numerous types of writing to be mixed within the same document (see figure 1).

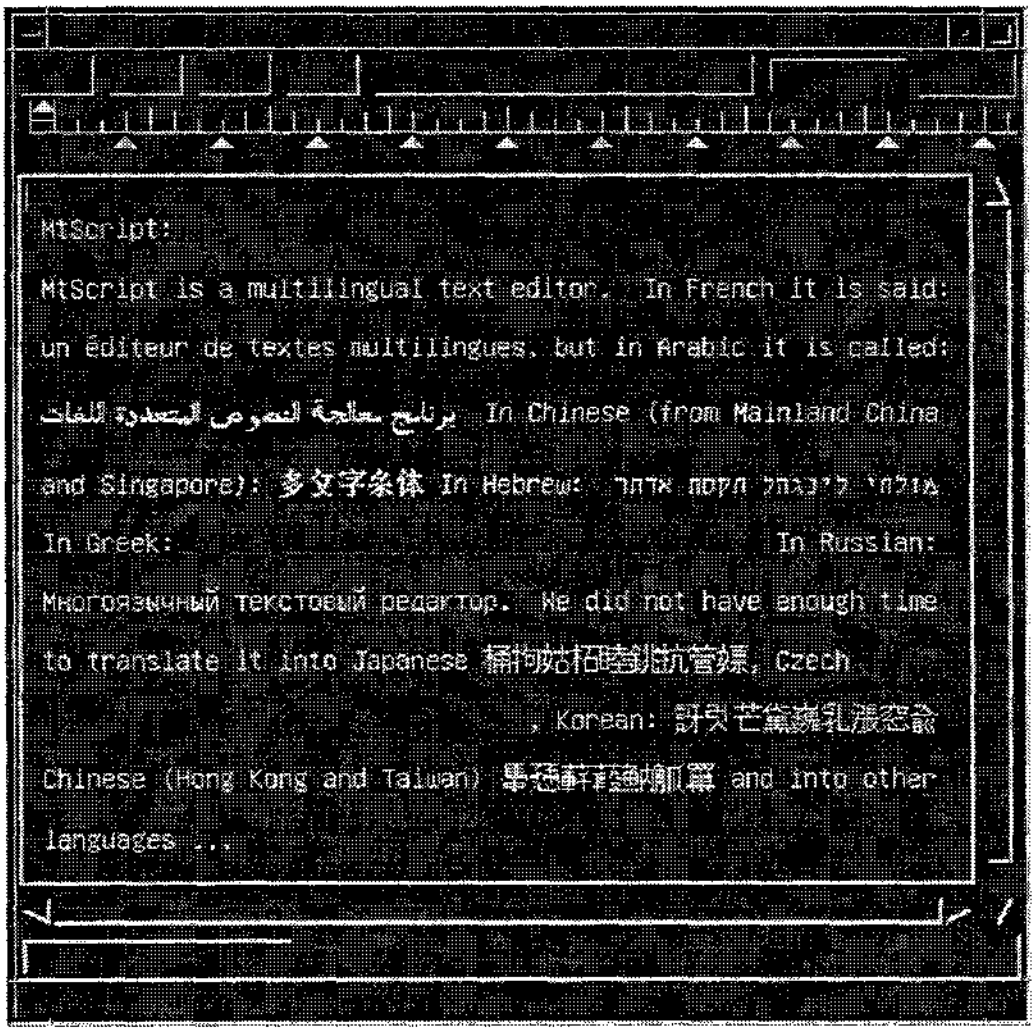

Figure 1. A view of MtScript editor

[1] MULTEXT is the generic name of a set of projects coordinated by the CNRS "Parole et Langage" Laboratory: LRE-MULTEXT *(Linguistic Research and Engineering* Program), MULTEXT-EAST *(Copernicus* Program), MULTEXT-CATALOC (Program of *Langues Régionales et Minoritaires* de la DGXXII), ALAF Research Shared Action (Alignment of African and French Languages, AUPELF•UREF). MULTEXT aims to build standard methods for linguistic data representation and to develop language processing tools. Actually MULTEXT project handles about fifteen languages.

The editing functions of **MtScript** permit insertion or deletion of text zones, even in the case of two texts written in opposite directions. **MtScript** also allows to identify the languages used within a given multilingual text and to associate them with keyboard specifications and with writing rules. It also recognizes different types of character codes (one or several bytes). Finally **MtScript** was developed in a portable environment (UNIX, X-Window, C, Tcl/Tk) and it is based on international coding standards.

2. Conceptual difficulties of multilingual editing tools

More and more applications now require multilingual text editing tools e.g. word processing, database creation and management and desktop publishing. In the area of automatic or machine aided translation, multilingual text editors are a basic tool for pre-editing source text and post-editing target text [BENT91]. Another new area where multilingual text editors could be of great use is that of internationalization and localization of software and associated documentation. Internationalization is a process for producing applications which can run in a multi-cultural environment. Localization of applications involves translation and adaptation of user interfaces and documentation. These areas were born as a direct effect of the emergence of new technology and the globalization of the Information Technology market. Many organizations and projects work to one extent or another within these areas (LRE Glossasoft project, CEC, CEN, Esprit, Eureka, Internet, ISA, Linux International, Unicode, TEI, etc.).

The processing of languages not based on the Roman alphabet poses a number of difficulties. For example Arabic is written from right to left, Chinese contains thousands of ideograms which obstructs a one byte coding. In Thai and certain Indian languages the sequence of characters in a word does not correspond to its phonetic equivalent, one character may even be drawn encircling others. In Korean, characters are fused to make syllables. Multilingual text editing implementational difficulties occur on different levels: input, coding, editing, printing and data exchange (see figure 2).

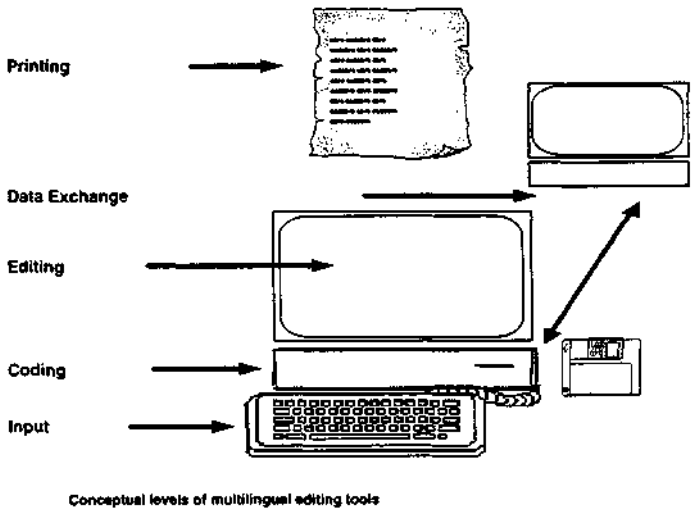

Figure 2.

2.1. Input

Though many keyboards only represent ASCII graphic characters (or ISO 646), certain localized (adapted) keyboards show some keys for special or accented characters. For example French keyboards generally feature keys corresponding to "à ç é è ù" accented characters, but characters which contain circumflexes or dieresis "ê ï" are formed by two successive key presses. In addition there is generally no single key on a French keyboard which allows production of characters which exist in other European languages such as "ñ" or "ò". In a broad multilingual context one could scarcely begin to imagine a keyboard which contains all possible characters. The inclusion of languages as Chinese (with more than 6000 ideograms) or Arabic (approximately 4 sets of 28 letters and 10 vowels) requires the definition of specific keyboard input programs.

Solutions proposed by the computer manufacturers are very heterogenous. Theoretically there exists a standard input method for keyboards with 48 keys (ISO/IEC 9995-3), at least for the Roman alphabet, but it is scarcely used. A number of input methods for the ISO 10646 characters was recently proposed [LABO95] using hexadecimal codes or composition. But these input methods always require the user to know and memorize a huge number of codes. Therefore it is necessary to develop more intuitive keyboard methods and if possible reduce the number of key presses by the user.

2.2. Coding

2.2.1. Character coding

Computer manufacturers and software developers use numerous specific and non compatible character codes (*MS-Windows character set for Western Europe MS CP1252, DEC Multinational character set, International IBM PC character set IBM CP850, Macintosh Extended Roman character set, Hewlett-Packard ROMAN8,* etc.). However successive versions of character coding norms have been standardized to an international level and are already used in some environments. In particular the ISO 8859 code proposes a standard character set for the Roman, Cyrillic, Greek, Arabic and Hebrew alphabets. More recently (1993) the ISO 10646 (*Universal multiple-octet coded character set or UCS*) proposed a universal character set including all the character sets of the ISO 8859 as well as those for Chinese, Korean, Japanese, International Phonetic Alphabet, etc. In its present form (ISO 10646-1) the UCS uses a 16-bit code (UNICODE) and will be extended to a 32-bit one in future editions, thus permitting an unlimited coding of characters [JAMG95]. However, existing environments are not yet ready to implement character sets on multiple-octet code, even though the situation is rapidly improving (e.g. Windows-NT, AT&T Bell Plan 9 and Apple QuickDraw GX).

2.2.2. Writing systems coding

In a multilingual text it is preferable not only to code individual characters but also writing systems (generalized notion of languages or scripts). In the case of a one octet based coding (for example the ISO 8859-1 character sets) it's necessary to mark the change from one set to another (e.g. changing from Greek to Cyrillic). This can be done using a code such as that proposed in the ISO 2022 which includes escape sequences (<SI> (shift in) and <SO> (shift out)) allowing the transition between the "main" and the "complementary" sets. However these techniques are limited and many difficulties can arise especially when mixing in the same document of characters coded in one-byte (e.g. ISO 8859) and two-byte code (e.g. gb-2312-80 or big5-0 for Chinese, jisx0208-1983-0 for Japanese or ksc5601-1987-0 for Korean).

The UCS code solves one part of the problem by combining all these character sets into a single set, since it is no longer necessary to implement the means for switching between character sets. Still the problem is not totally resolved because the UCS does not include explicit coding of each writing system, which is necessary, particularly in determining the writing direction.

2.2.3. Language coding

Linguistic processing of a multilingual text (segmentation, morphological and lexical analysis, etc.) requires the identification of the languages therein. Recognizing the character set or the writing system does not suffice to identify the language in which a portion of text is written: a document encoded in ISO 8059-1 could equally well be written in French, English, Spanish or even a combination of these languages.

Norms for coding the names of languages exist:

- ISO 639-1988: 2 alphabetic letters code for about 140 languages (e.g. "en" for English, "fr" for French, etc.).
- ISO 639-2: 3 alphabetic letters code, alpha-3, is currently in development (e.g. "eng" for English, "fra" for French, etc.).

However, in the internal code of a document, these codes cannot be used such as they are. At this time there is no established standard method for escape sequences which would permit the representation of the change from one language to another. Thought it has been proposed that one use the ISO/IEC 6429 set of control sequence codes with a numeric conversion of the above alphabetic codes [LANG93]. The language markup is also currently being defined in HTML language used by the World Wide Web [YERG95].

2.3. Editing

The majority of languages are written horizontally from left to right. Certain languages as Arabic or Hebrew are written from right to left. Other languages as Chinese or Japanese can even be written from top to bottom (ancient text). The co-existence of languages in the same document, and particularly on the same line of the text, poses huge problems when inserting or deleting text zones. The example of the figure 3 shows that it is necessary to rearrange words to maintain the semantic coherence of a sentence.

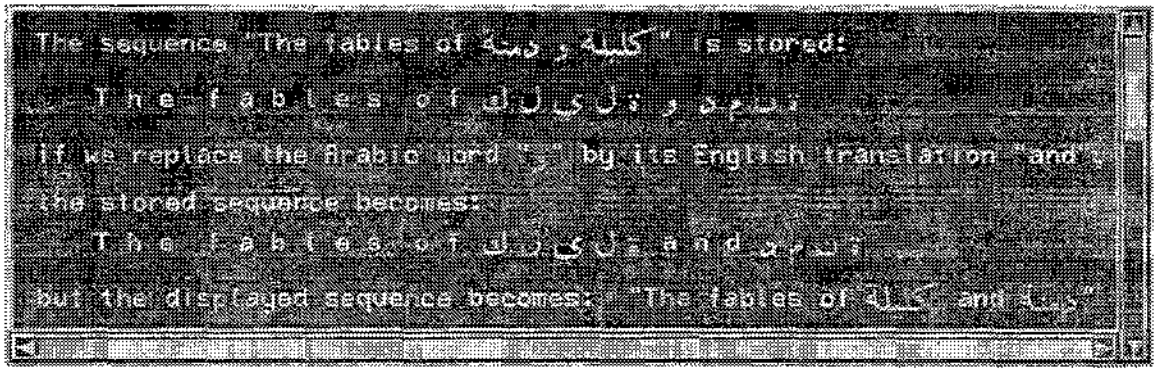

Figure 3. Editing functions in a multilingual text

2.4. Printing

New printing techniques including bitmap or postscript representation are not completely expanded to non Roman characters. Even though significant works are arising such as C.Bigelow and K.Holmes [CBKH95] in designing a UNICODE font *Unicode Lucida San* for editing and printing multilingual electronic documents.

2.5. Data exchange

With the Internet boom, the electronic transfer of multilingual documents has become more and more commonplace. Until recently, only standard invariant characters of the ISO 646-IRV (Ascii) could allow a non corrupted electronical text exchange. Thus it was necessary to use coding utilities such as *Uuencode* and *Binhex*. However the situation is improving: standards have been adopted on Internet which allow the transfer of 8-bit characters without corruption in the TCP/IP protocol (for example applications such *Telnet* and *Ftp* are "8-bit clean"). In addition the MIME norm *(Multi-purpose Internet Mail Extension: RFC-1521 and RFC-1522)* allows uninterrupted data transfer in any circumstance by compressing and decompressing the files. But this standard is not yet universal and some transfer problems persist. In saying this, one must point out that the guarantee for octet transferral without corruption (in particular without the loss of the eighth bit) does not extend to the transfer of multilingual data. It is necessary that both

parties involved in the transfer, the sender and the receiver, have the same systems of encoding characters, languages and writing systems.

3. Existing tools

The ground work for producing multilingual text editors has often been carried out under the form of independent experimental studies often leading to incompatible products which are difficult to use and do not conform to coding norms. In addition to this, the proposed solutions often concern only languages using the Roman alphabet and cannot be adapted to other families of languages.

Amongst the multilingual text editing tools we mention the "Universal Word" developed by Wysiwyg Corporation [UNIW96] and the "Wintext" program developed by Winsoft in the Apple Macintosh environment. These allow the mixture of several languages in the same document, even those written in an opposite direction. In the PC-Windows environment we mention the Microsoft word processing software "Word" which uses the multilingual interfaces TwinLink and TwinBridge.

On workstations TEX editor and its multilingual extensions (ArabTeX, etc.) is mostly present in the scientific applications. Various TEX fonts for different languages have been designed but this editor has the disadvantage of not being Wysiwyg. OMEGA project [YHJP95] includes a number of TEX extensions to improve the multilingual text processing. It uses the ISO 10646/UNICODE standard code with conversion mechanisms of other standard codes. Powerful algorithms allow the interpretation of the composition and the transliteration of non-Latin characters (user interface), the handling of different character codes (information exchange) and the generation of correct character graphical components as ligatures (typography).

More recent works in the multilingual domain can be mentioned such as the CRL Laboratory activities [CRL96] in designing tools for various domains (multilingual machine translation, text retrieval, multilingual dictionaries, etc.), the Accent company activities [ACC96] in designing multilingual WWW browsers and the LangBox International activities [LANG96] in the UNIX system localization.

With regards to the previously mentioned works in the multilingual text editors, **MtScript** is in one hand freely distributed via Internet and in the other hand it is based on a environment (UNIX, X-Window, C, Tcl/Tk) which is parametrable, evolutive and portable (PC/Windows and Macintosh).

4. Description of the MtScript editor

4.1. Features

The **MtScript** editor was developed in the UNIX, X, C, Tcl/Tk environment, which shows the following advantages:

- Tcl: script language (the commands are interpreted interactively)
- manipulation of textual data (characters, fonts, words, etc.)
- possibility to define character "attributes"
- possibility to manage X-Window events (mouse, keyboard, etc.)
- bitmap control
- easy to use (X-Window, buttons, icons, etc.)
- portability to several other environments (Windows, etc.)

MtScript is a text editor including all the characteristics of a standard monolingual editor plus those related to multilingual texts:

- mixing of opposite writing directions on the same line of text
- recognition of the language used in a given piece of text
- insertion/deletion of characters regardless of the direction in which the text is written
- text editing functions: copy, cut, paste, etc.

MtScript is independent of any language. The languages are considered as external parameters of the program represented by writing rules files and character fonts. The expansion of the editor to include a new language simply requires the inclusion of the character fonts and the writing rules.

4.2. Internal representation

In its current version, **MtScript** handles the following character sets:

- iso8859-l, 2, 3 and 4 (Roman Alphabet)
- iso8859-5 (Cyrillic)
- iso8859-6 (Arabic)
- iso8859-7 (Greek)
- iso8859-8 (Hebrew)
- gb2312-80 and big5-0 (Chinese)
- jisx0208-1983-0 (Japanese)
- ksc5601-1987-0 (Korean)

In future versions, we hope to adopt the UCS set (ISO 10646) which includes other writing systems and a large number of characters absent from the norms presently included (for example, the conjoined symbols "œ" and "Œ" which are considered in French to be distinct from their component parts).

Labelling of characters and languages in a multilingual text is executed with reference to a "style file" associated with each multilingual text. It contains values of the character properties. These properties describe for each part of text: languages, fonts, character sets, style, tabulations, height and colour of characters. They are associated with a fixed position in the text, expressed by line numbers and character numbers. Figure 4 gives a partial internal representation of the "style file" associated to the text in figure 1. We are currently developing a HTML exchange format which uses the <LANG> tag proposed by the HTML3.0 norm.

```
{mtscript_version 1.1}
{default_style
  {-width 80}
  { -height 40}
  {-tabs {52.0 104.0 156.0 208.0 260.0 312.0 364.0 416.0 468.0 520.0 572.0 624.0 676.0 728.0 780.0}}
  {-wrap char}}
{xx
  {-foreground IndianRedS -font iso_8859_1}
  {23.023.1}}
{ar
  {-foreground PaleGreen4 -font arabic}
  {8.18.40}}
{bg
  {-foreground DeepPinkl -font iso_8859_5}
  {}}
{zh_CN
  {-foreground brown -font gb2312_1980}
  {10.17 10.27 10.3910.40}}
{zh_TW
  {-foreground MediumPurple4 -font big5_0}
  {20.3220.46}}
{cs
  {-foreground DarkGoldenrod -font iso_8859_2}
  {16.5918.34}}
{nl
  {-foreground black -font iso_8859_1}
  {}}
{en
  {-foreground black -font iso_8859_1}
  {1.1 4.41 6.35 8.1 8.40 10.17 10.27 10.39 10.40 10.41 10.64 12.11 12.51 14.1 14.32 16.31 16.51 16.59 18.34 18.44
18.6920.3220.4623.0}}
```

Figure 4. Internal representation of multilingual texts

4.3. Multilingual text input

MtScript uses input programs based on those characters which are found on almost all keyboards, i.e. those of ISO 646-IRV. There are two types of input programs:

- **Alphabetical input programs** for alphabetical languages (French, Arabic, Russian, etc).
- **Phonetic input programs** for ideogram based languages (Chinese, etc.). This second program is currently in development.

4.3.1. Alphabetical input program

The alphabetical program is a single program used for all alphabetical languages. It uses separate "writing rules" and "transliteration rules" files for each language (figure 5).

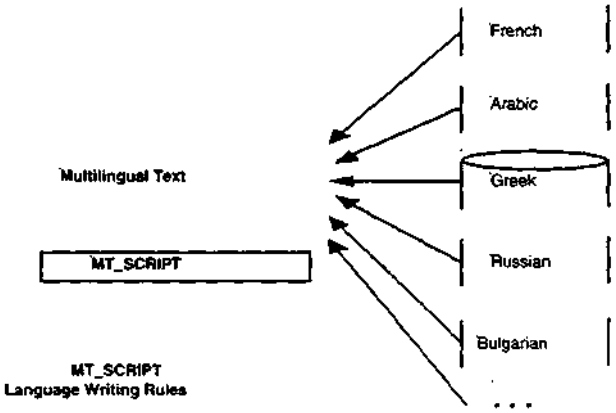

Figure 5

Each writing rules file contains:

• a classification of characters according to their behaviour e.g.

- lowercase letters which may have an accent
- lowercase letters which may not have an accent
- uppercase letters which can have an accent
- standard characters used for typing accents
- numbers
- etc.

• a set of writing rules e.g.

- French: e + ' => é ; c + , => ç; etc.
- Greek: σ (beginning and middle of a word), ς (word ending)
- German: s + s => ß; etc.

The writing rules for each language are expressed in an intuitive formalism based on finite state automata mechanism. The writing rules files are then compiled and converted into internal tables usable by the input programs.

A default set of rules are defined for each language, but they can be redefined at will by the user, according to specific needs or preferences or to suit the particular specifications of certain keyboards. The default rules are based on the following principles:

- **Characters with accents** require two key presses, according to the rules for the language. Thus, in the default rules for French, e+' gives é, but for the same combination in English, it gives e'. To produce e' in French (a combination which is far less frequent than é) one uses the escape sequence e + ESC + ' => e'

- **Non-Roman characters** are typed following, as closely as possible, the conventions of the concerned language and the transliteration norms where they exist. The transliteration tables are included in a "transliteration rules" file associated to each language (e.g. ISO 233-1984/1993 for Arabic, ISO 259-1984 for Hebrew, ISO/R 843-1968 for Greek, etc.). Thus, one types **a** for α, **b** for ß, **s** for س etc.

- **Variant forms,** such as σ and ς (Greek) ß (German) or the positional variants of Arabic letters, are generated automatically according to their context, so that the user does not have to intervene. The case of Arabic is particularly interesting [BOUA95b]: the alphabet contains 28 letters, the majority of which can be written in 4 different forms, according to their position in a word (see figure 6). **MtScript** can handle the positional variants of characters including in edition functions such insertion or deletion ones.

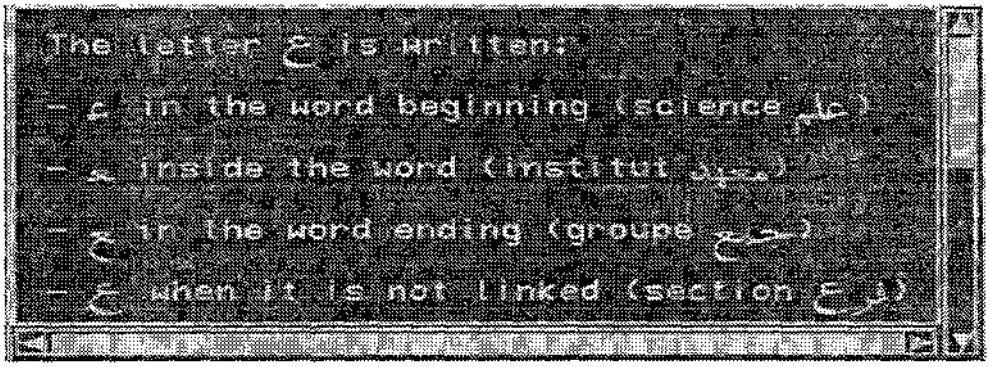

Figure 6. Positional variants of Arabic characters

4.3.2. Phonetic input program

Certain languages, such as Chinese, are based on a vast number of ideograms, each one representing a particular concept. Recently (1980's) simplified and varied versions have been adopted by the People's Republic of China, on one side, and by Taiwan and Hong Kong on the other. Several different input methods exist, such as input via character codes (e.g. **GB-2312-80** code) or the **Pinyin** method, which consists of a phonetic representation of ideograms in Roman characters (420 syllables, complemented by differing tones, up to five tones per syllable). In its next version, **MtScript** will include a method of input for Chinese ideograms based on their Pinyin phonetic transcription.

4.4. Display and retrieval

As stated earlier, the fundamental problem with displaying multilingual texts is the co-existence of opposite writing directions on the same line. Insertion and deletion of characters must take into account their writing direction according to quite complex rules. **MtScript** allows the user to define interactively a main and a secondary writing direction for a text-zone. The cursor moves only in the direction specified as the main one. When a sequence of characters is entered in a language written in the secondary direction, the cursor stays put and the characters are written in an insertion mode (see figure 7).

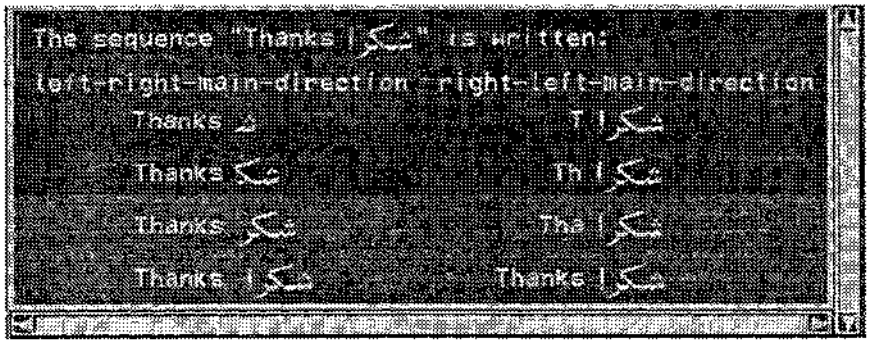

Figure 7. Opposite writing directions

Multilingual texts are stored in memory in the form of a sequence of bytes representing character codes. In order to obtain an output which is faithful to the multilingual text in memory, catering for positional variants, combinations of writing directions, etc., the output to the screen is displayed via the input programs. However the justification aspects of multilingual texts including opposite writing directions are still under development.

5. Future developments

At this point in time, a multilingual text editing prototype exists which is capable of most of the functions envisaged for the project. However, two modules are currently being developed, as was mentioned in the text: HTML data transfer and phonetic input programs for ideogram based languages. Amongst future developments (quite apart from other languages' inclusion) we envisage the inclusion of other aspects of HTML representations (headers etc.) which permit the improvement of texts and the use of **MtScript** with a WWW browser and the use of vowels in Arabic. Even though the Arabic texts one sees in newspapers and magazines do not generally contain vowels[2], these are of prime importance in Arabic processing (in particular when compiling lexicons).

[2] The Arabic vowels are represented by diacritics written above or below consonants. The human reader can generally ascertain the meaning of vowel-less words given their context, thanks to the morpho-graphematical structure of Arabic language.

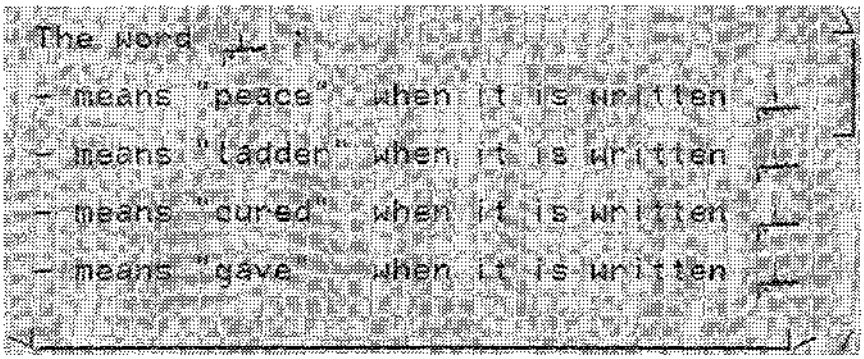

Figure 8. Arabic vowels

In the time being, the current version of **MtScript** is using a non voweled Arabic font generated from the Y.Haralambous T_EX metafont code. But we are including now a complete version of this font including most of the Arabic characters required by UNICODE with their contextual variants and vowels.

6. Conclusion

Today sees a new evolution of Information Technology, leading to a growing need for multilingual tools. The exchange of information between parties who speak different languages has become more and more frequent. The multilingual text editor is a fundamental tool of bringing about the globalization of information resources. **MtScript** is developed with a view to answering the needs for coding and processing tools required for multilingual documents in both European and non-European languages. This tool allows inputting, editing, storing and transferring multilingual texts and opens the door to many applications which require multilingual text processing such as the segmentation and morphological analysis of texts, machine aided translation, multilingual dictionaries and the localization of software and their associated documents in different languages.

Acknowledgments

This work has been financed by the European Community in the framework of the MULTEXT Project. Several persons contributed to the improvement of **MtScript**. The authors acknowledge particularly Greg Priest-Dorman for the intensive tests of **MtScript**, Emmanuel Flachaire for the compilation of the Linux (Intel) version and Nancy Ide for her help with the English documentation. We acknowledge Mark Leisher from CRL Laboratory (New Mexico State University) for his comments and his help with Arabic fonts and the anonymous colleagues for their help in reading this paper.

Bibliography

[ACC96] http://www.accentsoft.com

[BENT91] P. M. Benton, "The Multilingual Edge", *BYTE,* March 1991, 124-132.

[BOPI95] L. Bourbeau, F. Pinard, "Normalisation et internationalisation: inventaire et prospective des normes clefs pour le traitement informatique du français". *Progiciels BPI. ,* Montréal, Canada, 1995.

[BOUA90] A. M. Boualem, "The multilingual terminal", research report, INRIA Sophia Antipolis, January 1990, 1-4.

[BOUA93] A. M. Boualem, "ML-TASC: Système de traduction automatique multilingue dans un environnement à syntaxe contrôlée", SS'93, *7th annual High Performance Computing Conference,* Alberta, Canada, June 1993, 537-544

[BOUA95a] A. M. Boualem, "Multilingual text editing", *SNLP'95, The 2nd Symposium on Natural Language Processing,* NECTEC, C&C, Bangkok, August 1995, 336-342.

[BOUA95b] A. M. Boualem, "Arabic Language Processing", *SNLP'95, The 2nd Symposium on Natural Language Processing,* NECTEC, C&C, Bangkok, August 1995, 95-102.

[CBKH95] C. Bigelow, K. Holmes, "The design of a UNICODE font", version française dans le *Cahier GUTenberg* n°20, May 1995, 81-102.

[CRL96] http://crl.nmsu.edu

[LANG93] *Language Coding Using ISO/IEC 6429.* Draft circulated in January 1993 by the European Standardization Organization CEN technical committee TC304. Available electronically at:

http://www.stonehand.com/unicode/standard/tc304.html/

[IDVE95] N. Ide, J. Véronis, *The Text Encoding Initiative: Background and Context,* Kluwer Academic Publishers, Dordrecht, 1995.

[JAMG95] J. André, M. Goossens, "Codage des caractères et multi-linguisme: de l'ASCII à UNICODE et ISO/IEC-10646", *Cahier GUTenberg* n°20, May 1995, 1-54.

[LABO95] A. Labonté, "Input methods to enter characters from the repertoire of ISO/IEC 10646 with a keyboard or other input devices". *ISO/CEIJTC1/SC18/GT9 Working Draft,* February 1995.

ftp://ftp.funet.fi/pub/doc/charsets/ucs-input-methods

[LANG96] http://www.spartacus.com/langbox/

[MUL96] http://www.lpl. univ-aix.fr/projects/multext/

[UNIW96] http://www.wysiwyg.com

[YERG95] F. Yergeau, G. Nicol, G. Adams, M. Duerst, "Internationalization of the Hypertext Markup Language". *Internet Draft draft-ietf-html-i18n-02,* November 1995.

http://www.ics.uci.edu/pub/html/draft-ietf-html-il8n-02.txt

[YHJP95] Y. Haralambous, J. Plaice, "Ω, une extension de TEX incluant UNICODE et des filtres de type Lex", *Cahier GUTenberg* n°20, May 1995, 55-79.