Response of IP to Gurmukhi LGR of 30th Apr. 2018

DATE: 2018-05-16

Overview

This document provides IP response to the Gurmukhi proposal dated 2018-04-30.

## General comments

|  |  |  |  |
| --- | --- | --- | --- |
| Item | Issue | IP Comment | GP Comment |
| Previous responses: | The IP has reviewed ....2018-04-30.DOCX, .... XML, the test label files and the responses by the GP to ....DOCx which were updated in response to the previous feedback by the IP.  The Gurmukhi documents appear to the IP to close all previous issues, hence will not be further referenced. | IP thanks the GP for their ready responses. | - |
| Current situation | Now that IP has seen the complete set of other LGR proposals for Indic scripts, there are a few additional issues that IP thinks should be addressed, in addition to some further editorial suggestions. | The WLE rule issue, the variant issue, and the few editorial items from this feedback document do need to be addressed.  Details as below. | Tried to address the issues |
| Basis for labels in phrases as a well as words of ordinary language | A common issue in labels is the lack of interw0rd punctuation or space. Issues related to placing words or phrases together without any separation appear not to be addressed. Because multi-word domain names are common for other scripts, they can be expected for the Gurmukhi script as well. Therefore, a more complete discussion of the issue in the context of describing the script behavior in Section 5 would be advisable.   This issue is exacerbated by the fact that the test files / corpora that the IP has been using in reviewing LGRs generally contains single or compound words, but not multi-word expressions written without the expected inter-word separators. | It would be useful if the script proposals explicitly addressed any potential issues caused by concatenating words directly, presumably in § 4.1.2 No punctuation marks, or Section 5. | There are no word concatenation issues in Gurmukhi. |
| Policy on Joiner characters (general) | Zero-width joiner and Zero-width non-joiner are not part of the MSR, because the Procedure rules out any CONTEXTJ code points.  However, these code points are commonly used in Indic scripts to control the formation of conjuncts. That means that not incorporating them into an LGR prevents certain terms from being displayed and/or represented correctly.  We do have in all NeoB LGRs a part (§ 3) that purports to give the background information for the script; these chapters go into great details on things like the early history, but are curiously silent on the joiner characters. | Ideally, all NeoB LGRs should have a paragraph in that chapter about (a) whether the script commonly uses ZWNJ / ZWJ (b) what the effects are (i.e. what they control) (c) whether their use is 'optional' or stylistic, or not (i.e. semantic) (d) mentions that ZWJ / ZWNJ are out of scope for the Root Zone  When we run tests on the corpora, we will see many failing labels in some scripts that are due to U+200C ZWNJ. | Added in section 3.3.5 |
| Policy on Joiner characters (CONTEXTJ), specific to Gurmukhi | Gurmukhi LGR, Section 4.1.1  contains the text:  IDNA Protocol also excludes invisible characters Zero Width Non-Joiner (U+200C) and Zero Width Joiner (U+200D), as they require a CONTEXTJ rule. These are required in certain cases where a typical visual shape of an akshar is desired. | It would be helpful to add:  1) whether differently shaped akshars are a matter of style or orthography in Gurmukhi  2) whether these cases are frequent / rare  If the use of ZWJ/ZWNJ is rare (and/or somewhat optional) in Gurmukhi, then noting that in Section 4.1.1 would be adequate. Otherwise, a bit of  a discussion in Section 3 would be useful with an eye towards describing it in more detail as a feature of the script that cannot be modelled and to alert users that IDNs cannot express all features of the orthography. | Added in section 3.3.5 |
| § 6. Variants | 0A1C is also fit for a nukta in Gurmukhi, but corresponds to a geminate 0924 in Devanagari, which cannot be followed by a Devanagari nukta, so can be ignored. | Please therefore add this case, making the sequences  0917 093C  and  0A17 0A3C  blocked variants of each other.  Also consider whether to add a context rule for the first set: viz  ([^[:A:]])←  Meaning that none of the first set can follow an ADDAK. The motivation would not be to get the context right for ADDAK, but to be extra conservative where we allow a NUKTA. | 093C (Devanagari Nukta) and 0A3C (Gurmukhi Nukta) have been declared as cross-script variants |
| **§ 6. Variants**: on NUKTA, and similar marks of identical appearance across scripts: | NUKTA isn't accounted for in the cross-script variants, specifically re 0917 and 0A17.  In defining cross-script variants, the IP would like to call the GP's attention to the effect. Normally, cross script variants are not needed for diacritical marks because they cannot stand alone, so they cannot, by themselves, create variant labels. However, the situation changes if the respective base characters are variants. Because of the variant relation for 0917/0A17 any sequences of these followed by a NUKTA should also be considered variants (0917 093C / 0A17 0A3C). | The remedy would be to make the NUKTA a cross-script variant for Devanagari/Gurmukhi -- there is no need to restrict this relationship to specific sequences because the context rule and variant relation for the consonants must also agree to form a variant label, therefore implicitly providing that restriction.  Please verify that there are no other such cases. | 093C (Devanagari Nukta) and 0A3C (Gurmukhi Nukta) have been declared as cross-script variants |
| § 7. WLE Rules | The list of sequences in CN1 should be in code point order (here and in the XML) - but the details depend on whether the rules are reformulated in other changes. The principle is that all sets and lists are in code point order and currently, only CN1 is not. | Replace sequences in CN1 in code point order, in .docx and .xml | CN1 not being used now |
| § 7. WLE Rule 7.6 | WLE Rule 7.6 is overly specific.  This rule attempts to enforce that ADDAK precedes C3 or specific CN sequences. However, all of the specific sequences start with a code point that is a member of the set C3. Therefore, the sequences are not enforced.  The specificity of the rule for ADDAK makes the IP question whether it attempts to do more than necessary: at a minimum, the rules need to make sure that both Tippi and Addak cannot occur at the same time.  If it is acceptable, a simple fix would be to reduce the right-hand context to C3. | Fix and/or relax Rule 7.6 | Relaxed rule 7.6 |
| Alternative: | If, instead, the GP feels that Nukta following Addak needs to be constrained, then a Whole Label rule could be added enumerating the invalid sequences. Note that this would require also defining an <action> on that rule. | Because a Nukta, being a single dot below, is easily mistaken when it appears out of expected context, there may be some argument for an added restriction like this – however, there is also the argument that this level of detail begins to be a spelling rule, rather than a structural one.  GP is requested to consider the issue. | Already relaxed rule |
| § 9. References - MSR | The reference to MSR-2 should be replaced by a reference to MSR-3 (while the two versions are identical as far as Gurmukhi is concerned, we would like to claim that LGR-3 is based on "MSR-3" and therefore it helps to have the proposals reflect the latest version). | All citations, and the basic reference, should be updated to MSR-3 with correct publication date/ filename/ title for MSR-3 | Done |
| § 9. References  [0] | This claims to be a reference to the Unicode version 10 charts. However, the link is an "evergreen" one, that is it will point to the latest version of the Unicode chart then available. While Unicode does post "fixed" versioned versions of its code charts, those always cover \*all\* code points and are large and unwieldy as a reference.  (Code points will never move and names and decomps will never change, so accessing future versions will not invalidate the reference, but when accessed later it will identify as version 11.0 etc). | Simply remove the "Version 10.0" and replace it by an "Accessed on" with some reference date.  Note: While future versions might add some additional code points – that is of no issue for the purpose here, because MSR-3 defines the subset, so use of the evergreen URL for this purpose is OK. | Done |
| Detailed editing | Copy of .DOCX is attached | With “Track Changes” suggestions.  For some suggestions there’s an additional comment in the DOCX file alerting the GP that the suggested change should be carefully reviewed. | Taken care |

## XML points

|  |  |  |
| --- | --- | --- |
| **WLE (and corresponding changes in the XML**) | The LGR contains 2 unused classes. An unused class is one that is not referred to in any rule. The IP will remove unused classes from the published LGR during integration and leaving them in the proposal would set up for a needless inconsistency. Tags: there is no need to change the name for any tags, nor to remove "unused" ones. Tag values serve to document the function of a code point, whether they are used in any rule/class or not. | XML: remove any unused **named** classes, if any. |
| XML: Comments on <class> element: | There are no comments on the class elements. | please use this comment field to describe the contents of each *named* class. |
| Rule names | Please use lowercase for initial letter in rule names to match LGR-3 conventions | pls fix. |
| **Detailed editing** | Copy of XML included with suggested changes. | Please compare to the version submitted for feedback and note suggested changes, review and use as basis for further edits. |

## Test files

Details of test file evaluation have been submitted separately to the GP and confirmation has been received that results conform to expectations.

Here are some more detailed results:

I am sorry, I had submitted test label files with the errors as righty mentioned below. However I had later realised my mistake and thought I had sent the updated version of the test lable files. Its only today I found that they were not delivered, they were still in draft mode. I fully agree with the errors pointed out.

For Gurmukhi (testing for dispositions as valid/invalid)

* The  ExcludeCode file contains only invalid labels due to out-of-repertoire (or even "not NFC"). Check.
* The Invalid Label file contains two labels that against expectation resolve as valid:

#Rule 1: N must be preceded only by C1  
Validation: ‎ਤੁਤ‎ (0A24 0A41 0A24) : valid

There is no N in this label - error in test file? Already mentioned above. This is error in test file and the updated test file was not sent.

#Rule 6: A must be preceded by C, N or specific  V or M and followed by C3 or CN1  
Validation: ‎ਐੱਥ‎ (0A10 0A71 0A25) : valid

The context here is C A C3 so that label should be valid - error in test file?

Not all labels report invalid due to the condition predicted in the header comments, but that's probably because some are invalid for more than one reason and my tool only reports the first one encountered.

* The Valid label file contains 3 labels that are not valid because they are not in NFC (assuming that the library I use implements NFC correctly)

Label ਬਾਜ਼ਾਰ (0A2C 0A3E 0A5B 0A3E 0A30) not normalized to NFC (Skipped): (Line 113)  
Label ਸ਼ਿੰਗਾਰੇ (0A36 0A3F 0A70 0A17 0A3E 0A30 0A47) not normalized to NFC (Skipped): (Line 148)  
Label ਸ਼ਰਮਾ (0A36 0A30 0A2E 0A3E) not normalized to NFC (Skipped): (Line 174)

* the Crosscript test file contains one repeated label, all others report at least one out-of-repertoire variant as expected (variant testing to follow)