1. Identification of the proposed change

1.1. Title

REPLACE library function

1.2. MDC Proposer and Sponsor

This proposal originates from Ed de Moel.
Motions regarding the status of this document will be made by Taskgroup 2 (String Handling) of Subcommittee 13 (Data Management and Manipulation).
Ed de Moel can be reached at:

· 3950 Mahaila Avenue, apartment K12, San Diego, California 92122
· home phone: 619 455 7107
· office phone: 619 535 7566
· telefax: 619 535 7627
· email: demoel@fwva.saic.com

1.3. Motion

No motion; final MDC Type A write-up.

1.4. History of MDC actions

Date	Document	Action
October 1995	X11/95-112	Final MDC Type A write-up.
June 1995	X11/SC13/95-5	Document reformatted. Presented for elevation to MDC Type A. Passed 25:3:10.
January 1995	X11/SC13/94-42	No modifications. Presented for elevation to MDC Type A. Reformatting suggested, not presented for elevation.
June 1994	X11/SC13/94-30	No modifications. Presented for elevation to SC#13 Type A. Passed 21:0:4.
February 1994	X11/SC13/94-18	Editorial inconsistencies corrected. Presented to replace SC#13 Type B. Accepted 18:0:3.
February 1994	X11/SC13/94-3	Document brought in the format defined after the October meeting. Presented for elevation to MDC Type A. Editorial inconsistencies found, document not voted on.
October 1993	X11/SC13/93-54	Presented for promotion to MDC Type A. Vote postponed until document has been brought in concordance with the "format for library proposals" that is to be defined at a later time during this meeting.
June 1993	X11/SC13/93-34	Presented for promotion to SC#13 Type A. Accepted 13:3:5. Only con raised: libraries not yet formalized.
February 1993	X11/SC13/93-6	History section corrected. Submitted for promotion to SC#13 Type A. Not voted on, in order to keep PRODUCE and REPLACE at the same level.
October 1992	X11/SC13/92-64	Corrections applied as discussed in taskgroup; accepted as a replacement for a SC#13 Type B.
October 1992	X11/SC13/92-38	Presented for promotion to SC#13 Type A, but modified in taskgroup.
June 1992	X11/SC13/92-14	Corrections applied as discussed and passed during previous meeting. Proposal not accepted as SC#13 Type A.
February 1992	X11/SC13/91-5	History section put in correct order. Syntax changed to be a library

		function per subcommittee guidance. Presented for promotion to SC#13 Type A.
October 1991	X11/SC8/91-10	MDC#52 (St. Louis) Proposal modified to work without exceeding the portable subscript limitations. Substantive change, accepted as a Type B proposal of SC#13.
January 1990	X11/SC8/89-18	MDC#44 (San Francisco), MDC#44 (San Francisco), updated proposal for MDC, subcommittee #8. Proposal accepted as a type A proposal of subcommittee #8 with the amendment that the MUMPS-code should be corrected for the possibility that $ORDER yields an index-value for which $DATA equals 10.
October 1989	X11/SC8/89-2	MDC#43 (Andover), formal proposal for MDC, subcommittee #8. Proposal accepted as a type B proposal of subcommittee #8.
June 1989	...	MDCC-E#24 (Barcelona), extrinsic function approach approved. Proposal to be split up into two proposals to be submitted to MDC SC#8 in the document-format for that subcommittee.
March 1989	...	MDCC-E#23 (Frankfurt), re-evaluation of current proposal. Two separate functionalities found for proposed new function. Slight preference for extrinsic versus intrinsic approach.
November 1988		X11/SC1/89-30MDC#40 (Washington DC), formal proposal
February 1988	...	MDCCE#20 (Utrecht), ordering of diacritical signs resolved, proposal made simpler
January 1988	X11/SC1/88-20	MDC#37 (San Diego), first attempt for a formal proposal, problems with ordering of diacritical signs
November 1987		...MDCCE#19 (Vienna), further discussion, creation European task-group
November 1987		...MDC#36a (Washington DC), separate initial proposals by Richard Walters (MDC) and Alfons Puig (MDCC-E)

1.5. Dependencies

MUMPS Library Specification.

2. Justification of Proposed Change

2.1. Needs

While dealing with other languages than English, the user of a computer encounters discrepancies between the schemes for:

- encoding characters
- collating (sorting) strings
- printing text

These discrepancies have been extensively described in X11/SC1/88-20.
These schemes bear a stronger relation to the (human) language and equipment used, than to the programming language. However, the user of the MUMPS programming language needs a tool for the conversion of strings from one representation into the other in order to be able to use them for the various purposes.

2.2. Existing Practice in Area of the Proposed Change

Currently no standardized technique for the required functionality exists.

3. Description of the proposed change

3.1. General Description of the Proposed Change

This proposal is one of a series defining two new library-functions to be used for the replacement of substrings within strings. A difference with the currently available intrinsic function $TRANSLATE is that these functions will be able to change "one or more characters" into "zero or more characters", whereas $TRANSLATE is only able to change "one character" into "zero or one character". The "translation-specification" can be of arbitrary length and is stored in an array rather than in one single value.

3.2. Annotated Examples of Use

See the PRODUCE proposal.

3.3. Formalization

In the MUMPS Library Specification, add the definition of the REPLACE library function (at the time that this document is written, no document number or section numbering is known for the MUMPS Library Specification).

3.3.1. Library Element Description

String handling function; substring replacement.

The function scans a string for the occurrence of certain substrings and replace all such occurrences by another substring. This process is repeated until none of these substrings can be found anymore, yet no character in the input-string is replaced more than once. Replaced characters are not affected. The resulting string will be passed back to the caller as the function-value.

The function has two required parameters, a string-value and a translation-specification array.
The function converts the value of `IN`, according to the specification in `SPEC`.
The function scans a string for the occurrence of certain substrings and replace all such occurrences by another substring. This proces is repeated until none of these substrings can be found anymore, but in such a way that no character in the input-string is translated more than once. The resulting string will be passed back to the caller as the function-value.

For the purpose of this discussion, the string-value will be called `IN` and the translation-specifications will be called `SPEC(I,1)=FIND`,`SPEC(I,2)=OUT`, `FIND` being a substring to be located and `OUT` being the substring to be put in its place.

For each element of the form `SPEC(I,1)=FIND`, the function will scan whether `IN` contains the substring `FIND`. If such a substring occurs, and none of the characters in that substring has been translated because it was part of another substring to be translated, it is replaced by `OUT`, which is the value of `SPEC(I,2)`. After the replacement has been made, `IN` is scanned again for the occurrence of `FIND`. This process continues until the substring is no

longer found with no characters marked as being used before. After that, the next translation-specification is tried.

NOTE: The array `SPEC` may contain overlapping find-strings, e.g. `SPEC(1,1)="ABCDE"` and `SPEC(2,1)="ABC"`. Since the array `SPEC` is scanned using $ORDER on the first subscript, the longer substring will be replaced, before the shorter one is attempted. If the opposite behaviour is required, the order of the values in `SPEC` should be reversed: `SPEC(1,1)="ABC"` and `SPEC(2,1)="ABCDE"`.

Since any translation may cause a substring to be translated to be inserted again, the above process will not translate any character from the input-string more than once, and will not translate any character that is inserted as the result of a translation.

3.3.2. Definition

`REPLACE^STRING(IN,.SPEC)`

This function is computationally equivalent to:

```
REPLACE(IN,SPEC) ;
      NEW L,MASK,K,I,LT,F,VALUE
      SET L=$LENGTH(IN),MASK=$JUSTIFY("",L)
      SET I="" FOR  SET I=$ORDER(SPEC(I)) QUIT:I=""  DO
      . QUIT:'($DATA(SPEC(I,1))#2)
      . QUIT:SPEC(I,1)=""
      . QUIT:'($DATA(SPEC(I,2))#2)
      . SET LT=$LENGTH(SPEC(I,1))
      . SET F=0 FOR  SET F=$FIND(IN,SPEC(I,1),F) QUIT:F<1  DO
      . . QUIT:$E(MASK,F-LT,F-1)["X"
      . . SET VALUE(F-LT)=SPEC(I,2)
      . . SET $EXTRACT(MASK,F-LT,F-1)=$TRANSLATE($JUSTIFY("",LT)," ","X")
      . . QUIT
      . QUIT
      SET VALUE="" FOR K=1:1:L DO
      . IF $EXTRACT(MASK,K)=" " SET VALUE=VALUE_$EXTRACT(IN,K) QUIT
      . SET:$DATA(VALUE(K)) VALUE=VALUE_VALUE(K)
      . QUIT
      QUIT VALUE
```

3.3.3. Domain

Subscripts in the array SPEC must conform to the portability requirement on subscripts.

3.3.4. Range

Standard.

3.3.5. Side Effects

None.

3.3.6. MUMPS code to implement

See 3.3.2.

4. Implementation impacts

4.1. Impact on Existing User Practices and Investments

Since no standardized technique for this purpose currently exists, no upward compatibility issues exist.

4.2. Impact on Existing Vendor Practices and Investments

None.

4.3. Techniques and Costs for Compliance Verification

Create the following program:

```
KILL SPEC
SET SPEC(1,1)="aa",SPEC(1,2)="a"
SET SPEC(2,1)="pqr",SPEC(2,2)="alabama"
SET SPEC(3,1)="b",SPEC(3,2)=""
SET X="aaaaaaapqraaaaaaa"
WRITE !,$%REPLACE^STRING(X,.SPEC)
QUIT
```

An implementation that implements the function correctly would print `"aaaaalabamaaaaa"`.

4.4. Legal considerations

None.

5. Closely related standards activities

5.1. Other X11 Proposals (Type A or Type B) Under Consideration

The functions PRODUCE and REPLACE are very strongly related. Their functionality is sufficiently different to make them different functions, though.

5.2. Other Related Standards Efforts

None.

5.3. Recommendations for Co-ordinating Liaison

None.

6. List of Associated Documents

X11/SC1/88-20: Natural language handling.

7. Issues, Pros and Cons, and Discussion

7.1. June 1993, Washington DC

The only con raised was that the Library Specification Document is not yet available.

7.2. October 1993, Dublin Ireland
Formal vote postponed.

7.3. February 1994, Houston Texas
No cons raised.

7.4. June 1994, Reno Nevada
No cons raised.

7.5. January 1995, Albuquerque New Mexico
No cons raised. Formalism re-organized. Descriptive text moved to 3.3.1; M[UMPS] code moved to 3.3.2. Changed all occurrences of $& to $%.

7.6. June 1995, Chicago Illinois
Pro: 1. Useful functionality; 2. General solution to many problems; 3. Almost no cost to implementors.
Con: 1: Rarely required, best done by application [4]; 2. Poor cost/benefit for community [1]