L2/10-327

Subject: comments on L2/10-280, "Proposal to Add Variation Sequences..."
From: Peter Constable <petercon@microsoft.com>
Date: Sun, 8 Aug 2010 18:45:57 +0000
To: "unicore@unicode.org" <unicore@unicode.org>

I have several issues with the rationale provided in this proposal. Before launching into what may be felt to be a harsh critique, I wish to thank Karl for his work on this proposal and express my support for the user experience he seeks to enable: that users see text displayed in culturally-appropriate ways. I cannot agree with his proposed solution, though I do support the user impact which he hopes to achieve.

Part A of this document attempts to make the argument that variation sequences would be a good solution for certain needs in relation to Latin and Cyrillic scripts. In particular, A.1 attempts to make the case that alternatives to variation sequences are inadequate. The argumentation in A.1 has several significant holes, however, and I do not find it to provide a convincing case. I'll comment on select points from A.1, and then make some general comments.

The argumentation, in overview, runs as follows:

1. "universal fonts" [i.e., widely-used fonts such as Arial] are widely used but do not serve the needs of particular communities whose typographic conventions involve particular glyph forms

2. "universal fonts" are used on the Internet since site developers can't assume specific, non-widely-used fonts will be on the reader's system

3. Many Internet sites are maintained by non-pros who do not have skill needed to use server-side technologies that could provide the needed, non-widely-used fonts to the reader's system

[PC] It appears that the proposer may not be aware that such server-side functionality is being integrated directly into the latest versions of W3C recommendations without the need for special server-side technologies and that this functionality is already supported in at least some browsers, with strong momentum for adoption by most or all browsers. Thus, the barrier of amateur skill level is eliminated. (This was also mentioned by Martin D  rst in discussion on the Unicode list.

4. OpenType provides general mechanism to select alternate glyphs within a font, but not standard conventions to specify a particular glyph.

OpenType specifies clearly-enough a mechanism that is not difficult to implement and that can be used to specify culture-specific glyphs: the combination of a "language system" table (which every OT font that supports alternate glyphs cannot avoid including) and a "locl" (localized forms) feature table: a language system table can be associated with a "language system" tag, which declares the culture and which are standardized, being defined in the OT spec; the 'locl' feature is necessarily associated with one language system table and is the means to trigger the relevant glyph-substitution actions for that language system.

The OT spec recommends that 'locl' feature be enabled by default, and the expectation is that OT implementations will select an language system based on the language of the content or other appropriate means. Admittedly, many OT implementations today do not implement this behaviour, and that has led to fewer fonts that utilize this mechanism to support multiple cultures in a single font. More implementations that _*do*_ implement this behaviour are becoming adopted, though. This, it is expected that availability and utilization of this mechanism will become common in the next few years.

5. Moreover, OpenType fonts that include such functionality are beyond the reach of small font-development teams.

This part of the argument is seriously since since, in fact, there are more means available today to add support for OT features in fonts than there are to add support for variation sequences.

6. Furthermore, only expensive software products support this functionality; “Common text processing applications like older versions of Microsoft Word® do not”.

First, as mentioned above, this functionality is likely to become quite common in the not-too-distant future. Secondly, this argument is seriously flawed in that the proposed solution of variation sequences will also not be supported in “common text processing applications like older versions of Word”.

7. It is harder for a font to support a number of cultures [each of which may involve multiple distinctive glyph forms] than it is to support a mechanism such as variation sequences that allows alternate glyphs to be specified on an individual basis (e.g. if there were, say 50 variation sequences to be supported).

There is some truth that it would be harder for a font to support many cultures than to support selection of particular glyphs on an individual basis. But the argument has a fallacious logical leap: it assumes that only the variation sequence mechanism can support that glyph-by-glyph approach, implying that OpenType Layout mechanisms cannot. This is untrue, however. In fact, a set of OpenType features with precisely this functionality are defined: see http://www.microsoft.com/typography/otspec/features_ae.htm#cv01-cv99 for details. Using these features, a user could refer to documentation to find the right combination of cvXX features needed to select the glyphs in that font needed for their culture and configure their software to use that combination.

Granted, the association of individual cvXX features to particular glyph forms is not conventionalized and may vary from one font to another. But the cvXX features were not intended to replace the language system / ‘locl’ mechanism described above; rather, they were meant to complement one another so that a font developer could define language systems in their font for better-known cultures and still leave a way for a user whose culture is lesser-known to still get the glyph variations that they need.

8. Locale data could contain applicable variation sequences.

This entails that locales must be updated rather than fonts, and that text processing applications will start requesting the relevant locale data at authoring or display time. It does nothing for all the “[c]ommon text processing applications like older versions of Microsoft Word” or any application running on older systems without this updated locale data.

Note also that fonts must still be updated to support this.

9. Such locale data could be utilized in rendering systems.

Or, the rendering systems could apply the applicable OpenType language system. (Note: both solutions equally assume that fonts have been updated and that the language / locale of content is known.)

10. Expecting users to select culture-specific fonts is no different than an era of custom-encoded fonts.

This is clearly false: custom-encoded fonts do not permit _*any*_ interchange of data apart from the correct font. Unicode-encoded content that requires a user to select particular fonts to get certain glyph forms is still Unicode-encoded content: it is legible to all users and fully interchangeable.

Also, note that users already select particular fonts to achieve desired design effects. Expecting users to select particular fonts to get particular glyphs is something that most users already do to at least some extent.

But note also that, with increasing numbers of implementations supporting the OpenType Language System mechanism, it will not always be necessary for users to select particular fonts to get culturally-appropriate glyphs.

11. Having to create culture-specific versions of “universal” fonts should be an obsolete notion in a Unicode age.

Such an issue is completely orthogonal to Unicode, so the assertion is incorrect: being in a Unicode age implies nothing one way or another with regard to display details (which the Unicode standard makes clear).

But, it is certainly true that this should be obsolete in an age of smart-font rendering technologies, like AAT, Graphite and OpenType. With all technologies, adoption takes some time. In the case of the OpenType mechanisms relevant to this, it has taken longer than expected but, as mentioned, increasing adoption in implementations is taking place.

General comments:

I am given the impression that the author of this proposal started with a solution and then explored ways to argue in favour of that solution. But in forming his arguments he has not taken a comprehensive view of the overall issues. For instance, as noted, he infers that the proposed solution could be used with existing software and fonts and fails to notice that it would require comparable updates to software and fonts as a smart-font-based solution.

Not only that, he fails to note that his solution would also require changes to existing content in order to realize any enhancement in rendering that content, while a smart-font- based solution would not. This is a major oversight.

The impact on content also points to another problem with the variation-sequence solution: it is effective only when content is authored to include a variation sequence in each and every instance of a relevant character for a given culture. This presents many points of failure: to ensure consistency, every instance of a character must include the correct variation selector. It also assumes a major change in at least component of an authoring scenario:

a) that authors will revise their keyboard input behaviour to use alternate key sequences in every applicable case
b) that authors will only use input methods tailored specifically for their language
c) that authoring software will automatically insert the correct variation selector in all the right places

Not only does (a) assume significant changes in user behaviour, it also assumes that keyboard layouts are revised to support input of variation selectors, or new layouts provided with this ability.

Implicit in (b) is the requirement that language-specific input methods are created. Thus, while a single extended-AZERTY layout might suffice for many different West African languages if font-based mechanisms were used to select culture-specific glyphs, the variation-sequence solution apparently assumes that there must be several culture-specific, extended-AZERTY layouts.

Alternative (c) assumes significant changes in software. Not only is this counter to the argument in A.1 that a solution should work with “[c]ommon text processing applications like older versions of Microsoft Word”, it also misses the problem that software would need to include data for many different languages. This implies the likelihood that only a limited number of languages would be supported. Note that the complexity of supporting data for many different cultures was one of the arguments given in A.1 against an OpenType-based solution. (Regardless of the solution, information must come from somewhere: from data in software, from data in fonts, or from user knowledge.)

The scenarios that this proposal seems to have in mind are cases in which all instances of a character within a given record in some particular language should be displayed with a particular glyph form. Variation sequences are not the best solution for this scenario: they are much better suited to scenarios in which a variant must be specified for a limited number of cases.

Of course, the need to display a particular glyph is not the only requirement for variation sequences: there should be a need also to capture the particular glyph rendering in plain text. The proposal does not demonstrate a need. Indeed, because it is talking about behaviour that could be predicted from a cultural attribute applied to entire runs of text or entire documents, plain-text representation is hence never a requirement in any scenario in which such metadata can be provided.

In summary, there are numerous problems with the rationale provided in this proposal to use variation sequences for the scenarios covered. The costs of this solution versus alternative font-based solutions is not fairly assessed, not are the practical benefits. It does not establish a need for plain-text representation of the glyphs in question. Thus, I would not recommend its acceptance by UTC.

Peter