1. Identification of the proposed change

1.1. Title

Spaces at end-of-line

1.2. MDC Proposer and Sponsor

This proposal originates from MDCC-E, and is presented by Frans S.C. Witte.
Motions regarding the status of this document are made by Taskgroup 9 (Routine Structure) of Subcommittee 15 (Programming Structures).

1.3. Motion

This document is the final write-up for an MDC Type A extension.

1.4. History of MDC actions

Date	Document	Action
June 1995	X11/95-96	History updated. Final write-up for MDC Type A document.
February 1994	X11/SC15/93-39	Presented for elevation to MDC Type A. Passed 28:0:3.
October 1993	X11/SC15/93-39	Presented for elevation to SC#15 Type A. Passed 21:1:1.
19 August 1993	X11/SC15/93-39	Amendmends by taskgroup incorporated.
June 1993	X11/SC15/93-28	Presented for elevation to SC#15 Type B. Discussed and amended in taskgroup. Passed 33:0:2.
7 May 1993	X11/SC15/93-28	Original proposal by MDCC-E
...	X11/SC15/93-15	XECUTE command clarification: basis of discussion

1.5. Dependencies

None.

2. Justification of Proposed Change

2.1. Needs

The proposal on the XECUTE command (XECUTE command clarification, X11/SC15/93-15) emphasized the confusion that is present in the M[UMPS] world about the allowance of spaces at certain positions in a line. The intention of that proposal was to clarify the allowance of spaces. Unfortunately, the X11.1 standard does not allow spaces at the end of a line, and therefor the "clarification" did not clarify the standard but contradicted the standard. This proposal remedies the conflict, in that it removes the restriction that spaces are not allowed at the end of a line.

2.2. Existing Practice in Area of the Proposed Change

Confusion.
Some implementors do allow trailing spaces, others adhere to the standard and do not allow trailing spaces.

3. Description of the proposed change

3.1. General Description of the Proposed Change

This proposal changes two clauses of the standard. One clause deals with the formal definition of a linebody. This is the easiest change.
By making the comment optional (putting it in square brackets), a line containing at least one command may now have trailing spaces. (A line that does not contain a command is already allowed to contain trailing spaces, because the ls (that is always there (at least according to the standard)) consists of one or more spaces, and the li may contain any number (including zero) spaces after the dot.

The other change replaces clause (all clause and section numbers refer to the 1994 ANSI X11.1 Canvass

Document) **8.1.1. Spaces in commands**. This clause specifies rules for spaces in different places in a command, and is completely rewritten. The major reason to rewrite this clause is that most of the rules specified in that clause do not "add" anything to the language, but explain some of the syntax rules defined elsewhere in the standard. The risk of such clauses is that they are not updated when the clauses they intend to explain are changed. (In fact this happened with exactly this clause 8.1.1 when the rules for spaces in commands were adapted to allow multiple spaces between commands.)

The current clause (**8.1.1 Spaces in commands**) reads:

Spaces are significant characters. The following rules apply to their use in lines.

a) There may be a space immediately preceding eol only if the line ends with a comment. (Since ls may immediately precede eol, this rule does not apply to the space which may stand for ls.)

b) If a command instance contains at least one argument, the commandword or postcond is followed by exactly one space; if the command is not the last of the line, or if a comment follows, the command is followed by one or more spaces.

c) If a command instance contains no argument and it is not the last command of the line, or if a comment follows, the commandword or postcond is followed by at least two spaces; if it is the last command of the line and no comment follows, the commandword or postcond is immediately followed by eol.

Discussion:
* Rules a) and b) can be derived immediately, and unambiguously from the definitions of ls, li, and linebody, with the exception of trailing spaces in li (which is allowed by 3.2.1, but excluded by 8.1.1, rule a)).
* Note that the last part of the last sentence of rule a) "*the* space which *may* stand for ls" still is a left-over from a previous version of the standard, that left ls unspecified, for all purposes except for routine interchange. This explains why many implementations used a <TAB>-character as ls, which was allowed by the 1976 standard, but was (un)intentionally dropped in the editorial revision of a later version.
* Note that the first discussion statement may not be completely accurate. Both the BREAK command and the VIEW command contain the wording "argument syntax unspecified", without a SP separating the postcond from this wording.
* Rule c) is the most important rule. This rule is needed because of the way the argumentless versions of commands are defined. This rule reduces the "optionality" of the space in argumentless commands. E.g. one instantiation of the QUIT command (8.2.16) is defined as **"Q[UIT] postcond [SP]"**. According to rule c) of 8.1.1, the SP must be present if the command is not the last of the line, or if it is followed by comment, and must be absent if the command is the last of the line and no comment follows.

The proposed change replaces rules a) and b) by references to the appropriate syntax-definitions, and rephrases rule c) to allow trailing spaces.

3.2. Annotated Examples of Use

```
ex1  SP eol
ex2  LABEL SP eol
ex3  SP SP eol
ex4  LABEL SP SP eol
ex5  SP SET SP X=14 SP eol
ex6  SP QUIT SP SP eol
```

```
ex7  SP QUIT SP eol
ex8  SP . SP eol
```

Examples ex1, and ex2 are requirements of the standard (i.e. a line must contain at least the SP that represents ls.. The proposal does not change that.
Examples ex3, and ex4 are already allowed by the current standard (the SP is part of ls).
Examples ex5, and ex6 are not allowed by the current standard, but will be allowed by the proposed change.
Example ex7 is allowed by the current syntax-definition of the QUIT command, but excluded by rule c) in 8.1.1 (spaces in commands). The proposed change would make this valid syntax.
Example ex8 is allowed by the current syntax-definition of li (clause 6.2.1), but excluded by rule a) in 8.1.1 (spaces in commands). The proposed change would make this valid syntax.
NOTE that the (argumentless) QUIT at the end of the line may be followed by ONE or more spaces just like any other command. (the rule that it is followed by at least two spaces if it is not immediately followed by eol is changed).

3.3. Formalization

(all section and clause numbers refer to the 1994 ANSI X11.1 Canvass Document)

In **section 1, clause 6.2.5 Line body linebody**, replace the definition of linebody with the following definition:

```
linebody ::= [ commands [ cs [ comment ] ]
             [ commands cs ] extsyntax
                       comment             ] eol
```

Replace **section 1, clause 8.1.1 Spaces in commands** completely with the following:

Spaces are significant characters. The following rules apply to their use in lines.

a. If a command instance contains no argument, and it not the last command of a line, or if a comment or extsyntax follows, the commandword or postcond is followed by at least two spaces. If it is the last command of a line and no comment or extsyntax follows, the commandword or postcond may be followed by zero or more spaces.

b. In all other cases, the use of spaces is defined by the appropriate command definition and subclauses of 6.2 Routine body, and 6.4 Embedded programs.

4. Implementation impacts

4.1. Impact on Existing User Practices and Investments

This is intended to be an extension to the current standard, so there should be no impact on existing user practices. The benefit of this proposal is more generality, and probably less confusion.

4.2. Impact on Existing Vendor Practices and Investments

Implementations that do conform to the current standard, must be adapted.
Many implementations do already (intentionally or by accident) incorporate this change.

4.3. Techniques and Costs for Compliance Verification

A conforming implementation must pass all of the examples in section 3.2 of this proposal. Similar tests can be specified for other argumentless commands.

4.4. Legal considerations
None known.

5. Closely related standards activities

5.1. Other X11 Proposals (Type A or Type B) Under Consideration
X11/SC15/93-15: XECUTE command clarification (superseded by X11/SC15/93-28).

5.2. Other Related Standards Efforts
None known.

5.3. Recommendations for Co-ordinating Liaison
None known.

6. List of Associated Documents

None.

7. Issues, Pros and Cons, and Discussion

7.1. June 1993, Washington DC
Document presented to SC#15 for elevation to Type B. Remanded to Taskgroup 9 (show of hands 14:9).
Discussed and amended in Taskgroup.
1. Replace section 5.2 with "None known".
2. In section 3.1, near end, replace ELSE example with QUIT.
3. In section 3.3, use brackets instead of vertical lines for linebody.
4. Ass "is" in first line of paragraph ‘a)’ on page 767 where appropriate ("... and it is not ...").
Pro: 1. Needed functionality
2. Simplifies code generation.

7.2. October 1993, Dublin, Ireland
Document presented to SC#15 for elevation to Type A. Document editor was directed to expand and update the document history and use the same version of the RMDS throughout.
Pro: 1. Has been implemented.
2. More consistent language
3. Easier to generate.
Con: 1. Ambiguous with regard to the commandword or postcond.
2. Uses M instead of MUMPS.
Remanded to taskgroup (7:2:6).
Discussed in taskgroup.
Presented again for elevation to Type A.
Pro: 1. Avoids errors [3]
2. Removes unspecified error condition [2]
3. Removes unnecessary restriction [3]
Con: 1. Backwards incompatible [1].