Bloat of Data in the Unicode Era

Behdad Esfahbod
behdad@behdad.org

The FarsiWeb Project
http://www.farsiweb.info/

Desktop Developers' Conference '05

July 18, 2005

Agenda

- The Importance of Unicode
- Unicode Character Database
- Common Locale Data Repository
- Future Plan

The Importance of Unicode

- **The Old Days$^{\mathrm{TM}}$:** Gazillions of 8-bit character sets
- **ISO 10646:** A unified character set
- **The Unicode Standard:** And unified algorithms to deal with these unified character set

Intro to Unicode

- Currently at 4.1.0 release
- Not 16-bit, 21-bit
  $(16 + \log_2 17 = 20.087462841250339\text{-bit})$
- A unique non-negative integer less than 1,114,112 assigned to each character

Intro to Unicode (continued)

- Slightly less than 100,000 characters registered so far
- New scripts and characters are encoded with each release
- Major releases published as a book, with online updates for minor releases

Intro to Unicode (continued)

- The book is available online as PDF files
- The updates and other references are available in plain HTML
- Data files as text files

Architectural View of Unicode

- **The book:** The Unicode encoding model, Encoding model and *issues* for individual scripts

- **Key specifications:** *Standard Annex*, *Standard Report*, or *Technical Report*, algorithms for rendering or otherwise dealing with text

- **Data files:** *Unicode Character Database*, text files that define character properties and internal mappings

Key Specifications

- Unicode Collation (UCA)
- Bidirectional Algorithm (Bidi)
- Normalization (NFC, NFD, . . .)

Unicode Character Database

- More than 70 character properties
- The canonical character name, eg. U+0041 is *LATIN CAPITAL LETTER A*
- The most commonly used one is the *General Category*, eg. U+0041 is *Lu*: Letter, upper case
- Mostly binary and enumerated properties

They Show Up Everywhere

- Glibc character types: `isalpha`, `isdigit`, `isprint`, . . . (`ctype.h`)

  Warning: The C standard limits the value of some of these functions

- Convenience and module libraries: Glib has some, Qt's `QChar` class has some, Python's `unicodedata` module has the important ones, Perl supports all of them in regular expressions

And in (Some) Applications

- Gucharmap uses them of course
- Terminal emulators use the `wcwidth` function from Markus Kuhn
- But not much more

Where Else is it Useful?

- I want my editor to show the character names
- Unicode regular expressions (PCRE)
- Wherever a list of scripts is useful

The Problem

- Glibc is not available everywhere
- The manual and Perl-script approaches, the 2-year cycle, performance
- Different versions of the data around : And old Glibc, Glib, FriBidi in Pango, wcwidth in gnome-terminal, gucharmap, ...

The Problem (continued)

- File formats, default values, etc, change. Can go unnoticed
- High entry cost for getting the data in your application
- Support for new scripts is broken for years

Ideally

- A new approach to Unicode libraries: Only data, no converters, no algorithms
- A central library exporting the UCD efficiently
- Easier maintenance, easier update

Ideally (continued)

- Better memory overhead, more sharing
- Problems in format change, etc have more chance to get noticed
- Different versions of the UCD can live together (IDN requires 3.2)

Ideally (continued)

- A runtime library that you can query properties efficiently
- A development kit that generates efficient lookup-table code for pedantic projects
- Central translation effort for property names, script names, character names, etc

Where are We Now?

- Planning
- Got the name: **gNUichar**
- A binding-friendly efficient design
- Fetch compressor and bits from different projects
- Release and advertise

Localization

- Much trickier than internationalization
- $O(n^2 + k.n)$ where n is the number of languages and k is the number of different atoms
- More exposed to the end user: date formats, number formats, language names, country names, etc

Locale Data

- Glibc has the basic functionality, but very limited
- Evolution has a handful of date formats to translate, other modules have too
- Several projects maintain a list of language names and countries, that get translated separately
- Paper sizes, date formats, currency, timezone, etc

The Problem

- Again, Glibc is not available everywhere
- If no Glibc locale, no support
- Translating country names and language names is quite hard
- Maintenance is a nightmare

Common Locale Data Repository

- A group effort coordinated by the Unicode Consortium
- Backed by companies like IBM, Sun, Apple, etc

CLDR Architecture

- Current version is 1.3.0
- Released as a set of XML files
- Using inheritance to reduce the effort
- XML and file-based inheritance, makes it hard to use

The Problem (continued)

- The XML architecture makes it pretty hard to use CLDR in an application

- Overlaps with the Glibc data

Ideally

- A central library to export the CLDR efficiently
- Qt has its own locale system, convert
- GNOME doesn’t have a locale system, push in

Currently

- A new list created for discussion, `locale-list@gnome.org`
- ICU may be finally useful
- A long way to go, help needed, in design and implementation

In the Future

- Get these two libraries released
- Build a higher-level locale library for GNOME
- Start cleaning up GNOME and KDE
- What else? Questions?