**L2/02-401**

Suggested revisions to TUS 4.0 draft regarding Encoding Schemes
Peter Constable
2002-11-05

These comments are made in relation to the draft of Chapter 2 dated 11 Oct 02. Page number references are to the page numbers as they appear in the draft.

Suggested revisions to chapter 2, section 2.5:

1) change section heading to “Encoding Forms and Encoding Schemes”

2) Text beginning on p. 28, after figure 2.10, subsection heading “Encoding Schemes”:

   *Encoding Schemes*. Whenever character data must be *serialized* into a sequence of bytes, those resulting byte sequences must be exactly defined. Where 16- or 32-bit code units are involved, machine architectures differ in *ordering* in terms of whether the most significant byte of a code unit or the least significant byte of a code unit comes first. These alternate orderings are known as “big-endian” and “little-endian” orders, respectively. For Unicode character data, the details of serialization not only differ for each of the encoding forms but, for 16- and 32-bit encoding forms, they also depend on whether the CPU supports big-endian or little-endian integral data. Providing the detailed specification of serialization into bytes is the function of character encoding schemes.

   When a process receives character data, the details of serialization must be determined before processing begin. In general, this could be determined by negotiation between sender and receiver, or the receiver could attempt to detect this by inspecting the data received. To facilitate such detection, The Unicode Standard defines a unique code unit known as the *byte order mark* (BOM). Use of the BOM is one possible means for determining byte order. (For more on the byte order mark, see *Section 15.9, Specials*.)

   A *character encoding scheme* specifies a particular character encoding form plus the details of byte serialization, including both the byte order and the use of the BOM. Where 16- and 32-bit code units are possible, there are only two possible byte orderings of actual data. A character encoding scheme is a *specification* of byte serialization, however, not merely an attribute of actual data, and there are more than two possible specifications regarding details of byte serialization for 16- and 32-bit encoding forms.

   For UTF-8, serialized data consists merely of the UTF-8 code units (= bytes) in sequence. There is no byte-order issue, hence there is nothing more to be specified beyond the encoding form involved. Thus, there is only one encoding scheme associated with the UTF-8 encoding form. For the UTF-16 and UTF-32 encoding forms, however, byte serialization must break up the code units into two or four bytes, respectively, and so byte order is an issue. For each of these encoding forms, the Unicode Standard defines three specifications for byte serialization—that is, defines three encoding schemes. Thus, there are a total of seven Unicode encoding schemes, as shown in *Table 2-3*.

**Table 2-3. The 7 Unicode Encoding Schemes**

| Encoding Scheme | Endian Order | BOM Allowed? |
|---|---|---|
| UTF-8 | N/A* | yes* |
| UTF-16 | Not restricted (either big- or little-endian) | yes |
| UTF-16BE | Big-endian | no |
| UTF-16LE | Little-endian | no |
| UTF-32 | Not restricted (either big- or little-endian) | yes |
| UTF-32BE | Big-endian | no |
| UTF-32LE | Little-endian | no |

* Because UTF-8 code units are 8 bits in size, the usual machine issues of endian order for larger code units do not apply. The serialized order of the bytes must not depart from the order defined by the UTF-8 encoding form. Use of a BOM is neither required nor recommended, but may be encountered in contexts where UTF-8 data is converted from other encoding schemes which use a BOM.

Note that the encoding schemes named "UTF-16" and "UTF-32" do not require one particular endian order. A process that generates data in these encoding schemes may use other factors to determine which endian order to use. A process that must interpret data declared to be in either of these encoding schemes must inspect the data to determine the endian order (in the absence of other higher-level protocols).

Note that the UTF-16 and UTF-32 encoding schemes have the same labels…

3) Chapter 3, section 3.10, D38: revise definition as follows, and add the explanatory note shown:

D38 *Unicode encoding scheme*: a specification of byte serialization for a Unicode encoding form, including the specification of the handling of a byte order mark (BOM), if allowed.

- Encoding schemes are distinct from encoding forms mainly in that they deal with the issue of byte ordering where 16- or 32-bit code units are involved. Encoding schemes are *specifications* for byte serialization, not merely two different byte orders. Thus, while only two byte orders for actual data are possible, more than two specifications of byte serialization are possible.