E-48

多言語処理システム Unite の設計

蒙古文字の表示サブシステム

The Multilingual Text Processing system Unite applied to Traditional Mongolian

小林 丈二† 上園 一知†
Joji Kobayashi Kazutomo Uezono

1. 背景

早稲田大学筧研究室では、長年に渡り、多言語処理システムの研究を行ってきた。そして、2002 年度から多言語処理システム Unite を一から設計することにした。設計するにあたり、はじめに決めたことは、開発言語に Java を使って、プラットフォーム依存性をおさえることと、Unicode テキストを扱えるようにすること、モンゴル語を最初の対応言語とすることである。

ISO/IEC 10646(以下 10646 と略す)の 2000 年版に蒙古文字が追加された。10646 は、かならずしも言語処理上の文字の扱いにおいて統一方針をもったものではない。しかしながら、広く受け入れられ、使われ始めている。実際、MS-Windows や Solaris、Java などの内部文字コードとして使われ、高品質のフォントも開発され提供されてきている。これらの状況から、Java ベース、10646 準拠とすることにした。

モンゴル語は、とくに 10646 の文字と、実際にモンゴル語の文章を表示する上での字形の対応が一意的になっていない、という点で、今まで培ってきた知見と技術を発揮する適例であると思われる。そこでまず、モンゴル語の処理から研究をはじめることとした。

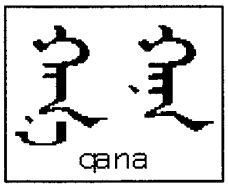
2. 蒙古文字表記の概要と Unicode における表現

2.1 蒙古文字によるモンゴル語表記

蒙古文字によるモンゴル語表記は、縦書きで、行は左から右へと書くという特徴を持つ。また英語でいえば筆記体のように隣り合う文字が連結していて、それぞれの字形が、単語中の位置により変化する（語頭形、語中形、語末形、独立形）。

†早稲田大学筧研究室

文字が単独で表示される場合には、独立形を、単語の先頭に位置すれば語頭形を、末尾に位置すれば語末形を、それ以外の位置にあれば語中形を選択するのが基本になっている。その他の語中変化形として、中間語末形がある。子音字に母音字 'a' または 'e' が続くかたちで、単語が終わっている場合に、子音字が語末形となり、'a'、'e' が  という字形をとる場合がある。この場合の子音字は、語そのものからみたら語中形であるが、字形は語末形であることから中間語末形という。'a'、'e' は、2 種類の語末形を持っているのである。右に中間語末の例を示した。左右同じ綴り(qana)であるが、左側の単語が中間語末形となっている。



2.2 Unicode による蒙古文字の表現

Unicode に登録されている蒙古文字は、母音字と子音字、それに数字や句読点だけである。語中変化による字形処理などは、実装にまかされている。字形変化については、2.1 で述べた語中変化が基本であるが、語中形だけを 1 文字として表現したいとか、語頭にある文字を語中形にするなど、基本以外の表記を可能にするための制御文字が、表 1 のように定義されている。

表 1

ニモニック	文字コード	名前
MVS	\u180e	Mongolian vowel separator
ZW Join	\u200d	zero width joiner
ZW N-Join	\u200c	zero width non-joiner

Unicode の文字コードは、\u に続けて 16 進数で表現している。

2.3.1 MVS

子音字の直後に MVS を置くことで、中間語末形を指定する。

2.3.2 ZW Join

モンゴル語専用の制御文字ではないが、蒙古文字とともに使われたときに、字形制御文字として扱われる。2 文字以上からなる単語の先頭に ZW Join を付加すると先頭の文字が語中形になる。単語末尾に ZW Join を付加した場合には、末尾の文字が語中形になる。

2.3.3 ZW N-Join

モンゴル語専用の制御文字ではない。蒙古文字の間に ZW N-Join が挿入されると、直前の文字は語末形、直後の文字は語頭形になる。

2.3.4 異体字

それぞれの蒙古文字は、高々3 個の異体字を持つ。文字の直後に variant selector を置くことにより、異体字の番号を指定する。variant selector による指定がなければ、基本形となる。variant selector を、次の表に示す。

ニモニック	文字コード	機能
FVS1	\u180b	異体字１を指定する
FVS2	\u180c	異体字２を指定する
FVS3	\u180d	異体字３を指定する

3. Unite の概要

複数言語を組み合わせてテキストを編集できるエディタの製作が目標である。それぞれの言語に適した方法で、編集作業ができる仕組みを構築する。また、中核部分を再利用可能なクラスとして実装して、これを利用したツール、たとえば Java の JTextArea のようなウィジェットを作成できるようにする。すでに存在している多くのフォントを利用できることと、幅広いプラットフォームへの対応を考えて、開発言語に Java を使い、当面内部コードに Unicode を採用する。

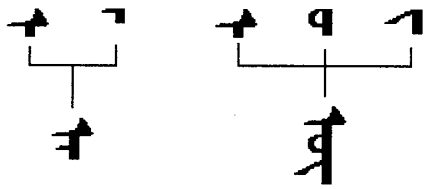
4. 蒙古文字の表示サブシステム

蒙古文字の字素を定義したフォントから、目的の字形に必要な字素字形をすべて取得し、それらを合成することで、１つの字形を生成するしくみを作成した。生成される字形は、Java2D の Shape インタフェースを実装したオブジェクトとして生成される。

4.1 字素フォント

高橋氏が作成した蒙古文字 TrueType フォント[3]は、蒙古文字の字素 141 個の形を定義したもので、質の高い字形を生成できる。フォント作成手間を省くため、本研究では、このフォントを利用することにした。

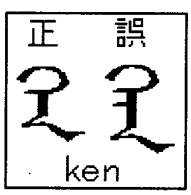
このフォントは、蒙古文字を構成する字素の外形を定義しただけのものなので、1 文字分の字形を得るには、１つ以上の字素を組み合わせる必要がある。つぎに、母音字 ‘a’ と ‘oe’ それぞれの語頭形についての組み合わせの例を示す。



aの語頭形　　oeの語頭形

4.2 字形テーブル

文字一つを構成する字素の組合せ方を記述したデータを数字、句読点、母音字、子音字について作成した。その他に、中間語末形や合字など、「子音字＋母音字」で特別な字形をとるパターンがある。右に合字の例をあげた。ken(誰)という単語である。合字を考慮せずに字形を合成すると誤例のようになってしまう。単に子音字の字形と母音字の字形を並べただけでは不十分なのだ。そこで、「子音字＋母音字」で１つの字形を表すこととして、このようなパターンすべてについて、字素の組合せデータを作成した。



4.3 文字データと字形データの対応

Unicode 文字１つが、１つの字形に対応しているわけではない。語中変化により１文字が複数の字形をもつ場合がある。制御文字を付加することにより、字形変化を強制的におこなった場合は、複数の Unicode 文字で１つの字形を表す。したがって、Unicode 文字列中のどこからどこまでが、ある１つの字形に対応しているのか（字形境界）を検出する必要がある。そのために、状態遷移表を使って文字列を解析し、Unicode 文字列中の字形境界を決定することにした。字形境界が決定したら、字形テーブルを参照して、字素の組合せデータ得ることができる。そして、各字素の字形は、Java2D API を使って、フォントから Shape オブジェクトとして取得する。各字素の Shape オブジェクトを合成して、１つの字形を生成する。

4.4 字形オブジェクト

字形を表す Shape オブジェクト、字形のメトリック情報、Unicode 文書内におけるオフセットと長さを保持するオブジェクトを、個々の字形１つづつ作成する。表示している文書全体分の字形オブジェクトは、リストで管理する。

5. 今後の課題

4.1 既存の文書への対応

モンゴル語を計算機で扱うときに、蒙古文字が使われることは稀であって、ほとんどの文書が、キリル文字または、英数字を使って蒙古文字を表現している。多くのモンゴル研究者が、英数字のみを使用した表記法を使って、文書を保存している。ただし、どの蒙古文字をどういった英数字の組合せに対応させるのかについては特に標準があるわけではない。これらの文書を Unite で扱うためには、文字列を Unicode に変換する必要がある。

6. 参考文献

[1] 三上喜貴/ 文字符号の歴史, 共立出版, 2002 年 3 月

[2] Namusrai Yumbayar/ Tradiotional Mongolian Script in the ISO/IEC 10646 and Unicode Standards, MLIT-4, October 1999

[3] 高橋まり代/ 言の葉, http://mariyot.hoops.livedoor.com/index.htm

[4] Unicode Consortium, http://www.unicode.org

[5] 片岡裕/ 国際化・多言語化の基礎と実際（前編） Bit 1997/3 Vol.29 No.3

[6] 片岡裕/ 国際化・多言語化の基礎と実際（後編） Bit 1997/4 Vol.29 No.4