Unicode, Aleph, and You

Prepared for presentation to the
North American Aleph Users' Group
14 June 2004

Charles Husbands
Harvard University Library
Office for Information Systems
charles_husbands@harvard.edu

Origins of Unicode

- Business internationalization (i18n)
  - Isolated national markets obsolescent
  - One need: an expanded character repertoire, a “Universal Character Set”
- Two groups begin to tackle this problem
  - ISO: The behemoth International Standards Organization
  - The Unicode Consortium: a group of U.S. hardware and software producers, etc. organized for this specific purpose.

Cooperation triumphs!

- In 1991, ISO and the UC begin seeking a common solution. They succeed.
  - Drafts from both bodies compared and unified in Unicode 1.1 and ISO/IEC 10646-1:1993
  - Character repertoire and encoded values precisely the same.
  - Some terminological differences
  - Grander scope of ISO's vision

A few of the major pre-1993 character encodings

- 7-bit encoding – 128 chars. max.
  - ASCII (34 positions reserved for control characters)
- 8-bit encoding – 256 chars. max.
  - ASCII extended by another 7-bit “code page”
    - e.g. ANSEL for MARC-8 Latin
    - e.g. ISO 8859-1 (Latin-1) for many other applications
- Variable length encoding – max. varies
  - Often ASCII is the Latin portion with another script employing 16 bits
    - e.g. Shift JIS (Japanese Industrial Standard)

The 1993 encodings

- Unicode 1.1
  - 16-bit encoding – 65,536 chars. max.
- ISO 10646
  - 32-bit encoding – 4,294,962,296 chars. max.
    - UCS-4 uses all 32
    - UCS-2 uses 16
- UCS-2 and Unicode 1.1 are identical
  - The “space” accommodating UCS-2/Unicode is called the Basic Multilingual Plane (BMP)
- It’s upward and onward since ’93.
  - Unicode 4.0 includes 96,382 characters

Unicode design principles

- There are about a dozen.
  - Here are two. They are related.
- Unicode encodes plain text.
  - “Plain text must contain enough information to permit the text to be rendered legibly, and nothing more.” The Unicode Standard 4.0, p.18
- Unicode encodes characters, not glyphs
  - A glyph is the visual expression of a character

Three* areas where these things matter

- Keyboard – the realm of input
  - Glyphs to characters
- Storage – the realm of processing
  - A character realm
  - Including inter-system communication
- Font – the realm of display
  - Characters to glyphs

*Thanks to Edwin Hart of John Hopkins for suggesting this simple topograpy.

What's a UTF?

- UTF stands for UCS Transformation Format. It is a method of representing an encoding in units of more convenient length with no loss of meaning.
- UTF-8 redistributes the bits of a UCS code value into one or more octets (bytes.)
  - The result is like a variable length encoding.
- There are also UTF-16 and UTF-32.
- All three are legitimate representations of the same code and can be transformed from one to another algorithmically without loss of meaning.

Is Unicode the answer?

That depends on the question.

It takes a village

- Operating systems
  - Windows
  - Unix
- Database software
  - Oracle
- Application software
  - Aleph
  - E-mail
  - Text editors
  - Input method editors
- Browsers
- Fonts
- Z39.50 conventions

Aleph…

- has used Unicode since Version 14, but
- database still defined to Oracle as ASCII-7 in Versions 14 and 15.
  - UTF-8 encoded data were correctly preserved.
  - Aleph functions worked correctly.
  - using non-ASCII data directly from Oracle was difficult.
- Database defined as UTF-8 in Version 16.
  - Unicode through and through.

Aleph input

- Online input tools
  - Standard Windows keyboards
    - various locales to choose from
  - Custom Windows keyboards
  - Input method editors, esp. for non-latin
  - The Aleph [formerly floating] keyboard
  - Aleph Unicode mode (pf_11)
  - Locally-built macros defining key combinations

Aleph processing

- Records going to or coming from other systems may need character conversion
  - Aleph programs
    - MARC8_TO_UTF
    - Others for other encodings
  - MARC8_TO_UTF tables in $alephe_unicode
    - marc8_lat_to_unicode
    - marc8_eacc_to_unicode
    - Others for other non-latin scripts

MARC-8 Latin review

- Repertoire
  - ASCII (ANSI X3.4, ISO/IEC 646)
  - ANSEL (ANSI Z39.47)
    - Includes combining marks (diacritics)
  - Subscript numerals and punctuation
  - Superscript numerals and punctuation
  - Three Greek “symbols”
- Use of special (technique 1) escapes for subscripts, superscripts, and Greeks

marc8_lat_to_unicode

- Replaces pair of older one-way tables
  - ansel_lat_to_unicode and its partner
- Supports conversion in either direction
- Format
  - Column 1 – MARC-8 value
  - Column 2 – Unicode value
  - Column 3 – Optional comment
- Does not need to be sequenced in column-2 ascending order

Customizing marc8_lat_to_unicode

- Why would you want to?
  - The march of time and technology
    - Precomposed vs. decomposed decisions
    - Differing opinions on the proper correspondences between MARC-8 and Unicode
    - New characters, possibly

Precomposed characters – pluses

- Fonts display them more successfully
- Avoid having to remember to put combining marks *after* the base character
- The common choice for Aleph, at least in North America

Precomposed characters -- minuses

- Larger repertoire for staff to cope with
  - Bigger Aleph keyboard
  - Longer list of codes for Unicode mode entry
- Not currently allowed in MARC21 UTF-8 records for exchange
  - A big problem for export from Aleph if your exchange partner plays by the MARC21 rule
  - But rule is likely to change.

Preparing for Version 15

- April 2002 meeting in Chicago
  - Ex Libris and customer dialog about Unicode implementation in Version 15
- Harvard STP planned for July 2002
  - Proposed enhancements to marc8_lat_to_unicode
  - Most accepted by Ex Libris

What was changed from 14 for Aleph generally?

- Extensive precomposed repertoire added
- Some of which involved folding multiple input strings to one output value
  - e.g. for Vietnamese
- But a few concessions to available fonts
  - e.g for Romanian s or t with comma below.

1EA5 E2E361 ấ
1EBF E2E365 ế
1ED1 E2E36F ố
. . .
1EA5 E3E261 ấ
1EBF E3E265 ế
1ED1 E3E26F ố

What else did Harvard change?

- Identification of some MARC-8 combining marks with Unicode interpretations
  - High comma offset = caron (haček) on letter with ascender -- d̓ t̓
  - High comma centered = cedilla on letter with descender -- g̓
- Miscellaneous
  - e.g. undotted i with circumflex = Latin-1 i with circumflex

What remains outstanding?

- Font-based decisions
  - e.g., review the Romanian case
- Compatibility area usage
  - Diacritics that span two characters
    - Combining double inverted breve (ligature)
    - Combining double tilde
- Add more precomposed forms
  - Languages infrequently cataloged may want them as need grows.

Going back to MARC-8

- What's not perfectly reversible?
  - Values folded during conversion to Unicode
  - The three Greek "symbols"
  - The "ASCII clones" including space
    - an issue for non-latin reconversion generally, not an Aleph issue

Unfolding the folded (not really)

- Conversion from Unicode to MARC-8
  - UTF_TO_MARC8 uses marc8_to_unicode tables
  - First line found for a given Unicode value is used, subsequent ones generate a warning message
  - MARC-8 sequence of table no longer required
  - You must choose the value you prefer to output for all occurrences.
    - Rearrange marc8_lat_to_unicode to achieve this.

Rearranging marc8_lat_to_unicode

- Create a preferred value section at the head of the table
- Copy lines with preferred values into it from their place in regular sequence
- Comment out the original lines so that you don’t lose track of them
- Only lines that come after a non-preferred line for the same Unicode value need to be given this treatment.

Rearranging marc8_lat_to_unicode example

2113 C1 SCRIPT SMALL L
1EAA E4E341
1EC4 E4E345
1ED6 E4E34F
1EAB E4E361
1EC5 E4E365
1ED7 E4E36F
010F ED64
01E9 ED6B
013E ED6C HVD
0165 ED74 HVD
!* Preferred values above

. . .

. . .
1EEE E4AD
1EE1 E4BC
1EEF E4BD
!*preferred 1EAA E4E341
!*preferred 1EC4 E4E345
!*preferred 1ED6 E4E34F
!*preferred 1EAB E4E361
!*preferred 1EC5 E4E365
!*preferred 1ED7 E4E36F
1EB4 E4E641
1EB5 E4E661

. . .

α β γ

- The three Greek symbols receive unusual treatment during reconversion to MARC-8
  - In Unicode they have no codes distinct from the letters in the Greek alphabet
  - On reconversion it is not possible to tell whether they are MARC-8 Greek letters or MARC-8 latin Greek symbols
  - They are converted as part of the Greek alphabet using the appropriate technique-2 (ISO-2022) escape sequences. Requires creation of an 066 field in the MARC-8 record.
    - Include fix_doc_create_066 in your output fix routines.

Aleph display

- Unicode requires special fonts for character repertoire extending beyond ASCII or Latin-1.
- The really big fonts are proprietary.
- One font or a collection can do the job.
  - Client and opac considerations differ.
  - What will users have available?
- FONT.INI can be manipulated for many areas of display in Aleph client.
  - Some manipulable areas are ASCII-bound.

…and you

- What's your role?
  - Know enough about character encoding so that you can understand what you see.
    - Look at www.unicode.org
    - Look at Aleph Document: Application of Unicode.
    - Don't be frightened, Persevere.
  - Choose
    - details of MARC-8 – Unicode conversion
    - how to enter and store characters
    - what fonts to use and support