Proposal for a TEI workgroup on the 'end of word' problem in Sanskrit

Sanskritists who work with electronic texts find themselves facing a serious problem: the way in which the language is written does not separate all words. Traditionally, scribes made no word divisions at all, but it has been standard practice since the mid-to-late 19th century to introduce them wherever they do no orthographic violence. However, the nature of the script means that many words continue to be joined together.

For the convenience of non-Sanskritists the following discussion uses Roman transliteration, but the actual issue involves the Devanāgarī script in which Sanskrit is normally written. Devanāgarī is a syllabary, in which a syllable consists of zero or more consonants followed by one vowel followed optionally by *ṃ* or *ḥ* (*anusvāra* or *visarga*).[1] If a word ends with a consonant, it therefore has to share a syllable with the next word, so that *āsīd rājā* ('there was a king') is written **ā - sī - drā - jā**. To make matters worse, sandhi (phonological change at word boundaries) may fuse two consecutive vowels together, so that, even ignoring orthography, the words can no longer be divided — for example *tathā api* ('even so') becomes *tathāpi*, where the single vowel *ā* is shared by two inseparable words. This is a feature which affects even Romanised versions of Sanskrit texts.

These usages present no problems for display, and the Unicode character block for Devanāgarī (U+0900–U+097F) is — with some marginal exceptions — capable of properly representing Sanskrit. However, for almost any kind of analysis it is necessary for the computer to 'know' where words begin and end, and it has long been clear that we need to find a solution to the problem of hidden word junctions. Various ad hoc solutions have been used by people who have typed up Sanskrit texts, but none is at all satisfactory.[2]

The solution cannot be implemented at the level of character encoding. It is sometimes suggested that the '*āsīd rājā* problem' could be solved by using a special Unicode character such as zero width space (U+200B) to indicate the hidden word junction, but this would in fact prevent the correct glyph from being formed. Even if this were not so, it is obvious that one cannot hope to use character encoding to indicate that the vowel of the syllable *thā* in *tathāpi* represents original *ā* + *a*. Clearly the solution belongs at the level of markup, and this leads one very quickly towards XML and the Text Encoding Initiative.

XML permits the problem to be solved cleanly. A tag can be defined to bracket the syllable in question, and to provide the analysed equivalent either as attribute text or as a child element:

āsī<foo bar="d rā">drā</foo>jā

or

āsī<foo>drā<bar>d rā</bar></foo>jā.[3]

We are therefore proposing that the TEI establish a workgroup to examine this issue, with the remit of devising a small tag-set for use in the encoding of Sanskrit texts. The workgroup should consider whether this tag-set should be completely Sanskrit-specific, or

---

[1] This is a slight simplification, deliberately omitting certain 'optional' features such as Vedic accents. These do not, however, affect the principle as stated here.

[2] Almost all involve the use of Roman script, which would find little favour in India. One Sanskrit text published in India (the *Jaiminīyabrāhmaṇa* edited by Raghu Vira and Lokesh Chandra) attempts a partial Devanāgarī solution, as in 'प्रजापतिर् वावेदम् अग्र एको ऽसृज्यत नान्यं द्वितीयं पश्यमानः। स ऐक्षताहं वाव प्रथमो ऽजनिष्य्, अहं श्रेष्ठो ऽस्म्य्, अस्ति स्विन् मद् इहान्याꣳ इति। स व्यैक्षत। सो ऽन्यद् आत्मनो ऽध्य् उत्तरतो ज्यायस् तिष्ठद् अपश्यत्।' (2.369). But such usage would be unlikely to be any more acceptable to Indian scholars than outright Romanisation, and it does not even solve the problem of vowel sandhi.

[3] Exactly the same issues occur within compound words, a feature which is heavily used in Sanskrit. If a scholar wished to create a text in which compounds were separated into their component members, he/she would be able to use use similar tags to do so. The name *gaṇeśa*, for example, is formed from *gaṇa-īśa*; it could be encoded as **ga<foo bar="ṇa-ī">ṇe</foo>śa** or **ga<foo>ṇe<bar>ṇa-ī</bar></foo>śa**.

whether it would be appropriate to employ a more generalised approach which could be adapted for use with any language that happened to pose similar problems. (Japanese may be a case in point.) We envisage that all discussion would be undertaken by email. Since the questions to be debated are very technical and very restricted in scope, we believe that it should be possible to complete the discussion and issue our recommendations within a period of six months.

As members of the workgroup we propose the following:

John Smith, University of Cambridge (john.smith@oriental.cam.ac.uk)
Raymond Doctor, University of Pune (dictdoc@hotmail.com)
Christian Wittern, TEI (wittern@kanji.zinbun.kyoto-u.ac.jp)
Lou Burnard, TEI (Lou.Burnard@oucs.ox.ac.uk)
Syd Bauman, TEI (Syd_Bauman@Brown.edu)
Jost Gippert, University of Frankfurt (gippert@em.uni-frankfurt.de)
Peter Schreiner, University of Zurich (peterschreiner@lycos.com)
Bob Hueckstedt, University of Virginia (rah2k@virginia.edu)
Felix Sasaki, University of Bielefeld (felix.sasaki@uni-bielefeld.de)

We would also welcome the participation of Madhav Deshpande (University of Michigan) and Yves Codet (University of Toulouse-Le Mirail), but efforts to get in touch with them have so far not succeeded.

It is worth observing that this proposal is not made in a vacuum. Work is currently under way in Cambridge and Pune (Poona), India, to develop software that will work intelligently with this type of markup. Named ***Vinayaka***, the program will allow the user to edit Sanskrit text, and to indicate the position of hidden word divisions (and compound divisions); these will be written to the output file as XML markup. Vinayaka will 'understand' enough Sanskrit to be able to suggest the correct division in the majority of cases. It will be able to handle the whole of TEI Lite, together with a small number of extra tags. It will also function as a web browser plugin, to encourage scholars to adopt TEI markup when making texts available via the Web. Vinayaka will be distributed as freeware. A specimen passage from the Sanskrit epic *Mahābhārata* has been prepared in Devanāgarī using tags of the type proposed along with (skeletal) TEI markup. It can be seen at **http://bombay.oriental.cam.ac.uk/john/ilsp/statement.html**.[4]

John D. Smith
University of Cambridge

Raymond Doctor
University of Pune

January 11, 2004

[4] ILSP was the working title for the program now named Vinayaka — the initials stood for InterLinear Sanskrit Processor.