X11/1998-31

MUMPS Development Committee

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

IF THEN & ELSE

June 28, 1998

Produced by the MDC Subcommittee #15
Programming Structures

Art Smith, Chairman
MUMPS Development Committee

Wally Fort, Chairman
Subcommittee #15

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 that 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 to 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:

IF THEN & ELSE

1.2 MDC Proposer and Sponsor:

Proposer:
Ben Bishop
Atlantic Consultants Inc.
64 Maolis Road
Nahant, MA 01908
Pager: (800) 913-4868
e-mail: aci@shore.net

Sponsor:
SC15/TG17 (Process Control)
Wally Fort, Chair

1.3 Motion:

Final publication version, superseding document X11/SC15/1998-7.

1.4 History:

Date	Document	Action	
01 Aug 1998	X11/1998-31	Final publication version	
01 May 1998	X11/SC15/1998-7	Proposed as MDC/A	Passed: 11:3:5
20 Mar 1998	X11/SC15/1998-3	Proposed as SC15/A	Passed: 12:6:3
24 Sep 1997	X11/SC15/TG17/1997-8	Proposed as SC15/B (new formalization)	Passed: 15:4:4
01 Jan 1997	X11/SC15/TG17/97-2	Proposed for recision	Rescinded: ?:?:?
01 Aug 1996	X11/SC15/96-17	Proposed as SC15/A; motion to demote to C	Passed: 12:11:1
01 Feb 1996	X11/SC15/TG9/96-2	Assigned to TG17, presented for SC15/B	Passed: 10:4:7
(early '90s)	(Otherwise/Alternatively)	Failed after discussion.	

1.5 Dependencies:

None identified nor expected.

2. Justification

2.1 Needs:

The dichotomy between IF and ELSE would be better handled if there were some way to stack $TEST between the line containing the IF and the line containing the ELSE. The 'problem' with IF-ELSE and $TEST was identified by ISO as an objection to the ISO version of the X11.1 standard.

2.2 Existing Practice:

Common practice is one of the following:

a. Avoid using ELSE
b. Explicitly re-set $TEST (using IF 1 or IF 0) if possible.
c. Use NEW $TEST in all subroutines.

3. Description

3.1 General Description:

Add a new command **THEN** which will stack the current value of $TEST to the end of the line containing the THEN. When execution leaves the line (by GOTO or hitting the end of the line) the value of $TEST at the time of the THEN is restored.

3.2 Annotated Examples of Use:

The Code:

```
FOR A=1,0 DO
. IF A THEN  W !,"TRUE" IF 0 ;reset $TEST after write
. ELSE  W !,"FALSE"
```

Should produce a line containing 'TRUE' followed by a line containing 'FALSE' (but only one such line)

3.3 Formalization: (modifications are to X11.1 Draft v13)

Insert a new command in section 8.2 (and in the command index) as appropriate:

8.2.* TH[EN] [SP]

This command creates a new CONTEXT-STRUCTURE consisting of a NEW NAME-TABLE and attaches it to a linked list of CONTEXT-STRUCTUREs associated with the current PROCESS-STACK frame, and modifies currently active NAME-TABLEs as per 'NEW svn' for the svn $TEST. The value of $TEST is restored from this CONTEXT-STRUCTURE (& the CONTEXT-STRUCTURE is removed unless otherwise indicated) by any of the following actions (note 'current line' refers to the line with the THEN command):

- Execution encounters the eol at the end of the current line.
- A QUIT command is encountered at the current execution level (note this includes RESTARTs).
- A QUIT command returns execution to the current execution level (but does not remove the CONTEXT-STRUCTURE).
- An explicit GOTO command, located in the current line, is executed.

(Note: an Error Processing transfer of control (c.f. 6.3) does not restore the value of $TEST)

4. Implementation Effects.

4.1 Effect on Existing User Practices and Investments:

None expected.

4.2 Effect on Existing Vendor Practices and Investments:

None expected (beyond costs of implementation).

4.3 Techniques and Costs for Compliance Verification:

The example in section 3.2; other tests would encompass timed reads/open/job/lock commands.

4.4 Legal Considerations:

None identified, nor expected.

5. Closely Related Standards Activities

5.1 Other X11 Proposals Under Consideration:

None.

5.2 Other Related Standards Efforts:

None identified.

5.3 Recommendations for Coordinating Liaison:

None.

6. Associated Documents

6.1 X11/TG6/98-1	X11.1 Draft v13
6.2 X11/SC15/TG17/97-2	Prior (rescinded) THEN command proposal

7. Issues, Pros and Cons, and Discussions

7.1 September 1997, Chicago -> SC/B passed 15:4:4

New Proposal using non-metalanguage definition

Pros:	Cons:
1. [6] Requested Fix	1. [4] Yet another MUMPSy solution
2. [4] MUMPSy solution	2. [6] Allows THEN to appear anywhere
3. [9] Simple	3. [4] Fails to solve block structuring solution
4. [5] Allows THEN to appear anywhere	4. [4] Adds very little functionality

Responses to Cons: #1 "*Yet another MUMPSy solution*": MUMPSy solutions would seem appropriate for a new MUMPS command. #2 "*Allows THEN to appear anywhere*": Binding THEN to IF or ELSE (or any other place where $TEST could be effected) would result in a command where the use of the command may be valid in some places and not in others, and the rules regarding that being not obvious. #3 "*Fails to solve block structuring solution*": The purpose of this proposal is to provide a solution for the IF-ELSE & $TEST problem and does not preclude a full block structuring solution. #4 "*Adds very little functionality*": But it does provide a simple construct which effectively eliminates the IF-ELSE & $TEST problem.

7.2 March 1998, Atlanta -> SC/A Passed: 12:6:3

Scope issues in the formalization clarified.

Pros:	Cons:
1. [7] Requested Fix	1. [8] Adds complexity with little gain
2. [3] MUMPSy	2. [5] Adds another ELSE-like command that needs defending
3. [9] Best available answer to problem	3. [2] Should leave it alone.
	4. [5] ?non-recorded con

In discussion at this meeting it was pointed out that "We seem to have four choices as far as the IF-$TEST-ELSE Problem. These should be added to the history of the document as we have considered all of them.

1) Leave as is (force use of IF 1, NEW $TEST, argumentless DO).
2) Redefine IF statement so that it does stack $TEST.
3) Create a new svn that is stacked and an "OTHERWISE" command that evaluates this.
4) Add a THEN command.

Responses to cons: First, a comment: I agree that the IF-$TEST-ELSE problem can be solved with existing constructs; that isn't the issue – if you know enough to put in the existing fixes, you probably are not having the problem to begin with. New Programmers, do, however, and THEN and its syntax is significantly easier to (a) teach, (b) understand, and (c) remember to use, than the other "solutions" to this problem. Adding 'always use THEN after an IF command' to a groups' set of programming standards would easily eliminate 99% of the bugs associated with this problem. That being said, on to the Con Responses:

(1) "*Adds complexity with little gain*": Complexity compared to what? Gain compared to what? Adding THEN 'reflexively' in new code eliminates the complexity of unwanted/unexpected $TEST changes and resulting bugs. It permits the ELSE command to be used the way it was initially intended (I find very few experience programmers use ELSE because of this problem – usually sacrificing performance for 'code safety'.

(2) "*Adds another ELSE-like command that need defending*": The same defense for ELSE (and $TEST for that matter) would apply for THEN – it is a command associated with a state-variable which tracks the result of the last truth-setting event; I would take this con more seriously only if there were a proposal advancing that would eliminate the ELSE command from the language (highly unlikely).

(3) "*Should leave it alone*": There is a real flaw in the language which encourages the development of routines which have a correctable, but elusive, bug in them. Persons learning the language are usually taught the solutions (IF 1, NEW $TEST, and argumentless DO constructs) which are non-obvious and not easy to remember (and are not single-purpose). Leaving this problem alone results in the continued production of code which cannot be completely trusted. Also, there is a lot of history concerning this 'flaw' in the language, and many of the MDC know it better than I do. It ('ELSE not bound to a correponding IF) is often pointed out as one of the "less thought out" constructs in the language. This proposal does not erase that criticism, but does provide a mechanism to dampen it significantly. The sentiment expressed by con #2 seems to indicate that this opinion is shared by some MDC members as well.

7.4 June 1998, Waltham, MA Proposed as MDC/A Passed: 14:0:5

Pros: Cons:

1. Requested Fix 1. THEN usage out of context with IF and ELSE
2. MUMPSy
3. Best available answer to problem

Response to cons: This proposal has as much context wth the IF command as the ELSE command does; i.e. only through the $TEST svn. Expecting a tighter binding is very non-MUMPS like.

8. Glossary

None.

9. Appendix

None.