X11/96-74

MUMPS Development Committee

Extension to the MDC Standard
Type A Release of the MUMPS Development Committee

Operator Overrides

October, 1996

Produced by the MDC Subcommittee #12
Environment

Ed de Moel, Chairman
MUMPS Development Committee

Larry Ruh, Chairman
Subcommittee #12

The reader is hereby notified that the following MDC specification has been approved by the MUMPS Development Committee but that it may be a partial specification which relies on information appearing in many parts of the MDC Standard. This specification is dynamic in nature, and the changes reflected by this approved change may not correspond with the latest specification available.

Because of the evolutionary nature of MDC specifications, the reader is further reminded that changes are likely to occur in the specification released herein prior to a complete republication of the MDC Standard.

1. IDENTIFICATION

1.1 Title

Operator Overrides

1.2 MDC proposer and sponsor

Proposer: SC12/TG2
Author: Jon Diamond, Cap Gemini UK

1.3 Motion

Proposed as MDC Type A superseding X11/SC12/96-4

1.4 History of MDC actions

Date	*Doc#*	*Action*
Oct 96	X11/SC12/96-10	Approved as MDC Type A (25:0:3) with editorial amendment
Mar 96	X11/SC12/96-4	Approved as SC12 Type A (8::3)
Dec 95	X11/SC12/96-1	Discussed at MDCC-E
Oct 95	X11/SC12/95-13	Proposed as MDC Type A - remanded to SC
Jun 95	X11/SC12/95-2	Remanded to SC (16:1) - error in COMPARE example code to implement
Jan 95	X11/SC12/94-9	Approved as SC12 Type A (5:0:1).
Nov 94	X11/SC12/94-9	Approved by MDCC-E (7:0:0)
June 94	X11/SC12/TG2/94-4	Approved as SC12 Type B
	This document also supercedes X11/SC1/TG2/91-?? Comparison Functions.	
August 93	X11/SC12/TG2/93-6	Proposed as SC12 Type B
June 93	X11/SC12/TG2/92-35	Proposed as SC12 Type B
Oct 92	X11/SC12/92-25	Reapproved as SC12 Type B
Jun 92	X11/SC12/92-12	Approved as SC12 Type B
Jun 92	X11/SC12/TG2/92-4	Approved as SC12 Type B with amendments.
Nov 91	X11/SC12/TG2/91-??	Comparison functions discussion Document
Oct 91	X11/SC1/TG1/91-34	Comparison functions approved as SC12 Type C
	Con: Some common functionality with]] (votes), Roughly ... is ambiguous (), Should include "sounds like" (), Use of [,^global] is objectionable ().	

For other history see Alternate Collating Sequence Proposal.

1.5 Document Dependencies

None

2. JUSTIFICATION

2.1 Needs

The major need is to be able to provide functionality which overrides the default behaviour of operators - specifically in terms of collation and pattern match. *See X11/SC1/TG1/91-1 for further information.*

2.2 Existing practice in the area of the proposed change

There are no known implementations that provide specific overrides for operator behaviour in these areas.

3. DESCRIPTION

3.1 General description of the proposed change

The proposal is for a number of library functions for each of the operators which need to have overrides. The first two are for the comparison of two strings according to collation either of a specific character set or of a specific global.

3.2 Annotated examples of use

a) Comparing two strings for sorts after in the collating sequence of a specific global:

```
$%COMPARE(A,B,"^GLOBAL")
```

b) As for a) except that the order is of a particular character set profile

```
$%COMPARE(A,B,"GERMAN")
```

c) As for a) except that the order is of a particular character set profile

```
$%COMPARE(A,B,"ISO:8859-1/USA")
```

d) Produce an index file for a subtree in another order (German/library sorting order etc.)

```
SORT(A,B,ORDER)
        n a,b,c
        s a="",c=$g(ORDER)
        f b=1:1 s a=$o(@A@(a)) q:a=""  s @B@($%COLLATE(a,c),b)=a
        q
```

could be used by

```
        d SORT("^A","B","GERMAN")
        s x="B("""")" f  s x=$q(@x) q:x=""  d
        . s a=@x
        . ; now do some processing for node ^A(a)
```

to process ^A in German order

3.3 Formalization

3.3.1.1 Library Element Description

String collation value function

3.3.1.2 Definition

COLLATE^CHARACTER(A,CHARMOD::O)

This function returns the collation value of a string according to the specification of the collation algorithm.

If CHARMOD is a Character Set Profile then the collation algorithm used is that specified in ^$CHARACTER for the profile

If CHARMOD is a global name then the collation algorithm used is that specified in ^$GLOBAL for that name

If CHARMOD is not specified, or the node specified above does not exist, then the collation algorithm used is the default process collating algorithm.

3.3.1.3 Domain

CHARMOD is either a Character Set Profile specification in the form charset or a global name specification in the form ^name.

3.3.1.4 Range

STANDARD

3.3.1.5 Side Effects

NONE

3.3.1.6 Example of MUMPS code to implement

```
COLLATE(A,CHARMOD)
 n x s x=""
 i $g(CHARMOD)'="" d
 . i $E(CHARMOD,1)="^" d
 . . s x=$e(CHARMOD,2,$l(CHARMOD))
 . . i x'="" s x=$g(^$GLOBAL(x,"COLLATE"))
 . i x="" s x=$g(^$CHARACTER(CHARMOD,"COLLATE"))
 i x="" s x=^$JOB($J,"COLLATE")
 s x=@(x_"("_A_")")
 q x
```

3.3.2.1 Library Element Description

String comparison function

3.3.2.2 Definition

COMPARE^CHARACTER:INTEGER(A,B,CHARMOD::O)

This function compares two strings according to the specification of the collation algorithm.

If CHARMOD is a Character Set Profile then the two strings are compared using the collation algorithm specified in ^$CHARACTER for the profile

If CHARMOD is a global name then the two strings are compared using the collation algorithm specified in ^$GLOBAL for that name

If CHARMOD is not specified, or the node specified above does not exist, then the two strings are compared using the default process collating algorithm.

3.3.2.3 Domain

CHARMOD is either a Character Set Profile specification in the form charset or a global name specification in the form ^name.

3.3.2.4 Range

-1 = A compares before B
0 = A compares the same as B
1 = A compares after B

3.3.2.5 Side Effects

NONE

3.3.2.6 Example of MUMPS code to implement

```
COMPARE(A,B,CHARMOD)
 n x,y
 ; assume current collation, i.e. ]], if no CHARMOD specified
 i $g(CHARMOD)="" q $s(A=B:0,A]]B:1,1:-1)
 ; otherwise need to override it and do string compare
 ; on collation value
 s x=$%COLLATE(A,CHARMOD),y=$%COLLATE(B,CHARMOD)
 q $s(x=y:0,x]y:1,1:-1)
```

4. IMPLEMENTATION IMPACTS

4.1 Impact on existing user practices and investments

None

4.2 Impact on existing vendor practices and investments

None for implementors, except for library code installation

4.3 Techniques and cost for compliance verifications

No significant costs are anticipated

4.4 Legal considerations

None

5. CLOSELY RELATED STANDARDS ACTIVITIES

5.1 Other X11 proposals

None

5.2 Other Related Standards Effort

SC22/WG20 activities.

5.3 Recommendations for coordinating liaison

None

6. ASSOCIATED DOCUMENTS

None

7. DISCUSSION

Oct 96l
Pro: Useful functionality for international applications

Mar 96
Clarification that environment not included for ^name. More examples requested.
Pro
?
Con
Should allow environment specification

Oct 95
Proposal not in correct format
Subsequently modified to add clarification for MDCC-E issues below

Sep 95 MDCC-E meeting
Cases to be clarified
1) charmod = ^ 2) $e(charmod,2,255) does not exist in ^$G 3) ^$C(charmod) does not exist

Jun 95
COMPARE example uses]] when should be]

Jan 95
Pros: Makes it easier to write international applications, Allows sorting English for application specific requirements
Cons: -

June 94
No Cons - error in code COMPARE corrected.

August 93
Minor corrections

June 93
Corrections as discussed.

Oct 92
Con: Potential conflict in vertical bar format. Too complex. Conflicts with MUMPS/VM usage for UCI and volume group.

Jun 92 *X11/SC12/92-12* *Approved as SC12 Type B*
Modifier changed to follow operator and also to be second field in | |, leaving environment as the first item in order not to conflict with possible later use by SC14.
Con: Too complex.