L2/03-116

Generative Arabic: implementation options

Jonathan Kew, SIL International
March 6, 2003

There appears to be broad agreement that we should encode a set (details still to be worked out) of combining dot patterns and similar marks for productive use with Arabic base letters. The most difficult issue to be settled is that of equivalence between the existing precomposed letters and the decomposed representations that could then be encoded.

We have several options to consider, each with different strengths and weaknesses:

1. No equivalence is defined; there are simply multiple spellings that are indistinguishable on the surface.
    a. Lowest documentation burden and cost of implementation
    b. Confusing/unhelpful for users, when search/sort/index/... operations don't produce expected results
    c. Entire Arabic writing system becomes open to "character spoofing" on a massive scale, with implications for security/fraud/deception
2. As (1), but some "weaker" equivalence is defined between decomposed and composed representations (similar to the concept of compatibility decompositions; published stability policy prohibits the addition of actual compatibility decompositions to the existing Arabic characters).
    a. Can be ignored by general processes; no cost burden in order to be conformant
    b. Defines a standard basis for specialist processes to implement the behavior users would expect
    c. Does not solve spoofing issues, as generic processes would not recognize the two representations as equivalent (e.g., in name/URL lookup or comparison)
3. No equivalence is defined, but the Standard prohibits the use of decomposed representations where composed characters are available (at least specifies visibly "broken" rendering; I don't think we can claim the text is actually ill-formed and should be rejected by other processes).
    a. Attempts to spoof are thwarted
    b. Fairly easy to document
    c. Confusing for users in that combinations that "should" work don't
    d. Significant burden for rendering implementers to special-case all sequences that correspond to existing composed letters
    e. Hinders consistent use of "fully decomposed" text for specialist analytical purposes
    f. Preserves an inconsistent "mixed model" forever as the only conformant way to handle extended Arabic-script text
4. Decomposed representations are canonically equivalent to existing composed letters, with the composed forms being the normal form (both NFC and NFD) to preserve stability of NFs.
    a. Behaves in accordance with user expectations as to what is "the same"
    b. Clean, consistent rendering model

c. Gives a single, consistent model based on the underlying structure of the writing system, with the precomposed letters available (and normally used) simply as a more convenient representation—much like Latin. Greek, etc.

d. Requires revision of normalization algorithm (preserving stability of results for all existing text): implementation cost of updating to the new version of Unicode becomes greater than simply loading new data tables

It seems to me that there is a cost to be paid for the new flexibility we gain; the different approaches call for this cost to be borne by different people. Under option (1), it is an ongoing cost for users of the standard (even users with no interest in Arabic, to the extent that the security issues of character spoofing are a concern). Option (2) is essentially similar, as the new equivalence it defines would be used only in specialist processes.

Under option (3), the cost is primarily borne by implementers of rendering systems (and fonts, perhaps—the exact mechanisms used might be system-dependent); there is also an ongoing cost for Arabic users in the requirement to work with a mixed model. Under option (4), the cost is borne by implementers of processes that normalize, in that the engineering requirements to support the new version of the standard are greater than otherwise (although not excessively so, in my judgment, given that the NFC algorithm already implements a composition step; NFD would simply have to adopt an equivalent step using a different data table).

There may be yet other solutions that could be considered, and that might offer different cost/benefit tradeoffs; any suggestions that offer a productive way forward are welcomed.