Bulletin 10
January 15, 1988

HOTLINE! is published periodically by the Customer Support group of Xerox Artificial Intelligence Systems to assist its customers in using the Xerox Lisp environment. Topics covered include answers to questions that are most frequently asked of Customer Support, suggestions to help you work in the Xerox Artificial Intelligence Environment (XAIE) as well as announcements of known problems that may be encountered.

Feel free to make copies of individual bulletin pages and insert them in the appropriate place(s) in your Interlisp Reference Manual, Lisp Library Modules manual or other relevant manual. The documentation reference at the end of each topic can be used as a filing guide.

For more information on the questions or problems addressed in this or other bulletins please call us toll-free in the Continental United States 1-800-228-5325 (or in California 1-800-824-6449). Customer Support can also be reached via the ArpaNet by sending mail to AISUPPORT.PASA@Xerox.com, or by writing to:

Xerox AIS Customer Support
250 North Halstead Street
P.O. Box7018
Pasadena, CA 91109-7018
M/S 5910-432

In this issue

In response to user requests we have decided to have *HOTLINE!* cover all supported releases of XAIE, instead of Lyric only. Supported releases include Koto and Lyric. Each item now contains a "Release" field for any item that is release specific. The following topics are covered in this issue:

- Make Script!
- Porting CL files to Lyric
- Preceding DEFxxx's with comments in SEdit
- Long copyright strings
- Advice replicated when loaded more than once
- LOGOUT resets the TTY parameters
- CL mapping functions

Terminology

Terminology used in this *HOTLINE!* bulletin:

UG – Users' Guide

AR – Action Request, a Xerox problem tracking number (e.g. AR 8321)

IRM – Interlisp Reference Manual

Make Script!

Topic It is necessary to specify a "LispFiles" volume with a minimum of 300 disk pages in order to store the required Lisp microcode.

Release Koto and Lyric

Keywords Make Script!, System Tools, Lisp microcode, LispFiles

Discussion During a Make Script session, the user is allowed to enter his/her desired partitioning scheme. The only volume that the tool supplies a default for is SystemTools.

It is also imperative to provide a volume named "LispFiles" with a minimum of 300 disk pages. This volume will contain the Lisp microcode upon initialization.

If this volume is not supplied, an error will occur at the time of initializing System Tools.

Example

Volume 1:	SystemTools	Volume 1 Size = 4200 (Lyric)
Volume 2:	LispFiles	Volume 2 Size = 300
Volume 3:	JimsFiles	Volume 3 Size = 10000
Volume 4:	JanesFiles	Volume 4 Size = 10000
Volume 5:	Lisp	Volume 5 Size = 32500

References Xerox 11xx User's Guide, System Tools.

Porting CL files to Lyric

Release Lyric

Keywords Common Lisp, Portability

Problem Comments from Common Lisp textual source files are not kept when the file is loaded.

Example The user loads a Common Lisp textual source file, such as EXAMPLE.LISP. This file has comments:

```
(LOAD 'EXAMPLE.LISP :PACKAGE (FIND-PACKAGE "XCL-USER"))
```

The functions are collected and assigned to the file EXAMPLE with the function IL:FILES?.

The user types (IL:MAKEFILE 'EXAMPLE) to create a new source file. The new file EXAMPLE doesn't have any comments.

Workaround The Lisp User's package PORT-CLFILE has a function which retains the comments when porting Common Lisp code to the Xerox environment.

References None.

Preceding DEFxxx's with comments in SEdit

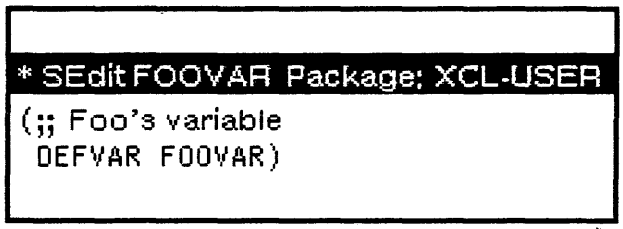
Release Lyric

Keywords SEdit, Comments

Problem In SEdit, inserting comments before DEFxxx signals an error.

Symptom Saving an SEdit definition causes a break window to appear with the error message "Unbound variable: DEFxxx".

Example In SEdit, the user has entered the following definition:



When the user types Control-X to exit SEdit, a break window with the message "Unbound variable: DEFVAR" will appear.

Workaround None. The comment should be moved; it can be placed anywhere after the DEFxxx symbol.

References None.

Long copyright strings

Release Lyric

Keywords Copyright, Comments

Problem Long copyright strings may cause a break when a source file made with a Common Lisp readtable is loaded.

Symptom Loading the file will result in a break window with the message "Bad compiled function." This message will be followed by the word from the copyright notice that was mis-interpreted as a function.

Background With a Common Lisp readtable, the copyright notice is printed in the semicolon format for comments. If the copyright owner string is "too long" (see example), then the notice is continued on the next line. However, a carriage return is inserted at the end of the current line.

When the file is loaded, the Reader terminates the comment when the carriage return (new line) is read, and the next line (still part of the notice) is interpreted as a function.

Example The value of the variable IL:FILELINELENGTH is 80, and the value of the variable IL:COPYRIGHTOWNERS is '((ACME "Acme Corp – The Nation's Number One All Purpose Company!!")). The length of the owner string is 57 characters.

The length of the copyright "prefix" string

```
"; Copyright (c) 1988 by "
```

is 24 characters. Since, the sum of the lengths of the prefix and owner string (24 + 57 = 81) is greater than IL:FILELINELENGTH, the remainder of the notice

```
"Acme Corp – The Nation's Number One All Purpose Company!!.
All rights reserved."
```

will be printed on the next line. If this source file is loaded, the "bad compiled function" will be "Acme."

Workaround Advise the function PRINTCOPYRIGHT as follows:

```
(ADVISE 'PRINTCOPYRIGHT 'AROUND
    '(LET ( (*READTABLE* (FIND-READTABLE "INTERLISP")))
       *))
```

This workaround is typed under the Interlisp Exec. Temporarily re-binding *READTABLE* to the Interlisp readtable will cause copyright notice to be printed as an Interlisp comment.

References AR 9197
IRM 17.52-54 on Copyright.
IRM 25.11, 26.48 on FILELINELENGTH.
Lyric Release Notes, page 47 on READTABLEPROP.

Advice replicated when loaded more than once

Release Lyric

Keywords Advise, Advice (Interlisp File Manager Command)

Problem Advice on a name is additive, so when a file containing advice is loaded more than once (e.g. load compiled file, load source file) the advice is replicated for each time it is loaded.

Background When you use the Advise File Manager command, Interlisp readvises the function when the file is loaded by writing a call to the Readvise function in to the file. When you use Advice, the advice is placed on the function's property list, but you must explicitly readvise the function after loading the file.

Example of Advise In the Interlisp Exec, an advised function AdviseExample works as follows:

```
> (DEFINEQ (AdviseExample (X Y) (IPLUS X Y]
(AdviseExample)
(ADVISE 'AdviseExample 'BEFORE 'FIRST '(PRINT "The Advise
File Manager Command"))
> (AdviseExample 4 5)
"The Advise File Manager Command"
9
> (SETQ THEADVISEEXAMPLECOMS
    (LIST
        (LIST 'FNS 'AdviseExample)
        (LIST 'ADVISE 'AdviseExample)))
((FNS AdviseExample) (ADVISE AdviseExample))
```

Make the file, load it again, and the advise will be replicated.

```
> (AdviseExample  4 5)
"The Advise File Manager Command"
"The Advise File Manager Command"
9
```

Example of Advice

```
>(DEFINEQ (AdviceExample (X Y) (IPLUS X Y]
(AdviceExample)
> (ADVISE 'AdviceExample 'BEFORE 'FIRST '(PRINT "The
Advice "))
(AdviceExample)
> (AdviceExample 3 4)
"The Advice Example "
7
```

Set the file's COMS so that an advice statement will put the advice on the property list for calling by the user with the function Readvise.

```
> (SETQ THEADVICEEXAMPLECOMS
    (LIST
        (LIST 'FNS 'AdviceExample)
        (LIST 'ADVICE 'AdviceExample)))
((FNS AdviceExample) (ADVICE AdviceExample))
```

Make and load the file.

```
> (AdviceExample 3 4)
7
> (READVISE)
(AdviceExample)
> (AdviceExample 3 4)
"The Advice Example"
"The Advice Example"
7
```

Compile the file and load it.

```
>(AdviceExample 3 4)
7
>(READVISE)
(AdviceExample)
>(AdviceExample 3 4)
"The Advice Example"
"The Advice Example"
"The Advice Example"
7
```

Workaround None.

Reference AR 8495, AR 9440
IRM Vol. 2, pp. 17.34-17.35
Xerox Common Lisp Implementation Notes, Lyric Release, page 111.

LOGOUT resets the TTY parameters

Release Lyric

Keywords RS232, DLTTY

Problem LOGOUT resets the TTY port parameters – but not to the values specified by the variable TTY.DEFAULT.INIT.INFO.

Note: This bulletin corrects an error in Bulletin #6.7 of *HOTLINE!* Thanks to Bill van Melle for bringing it to our attention.

Sympton A TTY device fails to work after the user has returned from a LOGOUT.

Workaround After every LOGOUT, the user must re-initialize the TTY parameters with the TTY.INIT function. If the user accidentally attempted to use the TTY port before resetting the parameters, the port may become "stuck." Then the user must LOGOUT, log back in, and use TTY.INIT to re-initialize the port parameters.

One way to automatically re-initialize the port after every LOGOUT is to define a function to initialize the TTY parameters, and add this function to the variable AROUNDEXITFNS (see example).

Example The user wants the TTY port to be initialized to 4800 baud, 8 bits, no parity, 1 stop bit, and XOnXOff enabled.

With SEdit, edit the variable AROUNDEXITFNS. Add the following function to the list of functions:

```
(LAMBDA (EVENT)
   (if  (EQ EVENT 'AFTERLOGOUT)
        then  (TTY.INIT4800 8 'NONE 1 'XOnXOff)))
```

This function will be called everytime the user returns from a LOGOUT.

Reference AR 9278
Lisp Library Modules Manual, Lyric Release, pages 212-213.
Lyric Release Notes, Changes to Interlisp-D, pages 58-59.

CL mapping functions

Release Lyric

Keywords cl:mapcar, cl:maplist, cl:mapc, cl:mapl, cl:mapcan, cl:mapcon

Problem Common Lisp mapping functions fail if given a circular list.

Note: This bulletin corrects an error in the example provided with Bulletin #9.9 of *HOTLINE!*. Thanks to Hans Koomen of the University of Rochester and Xerox for pointing out the error.

Example The Common Lisp mapping functions fail to return if one of the lists given is circular. The expected behavior is that "the iteration terminates when the shortest list runs out, and excess elements in other lists are ignored." (CltL p. 128.)

In the XCL exec:

```
(SETQ *PRINT-CIRCLE* T)
(SETQ  LINEAR-LIST (LIST 'A 'B 'C))
(SETQ  ANOTHER-LIST (LIST 'D 'E 'F))
(SETQ  CIRCULAR-LIST (NCONC ANOTHER-LIST ANOTHER-LIST))
(MAPCAR #'(LAMBDA (X Y) (PRINT (LIST X Y)))
    LINEAR-LIST  CIRCULAR-LIST)
```

Symptom The system will hang with the cursor indicating intermittent garbage collection.

Workaround A patch file is currently being tested.

Reference AR 9420