Accessibility of Unencoded Glyphs

Ken Lunde

CJKV Type Development
Adobe Systems Incorporated

Adobe

ftp://ftp.oreilly.com/pub/examples/nutshell/ujip/unicode/iuc13-a10-paper.pdf
ftp://ftp.oreilly.com/pub/examples/nutshell/ujip/unicode/iuc13-a10-slides.pdf

Status of Today's Input Methods

Adobe

- Input Methods (IMs) allow access only to "encoded" characters
  - In a legacy encoding, such as Shift-JIS, EUC-KR, or Big Five
  - In Unicode encodings, such as UCS-2, UTF-8, or UTF-16
- IMs send only character codes to applications after the user has gone through the input process
  - There are no APIs in place that can serve to enhance the communication between IMs and applications
- Almost all IMs allow the user to add new entries to the conversion dictionary
  - Most "gaiji" font packages come with supplemental conversion dictionaries for multiple IMs

Status of Today's Font Formats

- **Today's font formats are not bound to any particular encoding—all glyphs are indexed through simple integer values**
  - — GIDs (Glyph IDs) in TrueType
  - — CIDs (Character IDs) in CID-keyed fonts (PostScript)
  - — The largest CIDFont of which I am aware has 55,880 CIDs (includes CNS 11643-1992 plus CNS 11643-1986 Plane 15)
- **Today's CJKV fonts include many unencoded characters**
  - — Adobe Systems' Japanese fonts include a total of 250 JIS78 (JIS C 6226-1978) kanji variants that are not accessible under most circumstances
- **Today's CJKV fonts include glyph substitution tables that can be accessed by applications**
  - — Adobe Systems' sfnt-wrapped CIDFonts for MacOS

Status of Today's Applications

Adobe

- QuickDraw GX—now called AAT—applications dynamically support QuickDraw GX fonts' typographic features
  - Adobe Systems' sfnt-wrapped CIDFonts also supported
- Other applications recognize and react to fonts' glyph substitution tables
  - Adobe Illustrator Versions 5.5J, 7.0, and higher
  - Macromedia FreeHand 8.0J
  - Adobe PageMaker 6.5J (but not in a very WYSIWYG way)
- Common glyph substitution features
  - Simplified ⇒ Traditional
  - JIS78 kanji variants
  - "Expert" kanji variants

To Encode, Or Not To Encode

Adobe

- Encoding all additional characters in user-defined regions is always an option
  - Shift-JIS encoding supports up to 1,880 additional characters
  - UCS-2 encoding supports up to 6,400 additional characters
  - UTF-16 encoding supports 131,072 more
- Characters that can be considered variant forms of standard (that is, encoded) characters can be left unencoded
  - To be accessed through glyph substitution features
- Characters that are not considered variant forms should be encoded
- Approximately 40 of the 250 JIS78 kanji variants are in Unicode—the rest have been unified

Who Needs Variant Forms?

- Professional publishing requires access to a variety of variant forms
  - — For dictionary publishing and other complex documents
- Consider the following Japanese kanji variants:

Standard	Traditional	Encoded Variants	Unencoded Variants
学	學	斈	
剣	劍	劒 劔 劒 釼	
辺	邊	邉	邉 邊
弁	辨 瓣 辯	辦	

Other Types of Variants

- Annotated kana:

  あ ⇒ ㈀ ㊀ ㊁ ㊂ ㊃ ㊄ ㊅

- Annotated numerals:

  1 ⇒ ⑴ ① ❶ ⓵ ⓵ ❶ 1.

The Two-Stage Input Model

Adobe

- Given today's state of IMs and applications, two distinct and separate stages are required to access unencoded characters:
  - The user inputs characters through the IM, which get passed to the application—these are "encoded" characters
  - The user accesses glyph substitution features through the application's UI
- Consider the following phrase, which is input through the IM:

  学校の桜並木に黒い虫
- After accessing "Traditional" and "JIS83" glyph substitution:

  學校の櫻竝木に黑い蟲

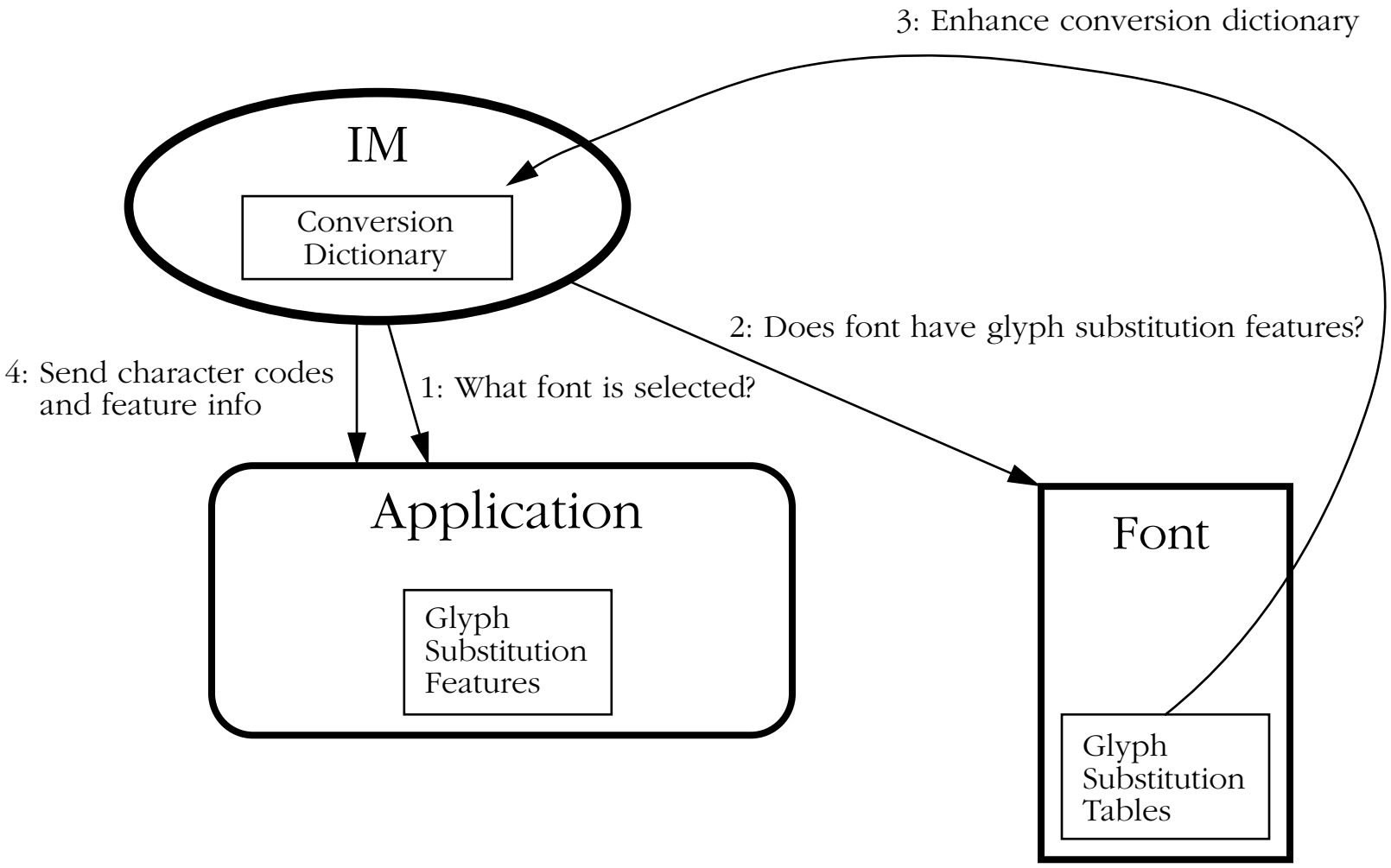
The Single-Stage Input Model

- **Put simply, to make glyph substitution features accessible at the IM level (upstream), where the users spend most of their time for character input**
  - Significant performance enhancements through time savings
- **Fundamental changes are required, such as new APIs and a common interchange format**
  - For IMs, applications, and operating systems
  - IMs no longer send only character codes to an application
  - Applications must be prepared to accept more information than only character codes
  - IMs and applications must be able to communicate better, perhaps through the operating system

The Single-Stage Input Model (Cont'd)

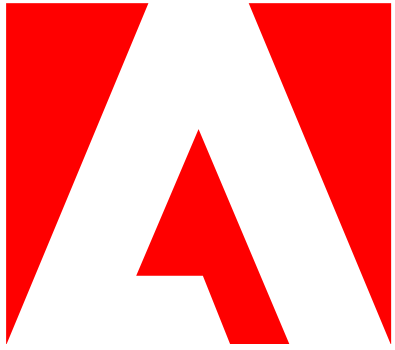
- Applications are still required to expose UIs for glyph substitution features
  - For post-input editing purposes
  - Working with legacy documents
- Necessary processes:
  - IM must ascertain what font is selected in active application
  - IM must ascertain whether font has glyph substitution tables
  - IM must use font's glyph substitution tables to enhance conversion dictionary by making unencoded glyphs accessible
  - IM must send character codes along with additional information that will automatically invoke the appropriate glyph substitution feature within the application

The Single-Stage Input Model (Cont'd)



Advantages of Unencoded Characters

- Treated the same as their encoded counterparts
  - Consider searching operations
  - Consider sorting operations
- Unencoded glyphs are treated the same as their encoded counterparts because the underlying character code is that of their encoded counterpart



Adobe