Standards Council of Canada

Conseil canadien des normes

SC22/WG11/N375

1993-03-04

SCC 605 (JTC1/SC22)-1

Mr. J.L. Cote
Administration Policy Branch
Treasury Board of Canada
140 O'Connor Street, 10th Fl E. Tower
Ottawa, ONCanada K1A 0R5

SUBJECT: **JTC1/SC22 COMMITTEE DRAFT NO. CD11404 N1305**

Dear Sirs:

The Canadian National Committee does not approve the committee draft for the attached reasons.

Yours truly,

Doug Langlotz, P.Eng.
International Standards Division

.../dl
Encl.

c.c. Mr. G. Warren, IBM Canada Laboratory
Mr. A. La Bonte, Gouvernement du Quebec

1200-45 O'Connor, Ottawa, K1P 6N7 Tel.: (613)238-3222 Telex: (Stancan Ott) 053-4403 Fax: (613)995-4564

CANADIAN COMMENTS ON DOCUMENT NO. JTC1/SC22 **CD11404 N1305**

- -

Canada does not approve the above document for the following reasons:

OBJECTION TO DISPOSITION OF CANADIAN COMMENT

Document SC22/N1304 (disposition of comments) rejects Canadian proposal to add a datatype for handling character string comparisons in saying:

> "The Working Group noted that this is a proposal for an entirely new datatype with which no language or application group has experience, except apparently for some experimentation in Canada"

We vigorously disagree: the way to handle character string comparisons is being described in the POSIX standard (particularly part 2) produced by SC22. It is widely known for those who studied the question more than in a shallow way that there is no other way to fully answer the ordering of character strings than by decomposing them in successive passes of comparison. So far that capability exists in POSIX, is implemented in real-life systems (example: IBM AIX) and has been proven to work. There should be an abstract datatype to describe the decomposition of characters for the purpose of achieving multi-pass ordering or one-pass multilevel ordering in this fashion. This is not a mere experiment limited to Canada. It is true that it all that began in Canada, and was implemented before a standard was published (CSA Z243.4.1), but other national standards are emerging in Sweden and Denmark (that we are aware of) and CEN is also working on a similar approach, to which there is no abstract datatype we can refer to to at least describe the process. POSIX Part 2 is on its way to be published as IS (now at the DIS ballot stage) and it is the model it is using.

> "The Working Group also notes that the proposal appears to deal primarily with the semantics and representation of character sets, which is outside the scope of CD 11404".

The Canadian proposal deals with ordering, or with the ordering process, and CD 11404 deals with ordering all the time when it describes the property "Ordered". We all know that for data processing to be successful, character string ordering is essential. We also know that current programming languages have failed to do that correctly in a way acceptable for humans. CD 11404 is a good opportunity to correct that situation in defining an abstract data type, an envelope whose actual content will vary, of course, from culture to culture, but for which there is a current attempt at defining a tailorable world template for all scripts of the world. So far, for this project that SC22/WG20 will be handling, there is no abstract way to define that kind of object suitable for programming languages implementation. POSIX LC_COLLATE is such a datatype, but there is no language-independent way to describe it so that it can be used by languages that have to order characters in a series of rules.

> "If the result of work on this proposal by international standardization activities relating to character sets is the identification of one or more useful datatypes, then it becomes appropriate for those datatypes to be included in the next revision of Langauge-Independent Datatypes".

As written earlier, LC_COLLATE data should be at least described by an abstract datatype right now. While WG11 has accepted the Canadian comment that Çharacter type is intrinsically unordered and that the text has been modified accordingly for that type, it also says that ordering is an application-defined extension to the type semantics: Canada believes it is not sufficient to say that because the results of ordering will be fuzzy and will not work across languages, the main reason why Language-Independent Datatypes should be useful as a tool and international reference. In a given culture, there exists a low level character string ordering under which ordering is not recognized and over which application- ordering can be built upon. All is a matter of level decomposition as POSIX LC_COLLATE allows to define outside of applications. However the application should be able to use these multilevel definitions by means of an abstract datatype to be bound with operating systems definitions (POSIX).

CONTRADICTION

While stating that the Character datatype is unordered, CD 11404 states in page 52 that Characterstring is a family of datatypes which represents strings of symbols from standard character sets. Canada strongly objects to the property "ordered" of this datatype, as well as it did for the character datatype. This is a disguised way to bypass the unordered property of the Character datatype. It is a way to propagate lousy inefficiency of applications to order textual elements properly for end-users.

There is a warning note after the description of this datatype that says:

> "There is no international standard for collating sequences, although certain international character-set standards require specific collating sequences. Applications which need the ordering on characterstring, and which share a character- set for which there is no standard collating sequence, need to create a defined datatype or a repertoire-identifier which refers to the character-set and the agreed-upon collating sequence".

There is an international standard in process to be adopted)POSIX.2) that describes, at the operating system level, how, systemwise, ordering of character strings should behave. It ought not be up to the application to define that but to rely on this systemwide facility. Now to achieve that, there must be a language-independent binding to POSIX or POSIX-like LC_COLLATE facility. An abstract datatype, composed of a set (multilevel, or multipass) of binary strings associated with Characterstring, is mandatory to achieve that goal. This should not be left to the application responsibility. Ordering has to be consistent across applications, and across languages.

For operations handling character strings, one should also refer to the White Paper published by SHARE EUROPE (Headquarters: 48, route des Acacias, CH-1227 Carouge/Genève, Switzerland) in 1990 on National Language Architecture, with which POSIX is in line, that describes how can be handled the multilevel data structure associated with any character string at the lowest character string comparison level.

As a matter of fact, no character set requires specific collating sequence, as suggested by WG11. End-users, humans, do, and application-to-application communication also requires consistency of operation, a goal that can't be achieved if a note like the above-mentioned one in the CD is as fuzzy.

OTHER OBVIOUS MISTAKES

Annex A specifies that the list defines character-sets and collating sequences. This is totally wrong that any character set defines collating sequence.

Canada urges WG11 not to spread such rumours. SC2 always objected to deal with semantics of characters and even more with collating sequences.

VOTE REVERSAL POSSIBILITY

The Canadian National Body is ready to reverse its vote if the following conditions are described in CD 11404 for character string processing to have a chance to work efficiently and properly:

1) Character string datatype is defined as intrinsically unordered whatever the standard character set used;

and

2) The CD should be updated so that it clearly states that if ordering is to be achieved, the mechanism shall be via an associated datatype whose content is filled at the operating system level. That datatype structure should be described to consist of items as noted in 3) below.

3) A series of n bit strings (corresponding to the n-level comparison that must be done to achieve a comparison in a fully-predictable way) filled by an operating system function or external function (equivalent functionally to POSIX LC_COLLATE) on which ordering is to be done with various possibilities according to the language needs/power: equality, less-than, greater-than [and eventually equivalences (like "abc" equivalent to "ABC" if needed in case of inequality, or "Résumé" equivalent to "RESUME" and so on), and fuzzy matching; it would be sufficient to describe property "Ordered" for this datatype, if one wants to avoid to deal with all the kinds of operation types envisaged, say, in SHARE EUROPE Requirements]. The number of levels should be variable like the number of bits for each distinct level (variable also), like for the variable number of characters already possible in the presently described Characterstring data type.