WG14/N508
X3J11/95-109

Date: Fri, 6 Oct 1995 20:49:48 +0100
Subject: WG20 N423: Guidelines on character data types

I have a WG20 action item to ask WG14 to comment on the following paper:

Keld

SC22/WG20 N423

Guidelines on Character Data Types in Programming Language Design

The character data types support should be done at 3 levels: abstract character level, encoded character level, and text level (combining sequences). The sequence of three levels indicates how WG20 sees the importance of the data type support in the programming language design.

A set of APIs needs to be defined to transform between the 3 levels.

Abstract Character Level

The programming language standard should facilitate a character data type design that is independent of the actual encoding of characters. This abstract character level should be the main form of the national character data type in programming language as it facilitates the portability among application programs across platforms. This level corresponds to the "character" term in SC2. The specification of the encoding should be hidden and transparent to application programs, thus the encoding is implementation defined. The character is presented in exactly one integral unit, therefore the indexing on the character array is permissible.

Encoded Character Level

The programming language standard may provide a data type to support the encoded character level, where the encoding storage requirement of the abstract characters is known. This level corresponds to the "coded character" term from SC2. This form of encoding can be used to meet the explicit storage requirements, and is useful in programming with multiple coded character sets.

One multioctet data type is sufficient, where the storage requirement is not determined by the string termination delimiter defined by programming language. The actual data storage can cater for the coded character sets support by the implementation and for a set of ISO character sets based the nature of the coded charcter set such as single byte single octet (i.e. ISO 8859-1), single byte multi-octet (i.e. ISO 10646 UCS-2), multi-byte single octet (i.e. ISO 10646 UTF-8), and multi-byte multi-octet (i.e. ISO 10646 UTF-16), where the encoding and date storage may be defined by POSIX charmap definitions or by other means. Only one set of the multioctet APIs needs to be defined. For example, a data definition could be done in the following pseudo statement:

```
encoded "UCS-2" character_string = (encoded "UCS-2") "literal
```

string"

Here "encoded" is a data type, and "UCS-2" is a reference to the actual data storage requirement defined in the charmap or by the standards to be found in the file system or similar places. The list of reserved keywords on the coded character sets which are based on the ISO SC2/WG3 and ISO SC2/WG2 should be provided to ensure the minimum portability among the ISO character sets.

Once the data type with referenced storage requirement (such as þencodedþ and þUCS-2þ) is specified, the programming language standard should provide the necessary code conversion between the machine coded character set and storage coded character set.

Text Level

The programming language standard may provide the data type support for the text level, which corresponds to the "combining sequences" term from SC2. The behavior of this level is currently not so well developed, and it is difficult to advise on required functionality.

Literal

There is a need to have literal in the programming language design. In the scope of the internationalization, the literal design should be on the abstract level, i.e. the abstract character level. For example, a specification of literal "This is a literal" should be in abstract data type and the encoding of th literal string should be determined according the runtime codeset not the codeset at the complication time. This may possibly be dependent on the actual language specification to transform such literal strings into the appropriate data type accordingly. For languages not having such specific requirement, the encoded representation can be specified explicitly. For example, in pseudo programming language, (encoded "UCS-2" "This is a literal").