Prof. Dr. H. Stoyan
Universität Erlangen-Nürnberg
Institut für Mathematische Maschinen
und Datenverarbeitung (Informatik VIII)
Am Weichselgarten 9
91058 Erlangen

A.I. Internal note NOV 25, 1968

A TEMPORARY UPDATER FOR LISP USERS

This short system note is intended to fill the information vacuum since the publication of A.I. memo #157 in March 1968. In addition to repairing many bugs and inadequacies, I have added some new features which are described on the succeeding pages; some are in the current LISP, all are in NLISP. On about Dec 1, 1968 NLISP will become the standard system. When this happens, "NEW LAP" will go away — use thenceforward either E LAP or C LAP.

NOTE WELL! Items 4) and 7) below document changes not upward compatible in READ with an argument and 'multiplexor' usage.

MODIFICATIONS APPARENT IN LISP 94

1) The "E" feature of the MIDAS Linking macros does not operate as advertised, and furthermore is superflous. A new macro "EXECUTE" (option #6) is installed, which signals a closed subroutine (exited by a POPJ P,) to be executed once immediately following the completion of the linking of the subprogram in which it occurs. This allows the MIDAS user to do automatically a little post-ALLOCing initialization, which would include, but is not limited to, changing some code of the already loaded LISP system.

2) ERRSET now works as advertised: errset [s] is list [eval] [s]] if no error occurs while evaluating s and is NIL otherwise. However, if the function ERR is called while evaluating s, then the error recovery routines are entered (but not error-message is printed on the console); and if ERR is given an argument, then this argument is returned by ERRSET instead of NIL.

IF a second argument is given to ERRSET, then the error-message printout is suppressed during the evaluation of the first argument.

Whenever an error occurs in an evaluation not under the control of ERRSET, LISP is re-initialized (inasmuch as this is possible) including the resetting of the I/O switches. Since there is no A-list in PDP-6 LISP, a paltry attempt is made to restore the value cells of those atoms having higher-level bindings (see LISP write-up, *RSET, SPECIAL VALUE CELL, variable bindings). Just after the re-initialization is completed, and just before the main top-level loop is re-entered, the following evaluation takes place mapc [EVAL; ERRLIST]. Thus ERRLIST provides a means for a user to add his own 2¢ worth to the re-initialization process.

Some errors are of such a drastic nature that ERRSET cannot be guaranteed to recover from them, and they always transfer to the re-initialization:

(a) PDL or Special PDL overflow

(b) No free storage or full word space left

(c) LG typed at console. LX typed is a quit signal which is caught by ERRSET; LG is unconditional.

(d) Illegal instruction execution. A typical bug leading to this error is a RPLACD which clobbers some part of the system.

3) DELETE (and DELQ) has an optional third argument, which is taken as an upper limit on the number of deletions to be made. delete[a; b] deletes only the first occurence of a in the list b, while delete[a; b; 104] deletes the first 104 such occurences. DELETE works by CDRing or RPLACDing, and returns a pointer to the deleted list.

4) READ (and READCH) has an optional argument, in order to allow end-of-file detection. The ordinary action upon reading a LC (the EOF character for MAC tapes) on an auxiliary device is to close the input file, reselect the job console for input, and wait for an expression to be typed. However, read[s] will close the file and return s as the value of READ; if no EOF is encountered in reading the next s-expression of the input file, then read[s] functions just like read[].

5) The character <alt mode> does not need a slash preceding it to be read in by LISP, but on print-out a slash will be inserted to distinguish it from <dollar>. The characters "t" and "-" may be used like any other single-character atom. Note: -5 and +A are numbers while /-5 and /+A read in as alphabetic atoms.

6) UREAD, UWRITE, and UFILE have default options in lieu of arguments, just like Teco. However, if UREAD is given any arguments, the first two are interpreted as filename1 and filname2; if a third is present it is device name; and if a fourth is present it is user name. Similarly, if UWRITE has any arguments, the first is taken as device name and the second, if present as user name.

7) The method of opening and closing the A/D and D/A multiplexor has been enlarged: Two arguments are now required, the first for the A/D input multiplexor and the second for the D/A output multiplexor. In each case, the meaning of the argument is as follows:

NIL	–	leave open-close status unchanged,
0	–	close device
1	–	open device in normal mode (image)
2	–	open device in fast mode (ascii)

[see A.I. Memo #161, ITS 1.4 Reference Manual, §§ 4.9, 4.10]

8) NVFIX can also read in block mode: nvfix[arrayname; n] will assume, for $0 \le i < n$, that $arrayname_{2i}$ has an x-value and $arrayname_{2i+1}$ has a y-value, and will replace the latter by nvfix[x; y]. LISP numbers are used both coming and going, but generally small numbers will suffice.

One can write a small MAP routine to fill the first n words of some array for input to NVFIX, if the points to be read are related in some reasonable fashion. In each word the left-half holds the x-value, the right-half the y-value, and the return value is in the right-half.

NVSET is used to set the conditions for NVFIX, regardless of which mode is used. An error occurs if the first argument to NVFIX is neither an array nor a number.

9) Some DISPLAY Features.

THE 340 DISPLAY MAY BE USED IN T.S. LISP AS AN ORDINARY OUTPUT DEVICE. JUST LIKE THE LINE PRINTER. THE PLOTTER. UTAPE AND DISC DEVICES. AND THE TELETYPE. ↑N TURNS ON AND ↑Y TURNS OFF THIS FEATURE. OF COURSE. ONE CANNOT SIMULTANEOUSLY AND AT THE SAME TIME BE OUTPUTTING TO THE DISPLAY AND DISPLAYING FROM A DISLIST [BY MEANS OF ↑F].

DISLIST IS USED AS BEFORE. EXCEPT THAT THE RELEVANT INFORMATION IS FOUND AS AN "ARRAY" PROPERTY INSTEAD OF "SUBR". THE SYSTEM TRIES TO BE CLEVER IN ORDER TO AVOID DISPLAY MEMORY PROTECT VIOLATIONS. REGARDLESS OF HOW THE LOSER LOSES HIS ARRAYS.

10) LU is no longer used to halt a plot started by IPL or by plot[0], but only for halting the top level output to the plotter (as LE is used for the line printer). The careful plotter-nik will say plot[200] to close his schemes.

11) The new disk units are available for I/O usage as device Dsk. Alternately one can use PKn to write on a specific disk pack.

12) Accumulator 13 is not used or referenced by LISP. One may safely design obscure uses for it with LAP and MIPAS sub-programs.

13) The length of the OBLIST may be prescribed at assembly time with minimal effort. The length of the OBLIST in LISP's with version numbers > 91 is set to 777. If you have slow-read-itis because of atom verbosity, contact Joel.

14) NOUUO, NOCHK, NORET, BAKGAG, GCTWA, *NOPOINT, *RSET all set accessible switches, namely their own special value cells. Thus nochk[8] accomplishes the same as set[NOCHK; 8].

There are a number of global variables critically pertinent to system operation — a brief catalog follows on the next three pages. This is primarily for convenience since nothing essentially new is documented therein.

ATOM	VALUE
BASE	Radix to be used when printing out fixed point numbers. Used by PRINT
GCMKL	cdr [GCMKL] is a list of pairs, each pair a special Array Cell and the number of cells The corresponding array occupies used by the Garbage Collector
CHRCT	Number of character spaces remaining in current output line. PRINT enforces a maximum output line length of LINEL (q.v.), and when CHRCT reaches zero, a carriage return is emitted and CHRCT reset to LINEL
BPORG BPEND.	When the system is initially loaded, all the memory From BPORG up to the top of the core allotment is available as a kind of amorphous storage (not part of Free Storage or Full Word Space). Subroutines loaded by LAP are generally stored in the lower end of this space, and Arrays are dynamically stored at the higher end. BPORG is updated by LAP to indicate the lowest currently available cell of this space, and BPEND is updated by the array handler and garbage collector to indicate the highest currently available cell.
DISCNT	When DISAD is used to prepare an array of Text for 340 Display, DISCNT serves the same dynamic function for pseudo-output to the array as CHRCT does for regular output, say, to the line printer.
DSLP	The number of carriage returns inserted by DISAD since the last DISINI; can affect the

number of lines of text inserted in the array to be displayed.

DISLIST — One means of displaying information on the 340 display scope is the creation of "display arrays" — each such array is a sequence of instructions to the 340 (see Manual for usage). DISLIST is a list of the special array cells of those display arrays one wants active. In Time Sharing version, the display is not actually carried out unless the ↑F switch is on (q.v.).

ERRLIST — When a non-ERRSET error is detected, control passes to an initialization routine, just before re-entering the top level main loop. Among other things mapc[EVAL; ERRLIST] is performed, thus allowing the user to add his own two cents to the error recovery process

IBASE — radix to be used by READ when converting a number-like string on input. May be any value from 1 to 36., except that any number-like string having non-decimal digits must begin with either "+" or "-". EXAMPLES:

IF IBASE is 8, Then "47" is read in as 39.
IF IBASE is 12, Then "14" is read in as 16.
IF IBASE is 16, Then "+1A" is read in as 27.

"1A", "+2-2", are illegal atoms as atoms
While "+", "-", "A1", "B-2" "/1A". are acceptable.

↑x — Herein, "x" and "y" are used as meta characters. Typing ↑X (control-x) from the console, or evaluating (nwritn) sets the x-switch to on. The x-switch is merely the value of the atom ↑x (↑-arrow x). Each such switch x (except ↑A) has a corresponding control character which turns it off.

x	meaning of switch	y
A	~~available to user~~ not used by system	(none)
B	turn on line printer to receive PRINT output	E
D	turn on output of garbage Collector statistics (one set produced after each collection)	C
F	enable displaying from the display arrays of DISLIST	Y
~~G~~	~~Unused~~	
N	in time sharing, turn on the DIS device to receive PRINT output. (Physically, the DIS device is the same as the 3 to display, and one cannot use them simultaneously)	Y
P	Turn on the Plotter to receive PRINT output.	U
Q	Let READ receive its input from the currently selected input file, rather than the job console	S
R	enable the currently open output file to receive PRINT output.	T
W	disenable the job console from receiving PRINT output	V

15) Relocatable Arrays.

ARRAYS ARE STORED ON THE PROPERTY LIST OF ATOMS AS THE "ARRAY" PROPERTY, INSTEAD OF "SUBR" PROPERTY. THUS ARRAYS ARE AUTOMATICALLY GARBAGED-COLLECTED WHEN NOTHING IN THE SYSTEM POINTS TO THEM. ARRAY SPACE IS THE AREA BETWEEN BPEND AND THE TOP OF CORE, AND BPEND IS RESET FROM TIME TO TIME REFLECTING THE VARYING AMOUNT OF SPACE REQUIRED. CALLING "(GETSP N)" WILL INSURE THAT THERE IS AT LEAST N FREE CELLS BETWEEN BPORG AND BPEND. (BPORG HOLDS THE ADDRESS OF THE FIRST CELL CURRENTLY AVAILABLE FOR BINARY PROGRAMS - - BPEND HOLDS THE CURRENT ADDRESS OF THE LAST CELL.) THE FUNCTION "ARRAY" IS USED AS BEFORE, BUT ALL ENTRIES ARE ZEROED AUTOMATICALLY. TO REMOVE AN ARRAY WITHOUT REMOB-ING ITS ATOM HEADER, USE "REMAR" IN A FASHION SIMILAR TO "REMOB". BOTH "ARRAY" AND "GETSP" WILL RETURN NIL IF THE REQUESTED SPACE JUST CANNOT BE OBTAINED.

ALL REFERENCES TO ARRAYS NOW ARE CHECKED TO SEE THAT THE CELL ACCESSED IS INDEED A PART OF THE ARRAY MENTIONED. THUS MISTAKES SUCH AS STORING OVER THE END OF THE ARRAY, OR CLOBBERING SOME CODE IN BPORG AREA, ARE PREVENTED. THIS FEATURE TAKES VERY LITTLE EXTRA TIME, BUT MAY BE INHIBITED BY EXECUTING "(NOCHK T)", AND DISINHIBITED BY "(NOCHK NIL)".

16) Garbage Collector goodies

THE GARBAGE COLLECTOR HAS A FEW OTHER LITTLE GOODIES IN ADDITION TO THE ARRAY RELOCATING FEATURES: A TRULY WORTHLESS ATOM (TWA) IS DEFINED TO BE ONE WITH A TRIVIAL PROPERTY LIST AND WHICH IS POINTED TO BY NOTHING ALIVE IN THE SYSTEM EXCEPT THE OBLIST. AN INTERNAL SWITCH, THE "GCTWA" SWITCH, IF NON-NIL WILL CAUSE THE GARBAGE COLLECTOR TO REMOVE ALL TWAS FROM THE OBLIST. EXECUTING "(GCTWA S)" WILL SET THE SWITCH TO THE VALUE OF S; SINCE THIS FEATURE ADDS ABOUT 15% IN TIME TO A GC, THE NORMAL MODE OF THE SWITCH IS NIL. AS A VARIANT, EXECUTING "(GCTWA)", WILL FORCE THE TWA REMOVAL PHASE TO HAPPEN DURING THE IMMEDIATELY ENSUING GARBAGE COLLECTION.

WHENEVER THE G.C. FINDS THAT BPORG IS SO MUCH LESS THAN BPEND THAT TWO OR MORE BLOCKS OF CORE COULD BE RETURNED TO THE TIME-SHARING SYSTEM, THEN IT WILL DO SO, FIRST COMPACTIFYING ARRAYS AND RELOCATING THEM DOWNWARDS. "(NORET T)" INHIBITS THIS FEATURE. WHICH IS DISINHIBITED IN THE USUAL WAY.

17) USING LISP PRINT WHILE IN DDT.

When debugging LISP programs with DDT, often sees a pointer typed out, instead of the beautiful fully-parenthesized ◎ style of PRINT output. Typing P.$X will cause DDT to enter a part of LISP which prints out the currently open cell in LISP format, and return to DDT when done. [Actually "P." is a symbol in the Job symbol table equal to "PUSHJ P, PSym"]

18) A quick look at some new LAP features

(a) LAP now communicates with the DDT symbol table for the LISP job in which it is running. Thus all operation codes are defined in LAP by means of the DDT op-decoder. The "SYMBOLS" pseudo-op, described in A.I. memo # 152 PDP-6 LAP, page 4, works quite well now.

(b) If there are any undefined referenced symbols in a LAP assembly, then the DDT symbol table is interrogated to try to make a definition. If a value is successfully pulled back from DDT and used, a warning message is printed: "DDT SYMBOLS" followed by the ones found. Otherwise, a message "UNDEF SYMBOLS" is printed followed by the recalcitrant ones.

(c) In both the above-discussed features, if a LAP symbol has characters not legal for MIDAS symbols, then these characters are converted to ".".

(d) If a non-drastic error occurs during assembly, LAP records as much as possible and aborts assembly. If the assembly occurred while reading from an auxiliary device, then the rest of the LAP subroutine is skipped over so that other routines may assemble successfully.

(e) REMLAP works well, needs no argument, and performs no dangerous remob-ings. One may even reload LAP after remlapping.

(f) Three new pseudo-ops are installed

(BLOCK n) assembles a block of n zeros

(ASCII &) does a PRINC of the S-expression & with ascii characters, just as MIDAS directly does.

(SIXBIT &) Ditto, but with sixbit characters

The ASCII and SIXBIT forms may be used as assembly word constituents, in which case they point to the first location of the ascii or six bit string. For example,

```
(MOVEI 1 (ASCII (LIST)))
NIL
```

would assemble as

```
MOVEI 1, .+1
ASCII /(LIST)/
```

(g) With reasonable allocations for free storage and full word space, LAP should assemble about 40 to 70 instructions per second of cpu time

19) PATOM is a new routine which returns non-nil only for pointers into the free storage area which are not pointing to atoms. Using PATOM, routines such as MEMBER, EQUAL, SUBST, etc no longer get hopelessly tied-up when a list has some pointers out of the free storage area.

20) If an auxiliary device runs out of space while a file is being written on it by LISP, then LISP prints on the console an appropriate comment and allows the user the same recourse as with Teco.

JonL