The Worldwide Multilingual Computing (5):

4Q－5 # Multilingual Text Manipulation and Text Widget

Toshio Oya†, Tomoko Kataoka*, Kazutomo Uezono†, Tadao Tanaka‡, Hidejiro Daikokuya†, Kenji Maruyama‡, Shoichiro Yamanishi†, Yutaka Kataoka* and Hiroyoshi Ohara†

* Centre for Informatics, Waseda University † School of Science and Engineering, Waseda University
‡ Research and Development, Japan Computer Corporation

1. Multilingual Text Manipulation

Generalization of Multilingual Text Processing has been established [1, Talk 4]. It is found to divide into two categories – *language/codeset/font independent* manipulation and *dependent* text processing. The process unit for the former independent manipulation has been decided as our WC, normalized to one character [2]. Text drawing has been discovered as one of the essentials of the text manipulation, to which Talk 3 and this talk pay major attentions. Without a proper unit of a character, disallowable line separations would occur for Thai scripts. More advanced processings will be treated by TMC codepoints designed to meet various purposes with high dependencies on language/codeset and user's personal needs [1, Talk 4]. Editing functions other than drawing have been supported for TMC.

Drawings have yet to be realized. Most systems support only horizontal, left-to-right drawings: *vertical* drawings must be realized. Waseda *MLCE* (Multilingual Computing Environment) [2] has already realized the principled vertical drawings (top-to-bottom, bottom-to-top) as well as right-to-left drawings. Current networkings on computers interwoven all over the world have been making it more serious to realize a proper handling mechanism of the *two-dimensional drawings* (2DD). The mixed text of the subtexts with different directions of line progress should be displayed correctly, although in fact the problem of 2DD also becomes apparent for line feeding. Logical horizontal and vertical writings mixed together necessitates the notion of a *page*, at which the two written texts cross.

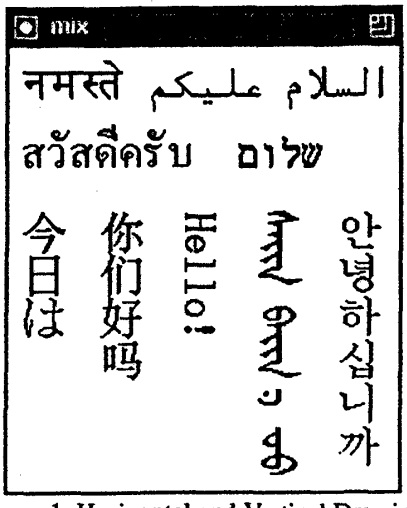

Figure 1: Horizontal and Vertical Drawings

Changeability of the certain bit(s) of WC can account for any possible case of displaying including origin of line or drawing direction; not only line feeding for multilingual texts. A set of both editing – deletion, insertion, search or line separation – and displaying functions are essential for the basic text manipulation. They, in turn, prove the validity of the WC's status as a process unit of editing and displaying.

Drawing directions are not the only problems. Current character cursor systems will be too poor to demonstrate even the appropriate position with the extent and its moving direction. Also, 'WC information window' has been supported to inform the character set name of a character and the whole window status to assist users when editing. When pointing devices send an event, a function must be prepared and ready to inform the location of, the number assigned for, a WC in memory image extracted from OM, since the location of WC, even the number of WCs, may be different from that in a display.

By the generalization of essential functions, the set of functions of editing and displaying would constitute an independent *subroutine*, able to be activated when called by applications. *Text Widget* within *Athena Widget* is a set based on such notions, but it lacks the criteria for prerequisite functions for editing and displaying all characters in the world correctly. Thus, trials have been given to replace Text Widget in Athena Widget to testify the functions. The necessity of *Filtering Functions* to be discussed below became clear in the process and they have been implemented in this version, an alternative to that in current Waseda MLCE.

2. Filtering Functions in New Widget

As noted above, Text Widget should be a set of functions to realize the basic text manipulation. In this respect, such *language dependent* operations like *spell checking* should not be supported. Considering the context of Text Widget in daily use, however, another independent *process*, a child process, had better be called for spell checking with the shell invoked. Mb/WC conversion is necessary between the Widget and a spell checker. A spell checker is highly codeset and encoding dependent. Meta Converter [3] takes these conversions.

For a spell checker to operate properly, the sender must specify the same encoding with that of the checker (receiver). Thus, for a multilingual text, characters of certain GCS(s) must be filtered through so that only they, not other characters in different GCSs, will be sent to be

checked. The distinction between 'A' in JIS X 0201 and that in ISO 646 cannot be made. For such codeset sensitive requirements, the function which returns a GCS name is prepared in *Interprocess Communication Assisting Functions* (ICAF).

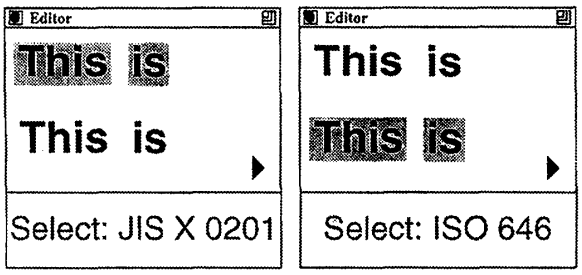

Figure 2: Filtering Characters of certain GCS

Cut & Paste brings more complicated troubles. A WC string marked by a cursor or a pointer. It must be converted into its corresponding mb string before the WC string is sent to *cut buffer* in X Server. X specification intends for *Compound Text* (CT) to be used, but ISO 2022 [4] should also be used. When the application hopes to send a string in certain GCSs, an ICAF function will realize it.

For a recipient, it does not necessarily receive GCSs which its local knows. When there are strings in the GCS/CCS which it does not know, there should be two choices to make: 1) it does not receive, or in other words, drops them; or 2) it receives all the data regardless of whatever GCS/CCS involved.

A step forward, with the filtering of GCSs established, cut & paste would operate more comfortably. For example, Widget reverses the characters of a certain codeset in the limited scope the cursor marked or of the whole in advance, before the cutting. With a function in WC Information Functions to extract the GCS name from WC bitfield, the *selecting/filtering certain GCS(s)* is realized. This function must be supported when saving files.

These filtering functions conclude the essential functions for Text Widget to cover. With these, all the WC's editing and displaying functions constitute a set of the Text Widget. Inter/Intra Communication Problem occurs with the Locale Model, while processing with the Global IOTMC Model causes no trouble [2, 5, Talk 2].

3. Implementation Remarks

New Text Widget in Athena Widget set has been provided by the set of functions generalized for text manipulation. It satisfies both *Multi-locale Model* and *Global IOTMC Model*. Of course, an interactive switching of locales is possible.

X11 provided no vertical drawing functions and no font metrics for vertical drawing. By our font format researches, essential metrics for the vertical drawing were discovered, and implemented to our X11 with new vertical drawing protocols. The vertical drawing speeds on X11 were dramatically increased.

Refer to Talk 3 on the problems of 1) resource files with the introduction of locales, and 2) cut & paste with encoding in CT. Note that CT is not proper for interprocess communication, since it depends on vendors. ISO 2022 should be used instead, and the support for CT is just for backward compatibility.

4. Summary and Further Study

An essential function set for the generalized language/codeset independent text manipulation has been realized. The WC codepoint has been defined as a process unit normalized to one character, and by each information for editing and displaying stored in its bitfield, Multilingual Widget in Athena Widget has been realized with the function set. The requirements of Communication and Processing in ML environment, that is, those of Interprocess Communication has been made definite, which has made ways to allow another process to deal with codeset dependent operations.

Satisfying the possible users' needs has led to only a partial generalization of the primary functions required for *Graphical User Interface* (GUI). Thus, researches for processings and their data based on function categorization can satisfy the structure and roles of generalized GUI with principles and essentials.

References

[1] Kataoka, T. et al., Multilingual I/O and Text Manipulation System (3): Extracting the Essential Informations from World's Writing Scripts for Designing TMC and for the Generalizing Text Manipulation, Proceedings of the 49th General Meeting of IPSJ, Vol. 3, September 1994, pp 303-304.

[2] Kataoka, Y., et al., 1995. Codeset Independent Full Multilingual Operating System: Principles, Models and Optimal Architecture, IPSJ SIG System Software & Operating System, 68-4, pp 25-32.

[3] Tanaka, T., et al., Multilingual I/O and Text Manipulation System(4): The Optimal Data Format Converter to/from MB/WC/TMC, Proceedings of the 49th General Meeting of IPSJ, Vol. 3, September 1994, pp 305-306.

[4] ISO/IEC 2022: 1986, Information processing – 7-bit and 8-bit coded character sets – Code extension techniques.

[5] Kataoka, Y., et al., Multilingual I/O and Text Manipulation System(1): The Total Design of the Generalized System based on the World's Writing Scripts and Code Sets, Proceedings of the 49th General Meeting of IPSJ, Vol. 3, September 1994, pp 299-300.