MUMPS Development Committee

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

Two Character Operators

June 1993
Produced by the MDC Subcommittee #13
Data Management and Manipulation

Thomas C. Salander, Chairman
MUMPS Development Committee

Russell White IV, Chairman
Subcommittee #13

The reader is hereby notified that the following language 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 MUMPS specifications. 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 MUMPS specifications, the reader is further reminded that changes are likely to occur in the specification released herein prior to a complete republication of MUMPS specifications.

1. Identification of the proposed change

1.1 Title Two Character Operators

1.2 MDC proposer/sponsor

Proposer:
Bo Midberg
University Hospital of Lund
Oncological Centre
S-221 85 Lund
Sweden
Tel: +46 46 177553
Fax: +46 46 188143

Sponsor:
Gretchen Bradfieid
(UC Davis)
5319 Colvin Court
Fair Oaks, CA 95628
USA
bradfiel@cs.ucdavis.edu OR
bradfield,gretchen@forum.va.gov
Tel: (916) 966-1050

1.3 History of MDC actions

Jun 93 X11/SC13/93-21 Passed as MDC Type A [29:10:5]

PROS:	Votes	CONS	Votes
1) More intuitive	6	1) Poss conflict w/SET incremental	1
2) Found in other languages	3	2) Poss conflict w/SET positional	4
3) Symmetry to operator	2	3) Functionality is already available	10
4) No conflict w/ SET incremental	0	4) Introduction of 3-char operator	2
5) More efficient	0	5) ']]=questionable desirability	2
6) Adds functionality	0	6) '>=questionable desirability	4
7) User-friendly	2		
8) ']]= desirable	0		
9) '>= desirable	0		

Mar 93 X11/SC13/93-21 Rev to RMDS 5, Submitted for elevation to MDC Type A

Feb 93 X11/SC13/92-66 Elevated to Subcommittee Type A [13:4:3]

PROS:	Votes	CONS	Votes
1) More intuitive	3	1) Conflict w/SET incremental	2
2) Found in other languages	2	2) Conflict w/SET positional	2

3) Symmetry to operator	0	3) Functionality is already available	3
4) No conflict w/ SET incremental	0	4) Introduction of 3-char operator	1
5) More efficient	0	5) ']]= (?)	0
6) Adds functionality	2		
7) User-friendly	3		
8) ']]= (!)			

Resolution of Cons:
1&2) [2@] There is still a possibility of conflict with these two proposals, but neither are ready for elevation to MDC Type A
3) [3] It was pointed out with that the only way to get 'sorts equal to' is with the condition sorts after AND not sorts after (}]&']]) which is a bit cumbersome.
4) [1] Agreed that this would be our first 3-char operator, but vendors seem to consider implementation of such to be trivial compared to the benefits

Nov 92 X11/SC13/92-66 Submitted for elevation to SC Type A in February

Oct 92 X11/SC13/92-30 Elevated to Subcommittee Type B [9:6:2]

PROS:	Votes	CONS:	Votes
1. More intuitive	4	1. Conflict w/ SET incremental	2
2. Found in other languages	3	2. Conflict w/ SET positional	1
3. Symmetry to operator	3	3. Functionality is already available	7
4. No conflict w/ SET incremental	2	4. Introduction of 3-char operator	4
5. More efficient	3		

Resolution of Cons:
3) [7] Growing support for this con.
4) [4] Sorts after could be excluded

Jun 92 X11/SC13/91-2 Amended to include Sorts After (]]=) [22:0:3]
Elevated to Subcommittee Type C [15:3]

PROS:	CONS:	Votes
1. More intuitive	1. Functionality is already available	3
2. Fixes some syntax errors	2. Conflict w/ SET Incremental	1
3. Found in many other languages	3. May conflict with other uses	3
4, Symmetry to operators	4. Conflict w/ SET Positional	2

5. No conflict w/ SET Incremental		

Resolution of Cons:
1) [3] Majority doesn't seem to care that it is redundant
2) [1] True, there is a conflict, but only if this proposal (out of three) is accepted
3) [3] Always possible, but without specifics, this con cannot be addressed
4) [2] There is minimal conflict here because these operators are relational while the SET POSITIONAL is an assignment operator.

Oct 91	X11/SC13/91-2	Initial proposal received by MDC from Sweden

2. Justification of proposed change

2.1 Needs

In complex condition constructs it becomes difficult for human thinking to understand and interpret the meaning of the syntax. Condition constructs should be formulated more according to how our brain works. This addition of operators would simply the condition syntax. It will also result in a productivity gain while writing condition constructs.

2.2 Existing practice in the area of the proposed change

Currently, the method is to use the apostrophe (') before the operators “>” "<” and "]", or to use two condition statements (one with the equal operator and one with one of the other operators). For example, in order to test for *A equal to or greater than B*, one can write

A'<B *[A is not less than B]*

or one can write

A=B!(A>B) *[A is equal to B or A is greater than B]*

3. Description of the proposed change

3.1 General description of the proposed change

Add the operators:

>= (greater than or equal to)
<=(less than or equal to)
]= (follows or is equal to)
]]= (sorts after or is equal to)

to the repertoire of relational operators

3.2 Annotated examples of use

```
SET A="ABC"
SET B="XYZ"
I A] B WRITE "TRUE" [Should not write anything]
I A] "ABC" WRITE "TRUE"    [Should not write anything]
I A]="ABC" WRITE "TRUE"  [Should write TRUE]
```

3.3 Formalization [Note changes refer to X11/TG6/93-1 RMDS Version 5)

Add to 4.2.2.1 Relational Operator relation

>=
< =
]=
]]=

Add the underlined to 4.2.2.2 Numeric relations to read as follows:

The inequalities <, ~~and~~ >, <= and >= operate on the numeric interpretations of their operators; they denote the conventional algebraic greater than, ~~and~~ less than, greater than or equal to and less than or equal to.

Add]= and]] = to the list of relations in 4.2.2.3 String relations:

The relations = [ ]]= and]]= do not ...

Add the following to the end of 4.2.2.3 String relations:

The relation]= is called *fol1ows or equal to*. A]=b is true if and only if A follows B as defined above or A is identical to B.

The relation]]= is called *sorts after or equal to*. A]]=b is true if and only if A sorts after B as defined above or A is identical to B.

4. Implementation impacts

4.1 Impact on existing user practices and investments

This proposal will not obsolete any current code nor is it backward incompatible.

4.2 Impact on existing vendor practices and investments

Modest change to parsing of relational operators.

4.3 Techniques and costs for compliance verification

The three statements:

```
IF X>=1
IF X'<1
IF +X-1!(+X>1)
```

should all produce the same answer. Similar statements for the other operators can be used to verify their compliance

4.4 Legal considerations

None identified.

5. Closely related activities

5.1. Other X11 proposals (Type A or Type B) under consideration

X11/SC13/92-17: SET INCREMENTAL
X11/SC13/92-19: SET POSITIONAL
have possible conflicts in syntax.

5.2. Other related standards efforts

Not applicable.

5.3 Recommendations for coordinating liaison

The sponsor for this proposal is currently the task group chair for the task group that is considering SET INCREMENTAL (among three differing proposals).

The sponsor requests that the TG3: SET POSITONAL discusses the alleged conflict in their next task group meeting to see if a resolution can be made.

6. List of associated documents

7. Glossary