L2/03-351

JTC1/SC2/WG2 N2638

Title:	Comments on N2621, Proposal to Encode Tibetan BrdaRten Characters
Doc. Type:	Expert contribution
Source:	**UTC / L2 position**
Date:	October 1, 2003
Action:	For consideration by JTC1/SC2/WG2
References:	WG2/N2621, N2558 (= L2/02-455), N2624 (= L2/03-315), N2625 (= L2/03-316), L2/03-002r
Distribution:	WG2 members, UTC members

Background

In document N2558, the Chinese national body proposed the addition of 956 characters for Tibetan pre-composed "stacks". This proposal has been re-submitted in document N2621 with only very minor changes.

The basis for proposing pre-composed Tibetan characters in N2558, rather than relying on the currently existing UCS characters and the dynamic-composition encoding model, was presented as follows:[1]

> However, because of technical reasons, this encoding scheme is not compatible with traditional education, publication and electronic desktop publishing systems. It seems still quite difficult to properly solve the problems with Tibetan information interchange and processing.
>
> 1) The biggest difficulty for Tibetan information processing is the vertical composition of Tibetan characters…
>
> 2) In the implementation level 1 of ISO/IEC 10646, one code corresponds one character but one Tibetan BrdaRten character needs several codes to represent with very length which is a big block to the implementation of Tibetan system. In practical applications, the bilingual processing such as Tibetan-Chinese or Tibetan-English at the same level of implementation is an underlying requirement of Tibetan users.

Further justification given was that existing implementations use character encodings based on pre-composed forms.

The revised proposal, N2621, does not present any new grounds for encoding these pre-composed forms.

Prior critique of the case for encoding Tibetan pre-composed forms

In document L2/03-002r, Andrew West provided comments on N2558, observing that the proposed inventory of pre-composed forms would not cover all of the Tibetan "stacks" that are, in fact, attested in usage. As a result, the need for implementations to support the

[1] N2558, pp. 1–2.

existing model using a limited character repertoire and dynamic composition of vertical stacks would remain. West further points out the problems that would result by having two ways to encode Tibetan text elements, with the inevitable result of inconsistent and mixed encoding of text in documents. These points have been reiterated in document N2624, in response to the revised proposal, N2621.

In document N2625, Steve Hartwell adds to this critique the observation that adoption of the new proposal would create additional burden for implementers, and would lead to differing implementations, some supporting one encoding model for Tibetan and some supporting the other, leading to significant obstacles to interchange resulting in considerable difficulties for users.

One of the issues alluded to but not stated explicitly in the comments by West and Hartwell has to do with string comparison and normalization: a given Tibetan text element could be encoded in two distinct ways, yet the Unicode normalization forms that are the basis for string comparison in widespread implementations would not recognize these distinct encoded representations as equivalent. This fact, combined with inconsistent encoding of data, would lead to serious problems in data management, reliability of information resources, security, and other aspects of information processing. For instance, queries into databases could give incomplete results, and different invocations of what are thought to be the same queries into a given database could give inconsistent results.

Moreover, suggesting that normalization forms be revised in a manner that would cause the different Tibetan representations to be considered equivalent could only work against the proposal for pre-composed Tibetan forms: due to requirements of existing, widely-deployed implementations, any revision to normalization forms would require that the existing representations for Tibetan be considered the normalized form. As a result, protocols that require interchanged data to be in normalized form, as is the recommendation for W3C protocols such as XML, would require Tibetan data to use the existing encoding model. Hence, all software implementations other than proprietary systems that are not expected to interoperate with other systems will need to interchange data using the existing characters and encoding model. Of course, a given product may choose to convert to a pre-composed character set and encoding model for internal processing, but that can already be done without requiring new characters in the UCS.

Comments on proposals N2558, N2621

In addition to the comments that have been made previously, I would like to comment directly on the grounds given in support of the proposed characters. The primary argument in support of pre-composed characters has to do with the “difficulty” of vertical composition of Tibetan characters. In view of the increasing number of implementations for complex scripts in recent software products, this can only be seen as a reflection on particular software implementations, and not on the existing characters and encoding model for Tibetan in the UCS. Reference was made in N2625 to implementations for dynamically-composed, vertical-stacking Tibetan being developed using the AAT font technology developed by Apple, and the OpenType font technology developed by Adobe and Microsoft; similarly, Tibetan implementations could be created today using the Graphite font technology developed by SIL.

To illustrate this point, the following screen shots demonstrate the ability to display vertically-stacked Tibetan text using existing UCS characters and the existing dynamic-composition model. The key to this is simply using advanced font technologies, such as AAT, Graphite and OpenType, that are designed to provide support for complex scripts.

Figure 1 shows UCS-encoded Tibetan text displayed without the benefit of advanced font technologies. This result would be seen if either the software or the fonts used did not support complex-rendering support for Tibetan.[2] Clearly, this is inadequate for Tibetan text; yet the limitation is a problem in the implementation used rather than the encoding. Moreover, this is a temporary problem that will be removed as adequate implementations are made available.

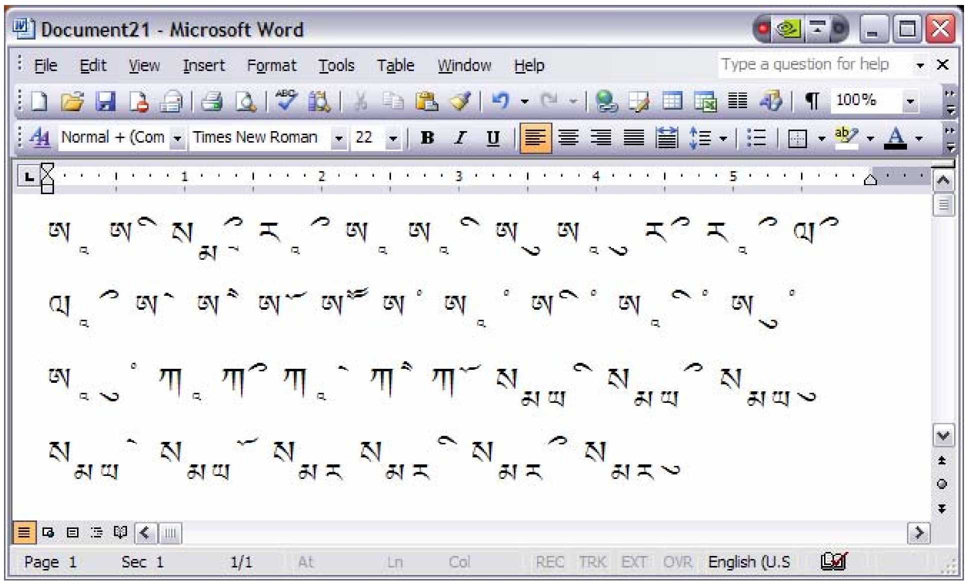
Document21 - Microsoft Word
File Edit View Insert Format Tools Table Window Help
Type a question for help
100%
Normal + (Com
Times New Roman
22
Page 1 Sec 1 1/1 At Ln Col REC TRK EXT OVR English (U.S

Figure 1. Tibetan characters displayed without Uniscribe/OpenType support

The feasibility of adequate implementations using the existing encoding for Tibetan is demonstrated in Figure 2. This screen shot shows exactly the same text as seen in the previous figure, but this displayed using an implementation that provides complex-script support for Tibetan.

[2] In this particular screen shot, the issue lies in the font: the text is formatted using Arial Unicode MS, which does not include OpenType information for Tibetan script.

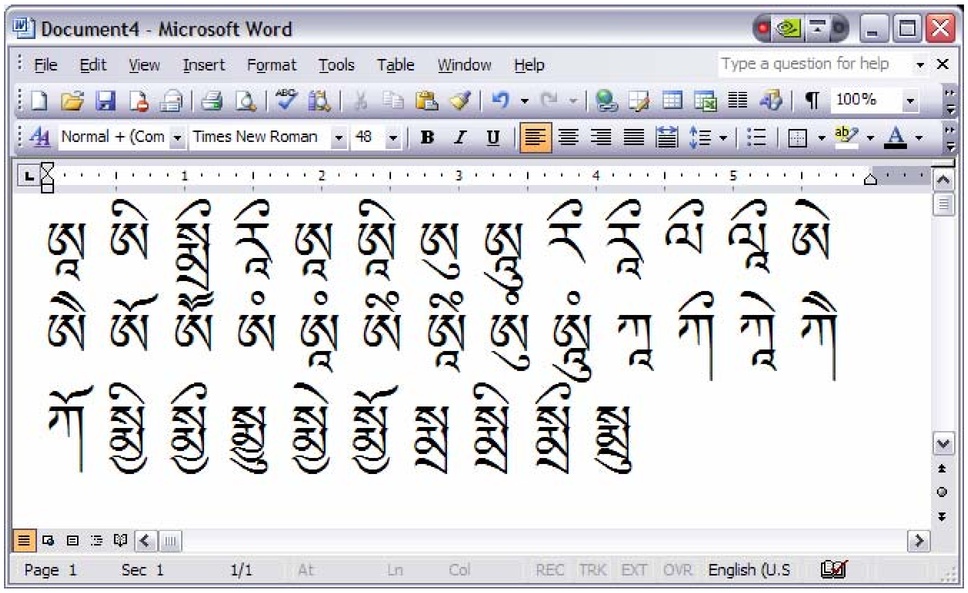
Document4 - Microsoft Word
File Edit View Insert Format Tools Table Window Help
Type a question for help
100%
Normal + (Com
Times New Roman
48
Page 1 Sec 1 1/1 At Ln Col REC TRK EXT OVR English (U.S

Figure 2. Tibetan characters displayed using Uniscribe/OpenType support

The text in Figure 2 was rendered using the existing UCS encoding and working implementations using Microsoft's *Uniscribe* complex-script engine and a Tibetan OpenType font.[3] The sample text used contains stacking combinations that are proposed as pre-composed characters in N2558 and N2621. It is clear, therefore, that the primary argument given in these proposals for encoding these characters—the problem of displaying dynamically-composed Tibetan vertical stacks—is invalid: this problem has been eliminated by advanced font technologies that are becoming widely available.

A secondary argument is given in N2558 and N2621 that text processing in which Tibetan stacks are encoded as sequences is an obstacle to implementation of Tibetan, and imposes increased complexity in bilingual processing over an encoding that represents stacks as a single character. No evidence in support of these claims is provided, nor is it clear what form any such evidence might take. The screen shots above demonstrate that encoding stacks as sequences of characters is not an obstacle to implementation in relation to display of text. In practical applications, input of Tibetan text would generally require multiple-keystroke sequences for either encoding, and any alternate input method that allowed entry of an entire stack using a single gesture could equally be implemented to generate either single-character or multi-character representations.

Moreover, there are ways in which a single-character-per-stack encoding would pose an obstacle to implementation, or at least make implementation more difficult. Consider, for instance, searching algorithms: when a user searches for a letter KA, that letter might occur in isolation, but it might also occur within a vertical stack. Thus, searching algorithms would need to decompose the single-character representations of stacks into their multi-character-sequence representations prior to doing the string comparison.

[3] This or a newer version of the *Uniscribe* engine and a Tibetan OpenType font such as this will appear in future versions of Microsoft products, and possibly as updates to existing Microsoft products.

Finally, N2558 and N2621 present a tertiary argument that existing implementations use a pre-composed encoding model. These documents do not clarify in what way these pre-existing implementations are seen to support a proposal for pre-composed encoding in the UCS. One possibility is that these implementations demonstrate feasibility or simplicity of creating implementations using a pre-composed encoding model. This is irrelevant given the demonstration of existing implementations using the dynamic-composition encoding model. The other possibility is that pre-existing implementations create a need for backward compatibility in encoding. This argument would be invalid, however: character-encoding conversion using one-to-many or many-to-one conversions is certainly possible, and the inventory proposed in N2558 and N2621 is such that there would be no ambiguity in mapping either from the legacy encoding to the UCS or vice versa. Moreover, since the legacy encodings presumably did *not* include any dynamically-composed representations, round-trip conversion of data should be possible.

Conclusion

The proposal in N2621 does not give adequate consideration to serious problems in interoperability and implementation that would result if the proposal were accepted. In addition, the primary basis for the proposal, which is an assumed obstacle to rendering sequences of characters as vertically-composed stacks, has been shown to be resolved in existing software implementations that utilize advanced font technologies that are becoming widely deployed.

Accordingly, it is recommended that the proposal presented in N2621 be rejected by WG2 and UTC.