X11/1998-29

# MUMPS Development Committee

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

# Local Variables in ^$JOB

June 28, 1998

Produced by the MDC Subcommittee #13
Data Management and Manipulation

Art Smith, Chairman
MUMPS Development Committee

Dan Bormann, Chairman
Subcommittee #13

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:

# Local Variables in ^$JOB

## 1.2 MDC Proposer and Sponsor:

**Proposer:**
Ben Bishop
64 Maolis Road
Nahant, MA 01908
(617) 593-3038
aci@shore.net

**Sponsor:**
SC13/TG13, MUMPS Data Traversal
Rod Dorman, Chair
Polylogics Consulting
(201) 489-4200
rodd@panix.com

## 1.3 Motion:

Final publication version, superseding X11/SC13/1998-3.

## 1.4 History:

| Date | Document | Action | |
|---|---|---|---|
| 01 Aug 1998 | X11/1998-29 | Final Publication version | |
| 01 May 1998 | X11/SC13/1998-3 | Proposed as MDC/A | Passed: 14:0:5 |
| 01 Aug 1997 | X11/SC13/1997-11 | Proposed as SC13/A | Passed: 15:2:2 |
| 01 Aug 1996 | X11/SC13/1996-8 | Amended, Proposed as SC13/A | Failed: 8:11:5 |
| 01 Feb 1996 | X11/SC13/1996-1 | Proposed as SC13/A (clarified) | Passed: 10:2:5 |
| 19 Apr 1995 | X11/SC13/1995-13 | Proposed as SC13/A | Failed: 9:14:2 |
| 01 Dec 1994 | X11/SC13/1994-47 | Proposed as SC13/A Amended | |
| 09 Jun 1994 | X11/SC15/1994-24 | Proposed as SC13/B | Passed: 15:8:5 |
| 20 Apr 1994 | X11/SC15/1994-24 | Initial proposal (transferred to SC13/TG13) | |

## 1.5 Dependencies:

No proposals have been identified which depend on this proposal.
No proposals have been identified upon which this proposal depends.

# 2. Justification

## 2.1 Needs

Recently there has been a lot of discussion regarding using $Order on local variable names in order to walk through the local symbol table. In the same proposal, $Order on a global variable name would return the next global variable name which exists within that environment. This functionality is already provided by the ^$Global ssvn, and is provided within a context obvious to M programmers. Using $Order to get the next name rather than the next subscript seems non-obvious.

A simple means to get the list of local variable names which are defined for a process would seem best located in ^$JOB(processid); the ^$JOB ssvn is specifically designed to provide information about a running process. In addition, vendors could, if they wished, extend this use to provide symbol table access to another process (similar to the job examination utilities provided by vendors currently).

### 2.2 Existing Practice

Existing practice for walking through the list of currently defined local variable names consists of either (a) using $Order on a name-value (non-standard) or (b) writing the symbol table to a file and parsing the resulting output for the variable names.

Existing practice for walking through another process' symbol table consists of highly vendor-specific code which parses the symbol table of the other process using non-standardized functions (such as $View and $Z-functions).

## 3. Description

### 3.1 General description

This proposal defines several new nodes within the ^$JOB ssvn; these nodes provide access to information about the local symbol table of a process. Most likely this information will be accessible by the owning process, but implementors may extend this for use in a job examination utility.

### 3.2 Annotated Examples of Use

```
;Copy current symbol table (except %ERR* variables to ^ERROR)
Lock +^ERROR("NEW") S %ERRN=$G(^ERROR)+1,^ERROR=%ERRN
Lock -^ERROR("NEW")
S %ERRV="" F  S %ERRV=$O(^$JOB($JOB,"VAR",%ERRV)) Q:%ERRV=""  DO
. I $E(%ERRV,1,4)="%ERR" Q  ;skip error utility variables
. Merge ^ERROR(%ERRN,%ERRV)=@%ERRV ;allow parsing later

or, using the full definition of ^$JOB:

Lock +^ERROR("NEW") S %ERRN=$G(^ERROR)+1,^ERROR=%ERRN
Lock -^ERROR("NEW")
Merge ^ERROR(%ERRN)=^$JOB($JOB,"VAR")
Kill ^ERROR(%ERRN,"%ERRN") ;remove only variable used
```

### 3.3 Formalization (References are to the X11.1-1995 Draft standard)

**Add to the end of section 7.1.3.4 (^$JOB):**

```
^$JOB(processid, expr V "VAR", lvnexpr )

lvnexpr ::= expr V name
```

This node exists for all local variables in the context of the specified processid. The $DATA value of this node is determined by the $DATA value of the specified variable; likewise, the value of this node is the same as that of the specified variable (including undefined). Only variables which have a $DATA value other than zero are represented by these nodes. If a lvn is of the form: *Name*($S_1$,$S_2$,...,$S_n$) for a process *Job*, then the reference to that lvn in an expr is identical to the reference: ^$*JOB*(*Job*,"*VAR*","*Name*",$S_1$,$S_2$,...,$S_n$).

Coordination issues may arise if these nodes are examined by another process (if permitted by an implementation); a specific reference may be atomic, but multiple references are not -- the specified local variable may be NEWed or KILLed while being examined through these nodes.

Note that for technical reasons or security concerns, implementations may restrict access to ^$JOB nodes for processids other than the current processid. An attempt to violate such a restriction causes an error condition with an implementor-specified ecode beginning with "Z".

## 4. Implementation Effects

### 4.1 Effect on Existing User Practices and Investments

None identified.

### 4.2 Effect on Existing Vendor Practices and Investments

None identified beyond the issues of implementation. Note that references to the nodes of ^$JOB/"VAR" could be considered 'syntactic sugar' for references to the actual local variables (but not for instances where the variable is not referred to as part of an expr; such as in NEW, KILL, or passing by reference).

### 4.3 Techniques and Costs for Compliance Verification

None identified.

### 4.4 Legal Considerations

None identified.

## 5. Closely Related Standards Activities

### 5.1 Other X11 Proposals Under Consideration

| | |
|---|---|
| X11/SC13/1994-16 | ssvn for local variables |
| X11/SC13/TG13/1995-5 | Unsubscripted $Order |

### 5.2 Other Related Standards Efforts

None.

### 5.3 Recommendations for Coordinating Liaison

| | |
|---|---|
| X11/TG18 | ssvn coordination |

## 6. Associated Documents

None.

## 7. Issues, Pros and Cons, and Discussion

This proposal attempts to accomplish 2 goals: (1) standard access to the list of local variables within a process, and (2) define a standard mechanism for job examination at the discretion of the implementor.

## June 1994, Reno, NV

Proposed as SC13/B: Passed 15-8-5

Proposed as SC15/B for SC15/TG13 (ssvn syntax); proposal was transferred to SC13/TG13 (MUMPS data structure traversal).

In SC13/TG13 there was discussion between ^$JOB and ^$LOCAL – ^$JOB was preferred due to the potential for jobexam/job monitoring utility. "Local variables in ^$JOB" also duplicated the features of the unsubscripted $ORDER proposal (X11/SC13/TG13/1994-4) but it was decided to proceed with both proposals since they did provide different functionality.

| Pros: | | Cons: | |
|---|---|---|---|
| 1. | Allows standard Job Exam | 1. | Lots of work |
| 2. | aids in internationalization | 2. | like $ORDER better |
| 3. | provides more functionality than $ORDER | 3. | excess baggage |
| 4. | Enhances indirect error processing (Heisenberg principle). | | |

Addressing cons: (1) I can't speak for implementors, but a reference to the LVN nodes of ^$JOB *from within a process* would be the same as a lookup on the variable/node in question and appears to be a application of syntactic sugar. As far references from an outside process are concerned, the implementor can either exclude that capability outright (through the non-$J reference clause in the ^$JOB definition), or treat the reference the same as what is currently used in their job examination utility (if any).

**January 1995, Dallas, TX**

Amended to remove internationalization nodes ("CHARACTER" and "COLLATE") into a separate document and presented to SC12/TG2 (internationalization) (which it was; SC12/TG2 indicated that the functionality was not needed at this time)

In addition, the "DATA" node definition was lacking any formalism on how scaler and array references are mapped to that node – only an incidental example of a single subscripted array. It was determined that this needed to be properly formulated, and could then be presented for SC/A status at the next meeting.

## June 1995, Chicago, IL

Presented without the author; straw poll discussion on viability of 'Unsubscripted $Order' and 'Local Variables in ^$JOB', results: (see JUL95-93)

**poll 1: which proposal to proceed with:**

| | | |
|---|---|---|
| 1. | Local Variables in ^$JOB | 6 |
| 2. | Unsubscripted $Order | 7 |
| 3. | Both of the Above | 6 |

**poll 2: which should the SC NOT proceed with:**

| | | |
|---|---|---|
| 1. | Local Variables in ^$JOB | 6 |
| 2. | Unsubscripted $Order | 2 |
| 3. | Not Both | 6 |

**poll 3: Which of the following options is preferred:**

| | | |
|---|---|---|
| 1. | Both proposals | 3 |
| 2. | NOT both proposals | 14 |
| 3 | Don't care | 1 |

As a result, this proposal was not presented at the October 1995 meeting. During that meeting, there was a request to re-present this proposal at the next meeting; both proposals are expected to be presented for elevation to SC13/A status.

## March 1996, Boston, MA

Proposed as SC13/A, Passed 10:2:5

Amended wording to clean up syntactic sugar definition

| Pro: | | | Con: | | |
|---|---|---|---|---|---|
| 1. | Helpful for error handling | [3] | 1. | Other ways to get functionality | [2] |
| 2. | Not new functionality | [0] | | | |
| 3. | Good for job exam | [2] | | | |

In addition, after lengthy discussion, the proposal was amended to (a) get rid of the "DATA" node [not needed], and (b) make 'LVN' be 'VAR' [LVN too cryptic], and (c) make sure that the ^$JOB(job,"VAR",*) expression can only be used in exprs [what did 'identical' really mean?].

- **September 1996, Toronto, Canada** Amended, Proposed as SC13/A, Failed 8:11:5

This document incorporated changes approved at the March 1996 meeting (Boston)

Pro: 1. Helpful for error handling
2. Helpful for Job examination utility
3. Better than previous versions of the proposal

Con: 1. Name level $ORDER is preferred approach
2. Name level $ORDER currently in use
3. Inelegant & unneccessarily complex
4. Not implemented yet
5. Inadequate formalization

In response to cons: Con 1: actually, 'not both' was the preferred approach; the straw polls came out 7:6:6 in favor of unsubscripted $ORDER which is not a mandate by any stretch. Con 2: on some implementations. DSM for VMS does not support unsubscripted $ORDER -- and on those platforms which have it, it predates the existence of ssvns. Con 3: this solution uses a standard construct (^$JOB ssvn) in an intuitive manner (normal $Ordering through a glvn), and provides the potential for cross-job examination, and this is inelegant? I even question 'complex' -- it is simple from the user's point of view. Con 4: I find very few SC Type A's which have been implemented yet -- this con would be better served during a discussion for elevation to MDC type A, and even then there have been a large number of MDC type A's approved before they had been implemented. Con 5: I believe the discussions and amendments at the March 1996 meeting dealt with the formalization issues -- if there are other inadequacies, please point them out to the author.

- **September 1997, Chicago, IL**, Proposed as SC13/A, Passed: 15:2:2

This document superseded both earlier versions of this proposal, as well as the Unsubscripted $Order proposal; the subcommittee indicated that they wished to go forward with only one proposal, the task group elected to go ahead with this one.

| | | | | |
|---|---|---|---|---|
| Pro: | 1. Helpful for error handling | [7] | | |
| | 2. Helpful for job exam | | [5] | |
| | 3. needed for error handling | [4] | | |
| | 4. better than previuos version of proposal | | [3] | |
| | 5. able to be MERGED | | [9] | |
| | 6. No more complex than OO | [2] | | |
| | 7. Better for internationalization | | [4] | |
| Con: | 1. Alternate method already implemented | | [3] | |
| | 2. Name level $ORDER preferred approach | [?] | | |
| | 3. Name level $ORDER currently in use | | [?] | |
| | 4. Inelegant & unnecessarily complex | [1] | | |

Addressing the cons, 1: ... and that method has been around for a considerable amount of time, before the MERGE command, before SSVNs, and certainly before i18n touched MUMPS; just because it has been (partially) implemented, doesn't mean that a "BETTER WAY™" doesn't exist, and the author feels this is a better way which encompases local variables in a intuitive, MUMPS-like, way. Con 2: I do not believe there is a response possible beyond: reconsider now that this is the expressed direction (this year) of the subcommittee. Con 3: to my eyes, this is identical to con 1. Con 4: I find this solution not only elegant (copying a symbol table becomes a single command), but simple as well (for the same reason).

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

Pro: 1. Removes a limitation of the language.

## 8. Glossary

None.

## 9. Appendix

None.