

File created: 16-Jun-88 11:17:18 {POGO:AINORTH:XEROX}<LOOPSCORE>LYRIC>LIBRARY>MASTERSCOPE.;  
11

changes to: (FNS MSADDANALYZE)  
previous date: 6-Jun-88 09:57:43 {POGO:AINORTH:XEROX}<LOOPSCORE>LYRIC>LIBRARY>MASTERSCOPE.;10  
Read Table: INTERLISP  
Package: INTERLISP  
Format: XCCS

::  
:: Copyright (c) 1983, 1984, 1985, 1986, 1987, 1988 by Xerox Corporation. All rights reserved.

```
(RPAQQ MASTERSCOPECOMS
  [(FILES MSPARSE MSANALYZE)
   (PROP FILETYPE MASTERSCOPE)
   (COMS * MSDATABASECOMS)
   (COMS * MSAUXCOMS)
   (COMS * MSDBCOMS)
   (COMS * MSCHECKBLOCKSCOMS)
   (COMS * MSPATHSCOMS)
   [COMS (FNS MSFIND MSEDITF)
          (VARS MSBLIP MSFNTPES)
          (COMS
            (INITVARS (MSCRATCHASH))
            (DECLARE%: DONTCOPY (MACROS SCRATCHHASH]
          ; SCRATCHHASH
   (COMS
     (FNS MSMARKCHANGED CHANGEMACRO CHANGEVAR CHANGEI.S. CHANGERECD MSNEEDUNSAVE UNSAVEFNS)
     (ADDVARS (COMPILE.TIME.CONSTANTS))
     (VARS (RECORDCHANGEFN 'CHANGERECD))
     (INITVARS (CHECKUNSAVEFLG T)
              (MSNEEDUNSAVE))
     (DECLARE%: EVAL@COMPILE DONTCOPY (MACROS GETWORDTYPE))
     (COMS
       [VARS * (LIST (LIST 'MASTERSCOPEDATE (SUBSTRING (DATE)
                                                         1 9]
                    (ADDVARS (HISTORYCOMS %.)
                             (FNS %. MASTERSCOPE MASTERSCOPE1 MASTERSCOPEEXEC)
                    ; marking changed
       ; interactive routines
       (FNS MSINTERPRETSET MSINTERPA MSGETBLOCKDEC LISTHARD MSMEMBSSET MSLISTSET MSHASHLIST MSHASHLIST1
            CHECKPATHS ONFILE)
       (FNS MSINTERPRET VERBNOTICELIST MSOUTPUT MSCHECKEMPTY CHECKFORCHANGED MSSOLVE)
       (DECLARE%: DONTCOPY (RECORDS GETHASH INRELATION PATHOPTIONS MSANALYZABLE)))
     (FILES MSCOMMON)
     (DECLARE%: DONTCOPY (COMS * MSCOMPILETIME))
     (DECLARE%: DONTVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILEVARS (ADDVARS (NLAMA %.)
                                                                           (NLAML)
                                                                           (LAMA MSEDITF MSEDITF]))
   (FILESLOAD MSPARSE MSANALYZE)
  (PUTPROPS MASTERSCOPE FILETYPE :COMPILE-FILE)
(RPAQQ MSDATABASECOMS
  ((FNS UPDATEFN MSEDITF MSGETDEF MSNOTICEFILE MSSHOWUSE MSUPDATEFN1 MSUPDATE MSNLAMBDAHECK MSCOLLECTDATA)
   (FNS UPDATECHANGED UPDATECHANGED1)
   (VARS TABLE.TO.NOTICED)
   (FNS MSCLOSEFILES)
   (VARS (MSFILELST)
          (MSOPENFILES))
   (VARS (MSPRINTFLG '%.)
          (MSPRINCNT 0))
   (ADDVARS (MSHASHFILE)
            (ANALYZEUSERFNS))))
(DEFINEQ
(UPDATEFN
 [LAMBDA (FN EVENIFVALID IFCANT TYPE)
  (PROG NIL
   (OR (AND FN (LITATOM FN))
        (RETURN))
   (MSINIT)
   (COND
    ((AND (NOT EVENIFVALID)
          (NOT (GETHASH FN MSCHANGEDARRAY))
          (TESTRELQ KNOWN FN))
     (* Previous valid analysis -- return)
     (RETURN))
    (SETQ DEF (MSGETDEF FN TYPE (SELECTQ IFCANT
                                             (0 'CURRENT)
                                             '?))
  ; Edited 27-Jan-88 16:49 by jrb:
  (* Update the analysis of a particular function -
  this is a "user" entry)
```

```

      ' (NOERROR NOCOPY MASTERSCOPE]
(COND
  ( (NULL DEF)                                     (* No definition can be found -- look at IFCANT)
  (SELECTQ IFCANT
    (ERROR (ERROR FN "can't be analyzed " T))
    ((NIL PRINT)
      (printout T "Sorry, the function " .P2 FN " can't be analyzed!" T))
    NIL)
  (COND
    ((TESTRELQ KNOWN FN)
     (MSERASE (LIST FN)))
    (T (PUTHASH FN NIL MSCHANGEDARRAY)))
  (RETURN)))
(MSUPDATEFN1 FN DEF])

```

(MSEDITF

; Edited 31-May-88 17:58 by jrb:

```

[LAMBDA ARGCOUNT
  (LET [(FNAME (ARG ARGCOUNT 1))
        (FEDITCOMS (for X from 2 to ARGCOUNT collect (ARG ARGCOUNT X)
                      (for FPTYPE in MSFNTPES bind FPNAME when (HASDEF FNAME (SETQ FPNAME (ffetch (MSANALYZABLE FILEPKGNAME)
                                                                                               of FPTYPE))))
                    do (if (EQ FPTYPE 'FNS)
                          then (APPLY 'EDITF (CONS FNAME FEDITCOMS))
                          else (EDITE (GETDEF FNAME FPNAME NIL ' (NOERROR NOCOPY EDIT))
                                       FEDITCOMS FNAME FPNAME))
                      (RETURN FNAME]))

```

(MSGETDEF

; Edited 1-Jun-88 15:14 by jrb:

```

[LAMBDA (NAME TYPE SOURCE OPTIONS)

```

::: Take a whack at getting the definition of NAME. Masterscope assumes a bijection between names and analyzable things; it caches that relationship in ::: the FPTYPE table

```

(LET (TABLEFPTYPE)
  (if [SETQ TABLEFPTYPE (CAR (GETRELATION NAME (PARSERELATION 'FPTYPE))
    then (APPLY* (ffetch (MSANALYZABLE GETDEF-FN) of (ASSOC TABLEFPTYPE MSFNTPES))
                 NAME TABLEFPTYPE SOURCE OPTIONS)
    else (for FPTYPE in MSFNTPES bind RESULT when (SETQ RESULT (APPLY* (ffetch (MSANALYZABLE GETDEF-FN)
                                                                                               of FPTYPE)
                                                                           NAME TYPE SOURCE OPTIONS))
          do (PUTTABLE NAME (LIST (ffetch (MSANALYZABLE FILEPKGNAME) of FPTYPE))
                                (CADR (FASSOC 'FPTYPE MSDATABASELST)))
          (RETURN RESULT)
      finally NIL])

```

(MSNOTICEFILE

; Edited 8-Apr-88 12:00 by jrb:

```

[LAMBDA (FILE)
  (DECLARE (GLOBALVARS MSHASHFILE LOADDBFLG))
  (PROG (FULL COMS TEM)
    LP (COND
      ((SETQ TEM (FASSOC FILE MSFILELST))
       (RETURN TEM))
      (OR COMS (SETQ COMS (FILECOMS FILE)))
      (SETQ FULL (FINDFILE FILE T))
    [COND
      ((NOT (FMEMB FILE FILELST))

```

(\* two possibilities%: either FILE is something like <LISP>FOO or it has not been loaded yet)

```

[COND
  ((AND (NOT FULL)
        (EQ FILE (NAMEFIELD FILE T)))
   (COND
     ((LISTP (GETATOMVAL COMS))
      (GO DUMMY)
      (* dummy or new file since COMS set but not on filest)
      (* either the file has never been loaded, or an explicit <FOO>
      was given)
     (OR FULL (ERROR FILE "not found"))
     [SETQ COMS (FILECOMS (SETQ FILE (NAMEFIELD FULL T))
     (OR (AND [EQ FULL (CDAR (GETPROP FILE 'FILEDATES])
              (LISTP (EVALV COMS)))
         (COND
           ((EQ 'Y (ASKUSER DWIMWAIT 'Y (LIST "should I LOADFROM" FULL)))
            (RESETVARS [(LOADDBFLG (COND
                               (MSHASHFILE LOADDBFLG)
                               (T 'NO)
                               (LOADFROM FULL)))
                       (T (ERROR!])
         (EQ [CDAR (SETQ TEM (GETPROP FILE 'FILE)
              'Compiled)

```

(\* If only the compiled version of the file has been loaded, still want to know about GLOBALVARS or other things)

```

(LOADVARS ' ((DECLARE%: -- DONTCOPY --))

```

```

      (OR (AND (SETQ FULL (GETP FILE 'FILEDATES))
              (INFILEP (CDAR FULL)))
          FILE))
      (/RPLACD (CAR TEM)
              'COMPILED]
DUMMY
      (RETURN (OR (FASSOC FILE MSFILELST)
                 (CAR (SETQ MSFILELST (CONS (CONS FILE COMS)
                                           MSFILELST])))

```

(MSSHOWUSE

```

[LAMBDA (SHOWFN SHOWTYPE SHOWSET SHOWEDIT IFCANT EDITCOMS) ; Edited 1-Jun-88 15:52 by jrb:
  (PROG (DEF ANYFOUND)
    (COND

```

```

      ([SETQ DEF (OR (MSGETDEF SHOWFN (AND (fetch (MSSETPHRASE KNOWN) of SHOWSET)
                                              (fetch (MSSETPHRASE TYPE) of SHOWSET))
                  (COND
                    ((EQ SHOWEDIT 'SHOW)
                     '?)
                    (T 'CURRENT))
                  ' (NOERROR NODWIM NOCOPY))
                (AND (EQ SHOWEDIT 'EDIT)
                     (LET ((FILE (EDITLOADFNS? SHOWFN)))
                       (COND
                        (FILE (LOADFNS SHOWFN FILE 'PROP)
                              (GETPROP SHOWFN 'EXPR]

```

; was (MSGETDEF SHOWFN IFCANT (EQ SHOWEDIT  
; (QUOTE SHOW)))  
; The SHOW command does not need to save

```

      (MSUPDATEFN1 SHOWFN DEF (LIST SHOWTYPE [FUNCTION (LAMBDA (ITEM SS SE PRNT INCLISP)

```

```

        (COND
          ((MSMEMBSET ITEM SS)
            (COND
              ((NOT ANYFOUND)
               (TAB 0 0 T)
               (PRIN2 SHOWFN)
               (PRIN1 " :
                    ")))
            (SETQ ANYFOUND
              (CONS (CONS PRNT
                        (AND INCLISP
                            (NOT (MSFIND INCLISP
                                         PRNT))
                                INCLISP))
                    ANYFOUND))

```

```

          (COND
            ([AND (EQ SE 'SHOW)
                  (NOT (FASSOC PRNT (CDR ANYFOUND)

```

:: The EDIT command works by collecting a list of the expressions, and then doing a (\*ORF\* (= . lst1)  
:: (= . lst2)) --- if within a CLISP translation (determined by the binding of the INCLISP variable) then  
:: want to point at the CLISP if the expression is not actually embedded in the expression

```

          (SPACES 3)
          (LVLPRINT PRNT (OUTPUT)
                    2)

```

```

          (COND
            ((CDAR ANYFOUND)

```

; This is under a clisp

```

            (PRIN1 " {under "})
            (LVLPRIN2 INCLISP (OUTPUT)
                      2)
            (PRIN1 " }
              "])

```

```

          (SHOWSET SHOWEDIT)))
      (T (printout T "Can't find a definition for " SHOWFN "!" T)
         (RETURN)))

```

(COND

```

  ((NOT ANYFOUND)
   (RETURN))

```

```

  ((EQ SHOWEDIT 'EDIT)

```

```

    [MAPC ANYFOUND (FUNCTION (LAMBDA (X)
                              (FRPLNODE X '== (OR (CDR X)
                                                    (CAR X)

```

```

          (SETQ ANYFOUND (CONS '*ANY* ANYFOUND))

```

```

          (PRINT [APPLY* 'MSEDITF SHOWFN (LIST 'BIND ' (E (SETQ %#1)

```

```

                    T)
                (LIST 'F ANYFOUND T)
                (LIST 'LPQ (LIST 'IF ' (NEQ (%##)

```

```

                    %#1)
                [LIST ' (ORR (P)
                            NIL)

```

```

                    ' (S %#1)

```

```

          (COND

```

```

            (EDITCOMS (CONS 'BIND EDITCOMS))
            (T 'TTY%:]

```

NIL)



**(MSCOLLECTDATA**

(\* Imm "30-OCT-80 10:00")

```
[LAMBDA (TNAME FLG)
  (COND
    ((LISTP TNAME)
     (SELECTQ (CAR TNAME)
              (- (LDIFFERENCE (MSCOLLECTDATA (CADR TNAME)
                                             T)
                             (MSCOLLECTDATA (CADDR TNAME)
                                             T)))
              (+ (UNION (MSCOLLECTDATA (CADR TNAME)
                                       T)
                      (MSCOLLECTDATA (CADDR TNAME)
                                       T))))
     (SHOULDNT 2)))
  (T (PROG NIL
        (RETURN (MSCOLLECTDATA (CADR (OR (AND (NULL FLG)
                                             (FASSOC TNAME TABLE.TO.NOTICED)))
                                       (RETURN (CDR (FASSOC TNAME FNDATA))))))
    )
```

(DEFINEQ

**(UPDATECHANGED**

(\* Imm "16-JUL-78 05:07")
(\* Update all functions marked as changed)

```
[LAMBDA NIL
  (MSINIT)
  (MAPHASH MSCHANGEDARRAY (FUNCTION UPDATECHANGED1))
  NIL])
```

**(UPDATECHANGED1**

; Edited 27-Jan-88 16:49 by jrb:

```
[LAMBDA (VAL KEY)
  (COND
    [(OR (EQ VAL T)
         (TESTRELQ KNOWN KEY)
         (TESTRELQ (CALL NOTERROR)
                   KEY T))
     (COND
       ([SETQ VAL (MSGETDEF KEY NIL ' ? ' (NOERROR NOCOPY MASTERSCOPE)
                           (MSUPDATEFN1 KEY VAL NIL T))
        (T (printout T KEY " disappeared!" T)
            (MSERASE (LIST KEY)
                     (PUTHASH KEY NIL MSCHANGEDARRAY))
        )
    )
```

**(RPAQQ TABLE.TO.NOTICED**

```
((BIND (- (- (- (- (+ BIND ARG)
                     REF)
                   SMASH)
             SET)
        TEST))
 (REFFREE (- (- (- REFFREE SETFREE)
                 SMASHFREE)
           TESTFREE))
 (REF (- (- (- REF SET)
             SMASH)
        TEST))
 (PREDICATE (- PREDICATE CALL))
 (EFFECT (- (- EFFECT CALL)
            PREDICATE))
 (CALL (- CALL NLAMBDA))
 (0 TYPE)
 (APPLY (+ APPLY STACK))
 (ARGS ARG))
```

(DEFINEQ

**(MSCLOSEFILES**

(\* Imm "24-JUN-78 17:18")
(\* this is RESETSAVE'd from MSGETDEF to close any files that

```
[LAMBDA NIL
  MSGETDEF leaves open)
  (for X in MSOPENFILES when (AND (NOT (CADR X))
                                   (OPENP (CADDR X)))
    do (CLOSEF (CADDR X)))
  (SETQ MSOPENFILES])
```

(RPAQQ MSFILELST NIL)

(RPAQQ MSOPENFILES NIL)

(RPAQQ MSPRINTFLG %.)

(RPAQQ MSPRINTCNT 0)

(ADDTOVAR MSHASHFILE )

(ADDTOVAR ANALYZEUSERFNS )

(RPAQQ MSAUXCOMS

```
(
  (COMS                                     ; things which are not in the 'main stream' of MASTERSCOPE
    (FNS MSDESCRIBE MSDESCRIBE1 FMAPRINT) ; Describe command
    (ADDVARS (DESCRIBELST))
    (GLOBALVARS DESCRIBELST))
  (COMS                                     ; Print help file
    (FNS MSPRINTHELPPFILE)
    (VARS MSHELPPFILE))
  (COMS                                     ; templates --- export TEMPLATE GETTEMPLATE
    (FNS TEMPLATE GETTEMPLATE SETTEMPLATE) ; SETTEMPLATE
    (FILEPKGCOMS TEMPLATES))
  (COMS                                     ; Fns that let users extend Masterscope
    (FNS ADDTEMPLATEWORD MSADDANALYZE MSADDMODIFIER MSADDRELATION MSADDTYPE)
    (INITVARS (MSCHECKFNS NIL))
    (GLOBALVARS MSCHECKFNS MSANALYZEFNS MSUSERVBTABLES))))
```

:: things which are not in the 'main stream' of MASTERSCOPE

:: Describe command

(DEFINEQ

(MSDESCRIBE

```
[LAMBDA (FN SN)                                     (* Imm "22-Jul-85 18:16")
                                                    (* Prints function name, arguments, local and free variables.
                                                    etc)
                                                    (* Make FN available to user DESCRIBELST forms)
```

(DECLARE (SPECVARS FN)

(PROG (GLOBALS FREES ARGS LINE)

```
[SETQ ARGS (COND
  ((SETQ ARGS (GETRELQ ARGS FN))
```

(\* The args in the argtable have precedence, even if the function is resident, cause they correspond to what was actually analyzed.)

(\* T is for an arglist of NIL)

```
(AND (NEQ ARGS T)
  ARGS)
((GETD FN)
 (SMARTARGLIST FN]
(printout NIL " (" .FONT BOLDFONT .P2 FN .FONT DEFAULTFONT)
(FMAPRINT ARGS NIL " " ")")
(OR (TESTRELQ KNOWN FN)
  (PRIN1 " (not analyzed)" T))
(COND
  ([AND [OR (HARRAYP SN)
    (HARRAYP (CAR (LISTP SN)
      (SMALLP (SETQ LINE (GETHASH FN SN)
        (TAB 45 T)
        (PRIN1 " {line "
        (PRIN1 (ABS LINE))
        (PRIN1 "};"))))
  (TERPRI)
  (MSDESCRIBE1 (GETRELQ (CALL NOTERROR)
    FN)
    ' "calls: "
  (MSDESCRIBE1 (GETRELQ (CALL NOTERROR)
    FN T)
    ' "called by:")
  (MSDESCRIBE1 (for VAR in (GETRELQ BIND FN) when (NOT (EQMEMB VAR ARGS)) collect VAR)
    ' "binds: "
  [for VAR in (GETRELQ (USE FREELY)
    FN)
  do (COND
    ((OR (FMEMB VAR GLOBALVARS)
      (GETPROP VAR 'GLOBALVAR))
      (SETQ GLOBALS (CONS VAR GLOBALS)))
    (T (SETQ FREES (CONS VAR FREES]
  (MSDESCRIBE1 FREES ' "uses free:")
  (MSDESCRIBE1 GLOBALS ' "globals: ")
  (MSDESCRIBE1 (GETRELQ (USE FIELDS)
    FN)
    ' "fields: ")
  (for D L in DESCRIBELST when (SETQ L (EVAL (CADR D))) do (MSDESCRIBE1 L (CAR D)))
  (TERPRI])
```

(MSDESCRIBE1

[LAMBDA (LST STR)

```
(* Imm " 9-AUG-77 04:45")
(* Imm%: 15 NOV 75 2248)
```

```
(COND
  (LST (SPACES 2)
        (PRIN1 STR)
        (SPACES 1)
        (PROG (LL P)
              (COND
                ((NULL LST)
                 (GO EXIT))
                ((NLISTP LST)
                 (PRIN2 LST)
                 (GO EXIT)))
                (SETQ LL (LINELENGTH))
                (SETQ P (POSITION))
                LP (COND
                  ((IGREATERP (IPLUS (POSITION)
                                     5
                                     (NCHARS (CAR LST))))
                   LL)
                  (TAB P)))
                  (PRIN2 (CAR LST))
                  (COND
                    ((NULL (SETQ LST (CDR LST)))
                     (GO EXIT)))
                    (PRIN1 '%,))
                    (GO LP)
                  EXIT
                    (TERPRI]))
```

**(FMAPRINT**

(\* Imm%: 28 OCT 75 757)

```
[LAMBDA (LST FILE LEFT RIGHT SEP)
  (PROG NIL
    (AND LEFT (PRIN1 LEFT FILE))
    (OR SEP (SETQ SEP '% ))
    (COND
      ((NULL LST)
       (GO EXIT))
      ((NLISTP LST)
       (PRIN2 LST)
       (GO EXIT)))
    LP (PRIN2 (CAR LST)
              FILE)
      (COND
        ((NULL (SETQ LST (CDR LST)))
         (GO EXIT)))
        (PRIN1 SEP FILE)
        (GO LP)
      EXIT
        (AND RIGHT (PRIN1 RIGHT FILE))
```

)

(ADDOVAR DESCRIBELST )

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS DESCRIBELST)

)

:: Print help file

(DEFINEQ

**(MSPRINTHELPPFILE**

(\* Imm "20-JAN-79 13:48")

```
[LAMBDA NIL
  (PROG (FL)
    [SETQ FL (OR (INFILEP MSHELPPFILE)
                 (RETURN (PROGN (printout T "Sorry, HELP file not available!" T)
                               NIL))]
    (SETQ FL (INPUT (INFILE FL)))
    (RESESAVE NIL (LIST 'CLOSEF FL))
    (COPYBYTES FL T 0 (GETEOFPTR FL))
```

)

(RPAQQ MSHELPPFILE <LISP>MASTERSCOPE.SYNTAX)

:: templates --- export TEMPLATE GETTEMPLATE SETTEMPLATE

(DEFINEQ

**(TEMPLATE**

(\* Imm "23-DEC-78 14:06")

```
[LAMBDA (FN TEMPLATE FLG)
```

(\* MSTEMPLATES is the hash table for pre-defined templates.  
USERTEMPLATES contains user defined templates. The split is so that DUMPDATABASE can dump the templates as well

check for MSDATABASE is so CALLS doesn't need to init database)

```
(PROG [(OLD (OR (GETHASH FN USERTEMPLATES)
                (GETHASH FN MSTEMPLATES)
                (COND
                 ((EQ OLD T)
                  (SETQ OLD)))
                (COND
                 ((AND (NOT (EQUAL TEMPLATE OLD))
                      (NEQ TEMPLATE T))
                  [COND
                   ((NOT FLG)
                    [AND FILEPKGFLG (MARKASCHANGED FN 'TEMPLATES (NOT (NULL OLD)
                    (AND MSDATABASELST (MSMARKCHANGED (GETRELATION FN '(CALL DIRECTLY)
                                                         T)
                                                         'FNS]
                    (PUTHASH FN (COND
                                ((NLISTP TEMPLATE)
                                 (SELECTQ TEMPLATE
                                  (MACRO TEMPLATE)
                                  (NIL (AND (GETHASH FN MSTEMPLATES)
                                           T))
                                  (ERROR TEMPLATE "Invalid template"))))
                                (T TEMPLATE))
                                USERTEMPLATES)))
                    (RETURN OLD])
```

**(GETTEMPLATE**

```
[LAMBDA (FN)
 (SETTEMPLATE FN T)]
```

(\* Imm "9-AUG-77 06:20")

**(SETTEMPLATE**

```
[LAMBDA (FN TEMPLATE)
 (PROG [(OLD (COPY (TEMPLATE FN (COND
```

(\* Imm "16-FEB-78 14:56")

```
((NLISTP TEMPLATE)
 (SELECTQ TEMPLATE
  (EVAL '(CALL |..| EVAL))
  ((NIL T MACRO)
   TEMPLATE)
  (ERROR TEMPLATE "Invalid template")))
(T (SELECTQ (CAR TEMPLATE)
 (MACRO TEMPLATE)
 (! (CDR TEMPLATE))
 (CONS 'CALL (AND (OR (CAR TEMPLATE)
                      (CDR TEMPLATE))
        TEMPLATE]
```

```
VAL)
[SETQ VAL (COND
 ((NLISTP OLD)
  OLD)
 (T (SELECTQ (CAR OLD)
 (MACRO OLD)
 (CALL (OR (CDR OLD)
           (CONS)))
 (CONS '! OLD]
[OR (EQ TEMPLATE T)
 (AND LISPXHIST (UNDOSAVE (LIST 'SETTEMPLATE FN OLD]
 (RETURN VAL])
```

)

```
[PUTDEF 'TEMPLATES 'FILEPKGCOMS ' ((COM MACRO [X (P * (MAPCAR 'X (FUNCTION (LAMBDA (FN)
                                                                    (LIST 'SETTEMPLATE (KWOTE FN)
                                                                    (KWOTE (GETTEMPLATE FN)
                                                                    CONTENTS NIL))
 (TYPE DESCRIPTION "masterscope templates"]
```

:: Fns that let users extend Masterscope

(DEFINEQ

**(ADDTEMPLATEWORD**

```
[LAMBDA (WORD)
```

(\* smL "27-Nov-85 17:49")

(\* \* Add a new word that can be used in TEMPLATES. This really means add a new MasterScope table.)

```
(MSINIT)
(if (NOT (ASSOC WORD MSFNDATA))
 then (PUTASSOC WORD NIL MSFNDATA))
(if (NOT (ASSOC WORD MSDATABASELST))
 then (PUTASSOC WORD (CONS (MAKETABLE 2)
                          (MAKETABLE 2))
              MSDATABASELST))
(if (NOT (ASSOC WORD MSDATABASEINIT))
 then (PUTASSOC WORD (CONS 2 2)
```



MSDATABASEINIT])

**(MSADDANALYZE**

[LAMBDA (PLURAL SINGLE FILETYPE GETDEF-FN MARKCHANGED-FN) ; Edited 16-Jun-88 10:35 by jrb:

(\* \* Defines a new MasterScope datatype)

```
[for word in (LIST PLURAL SINGLE) do (LET ((oldDef (GETHASH word MSWORDS)))
  (if oldDef
    then (PUTASSOC 'TYPE PLURAL oldDef)
    else (PUTHASH word (LIST (CONS 'TYPE PLURAL))
      MSWORDS])
  )
)
```

:: MSANALYZEFNS is bogus and is hereby removed. (PUTHASH PLURAL ANALYZEFN MSANALYZEFNS)

:: JRB -

```
(if FILETYPE
  then (LET ((oldEntry (ASSOC FILETYPE MSFNTPES)))
    (if oldEntry
      then (replace (MSANALYZABLE SETNAME)
        oldEntry PLURAL)
      (replace (MSANALYZABLE GETDEF-FN)
        oldEntry GETDEF-FN)
      (replace (MSANALYZABLE MARKCHANGED-FN)
        oldEntry MARKCHANGED-FN)
      else (push MSFNTPES (create MSANALYZABLE
        FILEPKGNAME _ FILETYPE
        SETNAME _ PLURAL
        GETDEF-FN _ GETDEF-FN
        MARKCHANGED-FN _ MARKCHANGED-FN))
    )
  )
)
```

**(MSADDMODIFIER**

[LAMBDA (RELATION MODIFIERS TABLES) (\* smL "16-Dec-85 15:39")

(\* \* Define a new modifier to a MasterScope relation, telling what tables should be combined to determine the modified relation)

```
(SETQ TABLES (MKLIST TABLES))
(SETQ MODIFIERS (MKLIST MODIFIERS))
(MSINIT)
(for adverb in MODIFIERS bind oldWordDef do (SETQ oldWordDef (ASSOC 'V (GETHASH adverb MSWORDS)))
  (if oldWordDef
    then (PUTASSOC 'V [CONS adverb (CONS RELATION
      (MKLIST (CDDR oldWordDef)
        (GETHASH adverb MSWORDS))
      )]
    else (PUTHASH adverb (CONS (CONS 'V (LIST adverb RELATION))
      (GETHASH adverb MSWORDS))
      MSWORDS)))
  )
  (PUTHASH RELATION (CONS (CONS MODIFIERS TABLES)
    (GETHASH RELATION MSUSERVBTABLES))
    MSUSERVBTABLES)
  (for table in TABLES do (ADDTEMPLATEWORD table])
)
```

**(MSADDRELATION**

[LAMBDA (RELATION TABLES) (\* smL "16-Dec-85 14:55")

(\* \* Let the user define a new MasterScope relation. -  
RELATION is a list of ROOT PRESENT PARTICIPLE and PAST conjugations of the new relation.  
They can then be used in MasterScope commands to specify relations.

-  
TABLES is a list of new MasterScope database tables. These tables can then be used in MasterScope templates.  
TABLES defaults to the ROOT of the relation.)

```
(LET ((ROOT (CAR RELATION)))
  (MSSETUP (LIST RELATION))
  [MSADDMODIFIER ROOT ' (NIL)
    (MKLIST (MKLIST (OR TABLES ROOT)
      ROOT])
  )
)
```

**(MSADDTYPE**

[LAMBDA (TYPE TABLES HOWUSED SYNONYMS) (\* smL "16-Dec-85 15:35")

(\* \* Defines the TYPE as the union of the TABLES so you can use phrases like "USE foo AS A <TYPE>" or "USE THE <TYPE> foo")

```
[SETQ HOWUSED (MKLIST (OR HOWUSED 'USE))
(SETQ SYNONYMS (MKLIST SYNONYMS))
(SETQ TABLES (MKLIST TABLES))
(MSINIT)
(for typeWord in (CONS TYPE SYNONYMS) bind oldWordDef
  do (SETQ oldWordDef (GETHASH typeWord MSWORDS))
    (if oldWordDef
      then (PUTASSOC 'TYPE TYPE oldWordDef)
      else (SETQ oldWordDef (LIST (CONS 'TYPE TYPE)))
    )
  )
)
```

```

    (PUTHASH typeWord oldWordDef MSWORDS))
  (PUTASSOC 'AS [CONS TYPE (APPEND HOWUSED (CDDR (ASSOC 'AS oldWordDef)
    oldWordDef))
  (MSADDMODIFIER 'USE TYPE TABLES])

```

)

(RPAQ? MSCHECKFNS NIL)

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS MSCHECKFNS MSANALYZEFNS MSUSERVBTABLES)

)

(RPAQQ MSDBCOMS

```

  [
    ; functions for manipulating the data base
    (FNS MSMARKCHANGE1 MSINIT GETVERBTABLES MSSTOREDATA STORETABLE)
    (ADDVARS (MSCHANGEDARRAY)
      (MSDATABASELST))
    (INITVARS (MSDBEMPTY T))
    (VARS MSDATABASEINIT NODUMPRELATIONS)
    (FNS PARSERELATION PARSERELATION1 GETRELATION MAPRELATION TESTRELATION)
    (COMS ; table lookup functions
      (FNS ADDHASH SUBHASH MAKEHASH MSREHASH EQMEMBHASH)
      [P (MAPC ' ((GETHASH GETTABLE)
        (GETHASH TESTTABLE)
        (PUTHASH PUTTABLE)
        (ADDDHASH ADDTABLE)
        (SUBHASH SUBTABLE)
        (MAPHASH MAPTABLE)
        (MAKEHASH MAKETABLE)
        (EQMEMBHASH EQMEMBTABLE))
        (FUNCTION (LAMBDA (X)
          (MOVD? (CAR X)
            (CADR X]
      (FNS MSVBTABLES MSUSERVBTABLES)
      (INITVARS (MSUSERVBTABLES (MAKETABLE 2))
        (MSANALYZEFNS (MAKETABLE 2)))
      (FNS BUILDGETRELQ BUILDTESTRELQ)
      (DECLARE%: DONTCOPY (MACROS GETRELQ TESTRELQ)))
    (COMS ; erase function
      (FNS MSERASE))
    (COMS ; dump data base
      (FNS DUMPDATABASE DUMPDATABASE1 READATABASE)
      (VARS DATABASECOMS))
    (ADDVARS (GAINSPACEFORMS (MSDATABASELST "erase current Masterscope database" (%. ERASE])

```

:: functions for manipulating the data base

(DEFINEQ

(MSMARKCHANGE1

[LAMBDA (FNS FLG) (\* rmk%: "19-FEB-81 14:53")

(\* mark the selected functions as "changed" -  
FLG=T in MSCHANGEDARRAY means to complain if the definition can't be found, 0 means give up quietly.)

(OR FLG (SETQ FLG T))  
(for FN inside FNS do (PUTHASH FN FLG MSCHANGEDARRAY))

(\* This isn't undone by simply restoring the previous entry in the hash array because the user might have asked a question that caused the functions to be reanalyzed. Against this possibility, we "undo" by re-marking the functions for reanalysis.)

(UNDOSAVE (LIST (FUNCTION MSMARKCHANGE1)
 FNS FLG))

(MSINIT

[LAMBDA (DATABASE) (\* lmm "29-Jul-85 21:06")

(\* This function should be called at the beginning of any function which accesses the data base)

(COND
 ((OR (NULL MSDATABASELST)
 (LISTP DATABASE))
 (SETQ MSDATABASELST)
 (pushnew MARKASCHANGEDFNS 'MSMARKCHANGED)
 (SETQ MSCHANGEDARRAY (HASHARRAY 128))

(\* MSDATABASEINIT is a pattern for how the data base list should look.  
The data base is stored in multiple hash tables. These hash tables are pointed to both by a list, MSDATABASELST which is an a-list of (TableName ForwardTable . BackTable) while the array, MSDATABASE, is an array of the same length as the list, with ELT's the forward htables, and ELTD's the back tables.)

(OR DATABASE (SETQ MSDBEMPTY))
[SETQ MSDATABASELST (for X in MSDATABASEINIT
 collect [CONS (CAR X)

```

(CONS [OR (CADR (ASSOC (CAR X)
                     DATABASE))
      (SELECTQ (CAR X)
               (USERTEMPLATES
                USERTEMPLATES)
               (MAKETABLE (CADR X)
                           (CAR X)
                           (COND
                            ((FIXP (CDDR X))
                             (OR (CDDR (ASSOC (CAR X)
                                                DATABASE))
                                 (MAKETABLE (CDDR X)
                                           (CAR X)
                                           T)))
                            (T (CDDR X]
      finally (SETQ MSFILETABLE (ASSOC 'FILE $$VAL))
              (SETQ MSARGTABLE (ASSOC 'ARGS $$VAL]
T])

```

**(GETVERBTABLES**

```

[LAMBDA (ROOT MODIFIER) (* Imm "28-FEB-79 16:08")
  (for Y in (OR (MSVBTABLES ROOT MODIFIER)
               (SHOULDN'T 3))
    collect (COND
             [(LISTP Y)
              (LIST (CDDR (FASSOC (CAR Y)
                                 MSDATABASELST]
                    (T (CDR (FASSOC Y MSDATABASELST]))

```

**(MSSTOREDATA**

```

[LAMBDA (FNAME FNDATA) (* Imm " 1-JUN-81 23:19")
  (PROG [NEWREL (KWN (PARSERELATION 'KNOWN]
        (SETQ MSDBEMPTY NIL) (* Database for FNAME about to become inconsistant -
                               mark it as changed)

    (PUTHASH FNAME T MSCHANGEDARRAY)

    (** Now update the database)

    (for TAB in MSDATABASELST when (AND (NOT (FMEMB (CAR TAB)
                                                    NODUMPRELATIONS))
                                         (NEQ (CDDR TAB)
                                               T))
      do (SETQ NEWREL (MSCOLLECTDATA (CAR TAB))
          (STORETABLE FNAME TAB NEWREL))
    [OR (TESTRELATION FNAME KWN)
        (PUTTABLE FNAME T (CADR (FASSOC 'NOBIND MSDATABASELST]

    (* Table NOBIND is for those functions which don't do very much.
    The idea is that the test that a function has been analyzed is whether it binds variables are calls functions, etc.
    However, for those functions which have no such entries, (e.g. their definition is
    (LAMBDA NIL NIL)) need to still be able to know that they were.)

    (PUTHASH FNAME NIL MSCHANGEDARRAY])

```

**(STORETABLE**

```

[LAMBDA (KEY TABLST VALUE) (* Imm "10-APR-81 08:46")
  (PROG [(OLDREL (GETTABLE KEY (CADR TABLST]
            (PUTTABLE KEY VALUE (CADR TABLST))
        (COND
         ((CDDR TABLST)
          (for Z in VALUE do
            (ADDTABLE Z KEY (CDDR TABLST)))
          (for Z in OLDREL do
            (* However, we must rely on the previous value to tell who values must be DELETED from)
            (AND (NOT (FMEMB Z VALUE))
                 (SUBTABLE Z KEY (CDDR TABLST]))

```

```

)
(ADDTOVAR MSCHANGEDARRAY )
(ADDTOVAR MSDATABASELST )
(RPAQ? MSDBEMPTY T)
(RPAQ? MSDATABASEINIT
 ((CALL 25 . 50)

```

```

(BIND 10 . 10)
[NLAMBDA 10 . 10]
(NOBIND 10)
(RECORD 20 . 10)
(CREATE 2 . 2)
(FETCH 10 . 10)
(REPLACE 10 . 10)
(REFFREE 10 . 1)
(REF 10 . 25)
(SETFREE 1 . 1)
(SET 20 . 30)
(SMASHFREE 1 . 1)
(SMASH 1 . 1)
(PROP 1 . 1)
(TEST 1 . 1)
(TESTFREE 1 . 1)
(PREDICATE 10 . 10)
(EFFECT 10 . 10)
(CLISP 10 . 10)
(SPECVARS 10 . 10)
(LOCALVARS 10 . 10)
(APPLY 10 . 10)
(ERROR 10 . 10)
(LOCALFREEVARS 10 . 10)
(CONTAINS 10 . 10)
(FILE 10)
(ARGS 10)
(USERTEMPLATES NIL . T)
(0 10 . 10)
(FPTYPE 10 . 10)
(KEYACCEPT 2 . 2)
(KEYSPECIFY 2 . 2)
(KEYCALL 2 . 2)
(FLET 2 . 2)
(LABEL 2 . 2)
(MACROLET 2 . 2)
(COMPILER-LET 2 . 2)
(SENDNOTSELF 2 . 2)
(SENDFREE 2 . 2)
(IMPLEMENT 2 . 2)
(GETNOTSELF 2 . 2)
(GETSELF 2 . 2)
(GETCVSELF 2 . 2)
(GETCVNOTSELF 2 . 2)
(PUTNOTSELF 2 . 2)
(PUTSELF 2 . 2)
(PUTCVSELF 2 . 2)
(PUTCVNOTSELF 2 . 2)
(OBJECT 2 . 2)))

```

(RPAQQ **NODUMPRELATIONS** (CONTAINS FILE))

(DEFINEQ

**(PARSERELATION**

```

[LAMBDA (RELATION)
  (MSINIT)
  (COND

```

(\* Imm "11-Jul-86 15:50")

```

    ((EQ (CAR (LISTP RELATION))
         'TABLES)
     RELATION)
    (T (CONS 'TABLES (for Y in (PARSERELATION1 RELATION) collect (COND
                                                                    [(LISTP Y)
                                                                     (CDR (CDR (FASSOC (CAR Y)
                                                                MSDATABASELST)
                                                                (T (CDR (FASSOC Y MSDATABASELST))

```

**(PARSERELATION1**

```

[LAMBDA (ROOT MOD TAIL)
  (COND

```

(\* Imm "30-DEC-78 17:06")

```

    [TAIL (APPLY* (SELECTQ (CAR TAIL)
                          (ANDNOT (FUNCTION LDIFFERENCE))
                          (AND (COND
                                ((EQ (CADR TAIL)
                                     'NOT)
                                 (SETQ TAIL (CDR TAIL))
                                 (FUNCTION LDIFFERENCE))
                                (T (FUNCTION INTERSECTION))))
                  (OR (FUNCTION UNION)
                      (ERROR TAIL '?))
                  (PARSERELATION1 ROOT MOD)
                  (PARSERELATION1 (CADR TAIL)
                                   (CDDR TAIL)
                                   (CDDR TAIL]
    ((LISTP ROOT)
     (PARSERELATION1 (CAR ROOT)
                     (CDR ROOT)))

```

```

[ (LISTP MOD)
  (SELECTQ (CAR MOD)
    ((A AS AN FOR)
      (PARSERELATION1 ROOT (CDR MOD)))
    ((AND OR ANDNOT)
      (PARSERELATION1 ROOT NIL MOD))
    (PARSERELATION1 ROOT (CAR MOD)
      (CDR MOD))
  )
(T (OR (MSVBTABLES ROOT MOD)
  [MSVBTABLES (GETWORDTYPE ROOT 'S)
    (CAR (OR (GETWORDTYPE MOD 'V)
      (GETWORDTYPE MOD 'AS)
      (GETWORDTYPE MOD 'FOR)
      (ERROR MOD '?])
    )
    (ERROR ROOT '?])
  )

```

**(GETRELATION**

```

[LAMBDA (ITEM RELATION INVERTED)
  (PROG (VAL)
    (for TABLE in [CDR (COND
      ((EQ (CAR (LISTP RELATION))
        'TABLES)
        RELATION)
      (T (PARSERELATION RELATION])
    )
    do (SETQ VAL (UNION [GETTABLE ITEM (COND
      (INVERTED (COND
        ((LITATOM (CDR TABLE))
          (ERROR RELATION "CAN'T BE INVERTED")))
      (CDR TABLE))
      (T (CAR TABLE]
        VAL)))
    (RETURN VAL])
  )

```

(\* Imm "11-Jul-86 15:51")

**(MAPRELATION**

```

[LAMBDA (RELATION MAPFN)
  (DECLARE (SPECVARS MAPZ MAPW MAPFN2 MAPFN))
  (PROG ((MAPZ (NARGS MAPFN))
    (MAPW (PARSERELATION RELATION)))
    (MAP (CDR MAPW)
      (FUNCTION (LAMBDA (MAPFN2)
        (MAPTABLE (CAAR MAPFN2)
          (FUNCTION (LAMBDA (DUMMY MAPX)
            (OR [SOME (CDR MAPFN2)
              (FUNCTION (LAMBDA (HT2)
                (TESTTABLE MAPX (CAR HT2]
            )
          )
        )
      )
      (COND
        ((EQ MAPZ 1)
          (APPLY* MAPFN MAPX))
        (T (MAPC (GETRELATION MAPX MAPW)
          (FUNCTION (LAMBDA (Z)
            (APPLY* MAPFN MAPX Z])
          )
        )
      )
    )
  )

```

(\* Imm "21-SEP-78 04:20")

**(TESTRELATION**

```

[LAMBDA (ITEM RELATION ITEM2 INVERTED)
  (AND [SOME [CDR (COND
    ((EQ (CAR RELATION)
      'TABLES)
      RELATION)
    (T (PARSERELATION RELATION])
  )
  (FUNCTION (LAMBDA (TABLE)
    (COND
      [ITEM2 (FMEMB ITEM2 (GETTABLE ITEM (COND
        (INVERTED (CDR TABLE))
        (T (CAR TABLE]
      )
      (T (TESTTABLE ITEM (COND
        (INVERTED (CDR TABLE))
        (T (CAR TABLE]
      )
    )
  )

```

(\* Imm "25-JUN-78 01:16")

)

:: table lookup functions

(DEFINEQ

**(ADHASH**

```

[LAMBDA (ITEM VAL ARRAY)
  (PROG ((OV (GETHASH ITEM ARRAY))
    (COND
      (OV (OR (FMEMB VAL OV)
        (NCONC1 OV VAL)))
      (T (PUTHASH ITEM (LIST VAL)
        ARRAY])
    )
  )

```

(\* Imm "10-JUL-78 03:03")

(\* Add VAL to the hash-key of ITEM in ARRAY)

**(SUBHASH**

[LAMBDA (ITEM VAL ARRAY) (\* Imm "10-JUL-78 03:03")  
(PROG ((OV (GETHASH ITEM ARRAY)))  
(AND OV (OR (DREMOVE VAL OV)  
(PUTHASH ITEM NIL ARRAY]))

**(MAKEHASH**

[LAMBDA (N) (\* rmk%: " 3-Jan-84 21:31")  
(HASHARRAY N (FUNCTION MSREHASH))

**(MSREHASH**

[LAMBDA (HA) (\* rmk%: "30-Dec-83 11:45")

(\* The hash tables in the database rehash using this algorithm;  
they increase size by 25% + 50 This insures that even though some tables start out small  
(e.g. 1 or 2 elements) they will rehash to larger ones.)

(IPLUS (IQUOTIENT (ITIMES 5 (HARRAYSIZE HA))  
4)  
50))

**(EQMEMBHASH**

[LAMBDA (X V H) (\* rmk%: "10-JUN-79 21:00")  
(MEMB V (GETHASH X H]) (\* Provided in case MSHASH is loaded without MSSWAP)

)

[MAPC ' ((GETHASH GETTABLE)  
(GETHASH TESTTABLE)  
(PUTHASH PUTTABLE)  
(**ADDDHASH** ADDTABLE)  
(**SUBHASH** SUBTABLE)  
(MAPHASH MAPTABLE)  
(**MAKEHASH** MAKETABLE)  
(**EQMEMBHASH** EQMEMBTABLE))  
(FUNCTION (LAMBDA (X)  
(MOVD? (CAR X)  
(CADR X]

(DEFINEQ

**(MSVBTABLES**

[LAMBDA (VERB MOD) ; Edited 30-Jun-87 10:32 by jrb:

;; The call to MSUSERVBTABLES checks a user hash table to allow extensions.

[COND  
( (LISTP VERB)  
(SETQ MOD (CADR VERB))  
(SETQ VERB (CAR VERB])  
(MKLIST (OR (SELECTQ VERB  
(BIND (SELECTQ MOD  
(NIL ' (**BIND** REF SET SMASH TEST))  
(NOTUSE 'BIND)  
NIL))  
(CALL (SELECTQ MOD  
(DIRECTLY ' (CALL EFFECT PREDICATE NLAMBDA))  
(EFFECT 'EFFECT)  
(INDIRECTLY 'APPLY)  
(NIL ' (APPLY CALL EFFECT ERROR PREDICATE NLAMBDA))  
(NOTERROR ' (APPLY CALL EFFECT PREDICATE NLAMBDA))  
(PREDICATE 'PREDICATE)  
(TESTING 'PREDICATE)  
(VALUE ' (CALL NLAMBDA))  
(NLAMBDA 'NLAMBDA)  
NIL))  
(CREATE (SELECTQ MOD  
(NIL 'CREATE)  
NIL))  
(DECLARE (SELECTQ MOD  
(CL:LOCALLY 'LOCALVARS)  
(LOCALVARS 'LOCALVARS)  
(NIL ' (LOCALVARS SPECVARS))  
(SPECVARS 'SPECVARS)  
NIL))  
(FETCH (SELECTQ MOD  
(NIL 'FETCH)  
NIL))  
(IS (SELECTQ MOD  
(FIELDS ' ((**FETCH**)  
(**REPLACE**)))  
(FNS ' (CALL NOBIND REF (CALL)

```

      (APPLY))
(KNOWN ' (CALL NOBIND REF))
(NIL ' (CALL NOBIND REF (CALL
      (BIND)
      (REFFREE)
      (REF)
      (SETFREE)
      (SET)
      (SMASHFREE)
      (SMASH)
      (RECORDS)
      (FETCH)
      (REPLACE)
      (PROP)
      (APPLY)
      (TEST)
      (TESTFREE)))
(PROPS ' ((PROP))
(RECORDS ' ((RECORD)
      (CREATE)))
(VARS ' ((BIND)
      (REFFREE)
      (REF)
      (SETFREE)
      (SET)
      (SMASHFREE)
      (SMASH)
      (TEST)
      (TESTFREE)))
(TYPE ' ((0))
NIL)
(KNOWN (SELECTQ MOD
      (NIL ' (CALL NOBIND REF))
      NIL))
(PROG (SELECTQ MOD
      (NIL ' PROG)
      NIL))
(REFERENCE (SELECTQ MOD
      (FIELDS ' FETCH)
      (FREELY ' (REFFREE TESTFREE SMASHFREE))
      (CL:LOCALLY ' (REF TEST SMASH))
      (NIL ' (REF REFFREE TEST TESTFREE SMASH SMASHFREE))
      NIL))
(REPLACE (SELECTQ MOD
      (NIL ' REPLACE)
      NIL))
(SET (SELECTQ MOD
      (FIELDS ' REPLACE)
      (FREELY ' SETFREE)
      (CL:LOCALLY ' SET)
      (NIL ' (SET SETFREE))
      NIL))
(SMASH (SELECTQ MOD
      (FIELDS ' REPLACE)
      (FREELY ' SMASHFREE)
      (CL:LOCALLY ' SMASH)
      (NIL ' (SMASH SMASHFREE))
      NIL))
(TEST (SELECTQ MOD
      (FREELY ' TESTFREE)
      (CL:LOCALLY ' TEST)
      (NIL ' (TEST TESTFREE))
      NIL))
(USE (SELECTQ MOD
      (FIELDS ' (FETCH REPLACE))
      (FREELY ' (REFFREE SETFREE SMASHFREE TESTFREE))
      (I.S.OPRS ' CLISP)
      (INDIRECTLY ' LOCALFREEVARS)
      (CL:LOCALLY ' (REF SET SMASH TEST))
      (NIL ' (REF REFFREE SET SETFREE SMASH SMASHFREE TEST TESTFREE))
      (PREDICATE ' (TEST TESTFREE))
      (PROPNames ' PROP)
      (RECORDS ' (CREATE RECORD))
      (TESTING ' (TEST TESTFREE))
      (VALUE ' (REF REFFREE SMASH SMASHFREE))
      (TYPE ' 0)
      NIL))
NIL)
(MSUSERVBTABLES VERB MOD))

```

**(MSUSERVBTABLES**

[LAMBDA (VERB MOD)

(\* sml "20-Dec-85 17:03")

(\* \* Find the relation tables for a user-defined relation)

(OR [AND (BOUNDP 'MSUSERVBTABLES)

```
(HASHARRAYP MSUSERVBTABLES)
(CDR (for modifier in (GETHASH VERB MSUSERVBTABLES) thereis (EQMEMB MOD (CAR modifier]
VERB]))
```

)

(RPAQ? MSUSERVBTABLES (MAKETABLE 2))

(RPAQ? MSANALYZEFNS (MAKETABLE 2))

(DEFINEQ

**(BUILDGETRELQ**

; Edited 16-Jun-87 12:36 by jrb:

```
[LAMBDA (X)
  (PROG ([VAR (COND
    ((LITATOM (CADR X))
     (CADR X))
    (T '$$1]
    FORM F1)
    [for REL in (MSVBTABLES (CAR X)) do [SETQ F1 (LIST 'GETTABLE VAR (LIST (COND
      ((CADDR X)
       'CDDR)
      (T 'CADR))
      (LIST 'FASSOC (KWOTE REL)
       'MSDATABASELST]
      (SETQ FORM (COND
        (FORM (LIST 'UNION F1 FORM))
        (T F1]
    (RETURN (COND
      ((EQ VAR (CADR X))
       FORM)
      (T (LIST (LIST 'LAMBDA (LIST VAR)
        FORM)
        (CADR X]))
```

**(BUILDTESTRELQ**

; Edited 16-Jun-87 12:41 by jrb:

```
[LAMBDA (X)
  (PROG ([VAR (COND
    ((LITATOM (CADR X))
     (CADR X))
    (T '$$1]
    FORM)
    [SETQ FORM (CONS 'OR (for R in (MSVBTABLES (CAR X)) collect (LIST 'TESTTABLE VAR
      (LIST (COND
        ((CADDR X)
         'CDDR)
        (T 'CADR))
        (LIST 'FASSOC (KWOTE R)
         'MSDATABASELST]
    (RETURN (COND
      ((EQ VAR (CADR X))
       FORM)
      (T (LIST (LIST 'LAMBDA (LIST VAR)
        FORM)
        (CADR X]))
```

)

(DECLARE%: DONTCOPY

(DECLARE%: EVAL@COMPILE

(PUTPROPS GETRELQ MACRO (X (BUILDGETRELQ X)))

(PUTPROPS TESTRELQ MACRO (X (BUILDTESTRELQ X)))

)

)

:: erase function

(DEFINEQ

**(MSERASE**

(\* lmm " 1-JUN-81 22:56")

```
[LAMBDA (ATOMS)
  (DECLARE (SPECVARS ERASESET))
  (COND
    ((EQ ATOMS T)
     (SETQ MSCHANGEDARRAY)
     (SETQ MSDATABASELST)
     (SETQ MSFILETABLE)
     (SETQ MSARGTABLE)
     (SETQ MSDBEMPTY T))
    (T (for AT in ATOMS do (MSSTOREDATA AT]))
```

)



:: dump data base

(DEFINEQ

**(DUMPDATABASE**

(\* lmm "12-APR-81 15:57")

```
[LAMBDA (FNLST)
  (PROG (DUMPEDFLG)
    (DECLARE (SPECVARS DUMPEDFLG DUMPTABLE))
    (COND
      (FNLST (MAPC FNLST (FUNCTION UPDATEFN)))
      (T (UPDATECHANGED)))
    (PRINT ' (READATABASE))
    (PRIN1 ' %())
    (TERPRI)
    [for DUMPTABLE in MSDATABASELST when (NOT (MEMB (CAR DUMPTABLE)
      NODUMPRELATIONS))
      do (SETQ DUMPEDFLG NIL)
        [COND
          ((OR (NOT FNLST)
            (EQ (CDDR DUMPTABLE)
              T))
            (MAPTABLE (CADR DUMPTABLE)
              (FUNCTION DUMPDATABASE1)))
          (T (MAPC FNLST (FUNCTION (LAMBDA (FN)
            (DUMPDATABASE1 (GETTABLE FN (CADR DUMPTABLE))
              FN])
            (COND
              (DUMPEDFLG (PRINT]
            (TERPRI)
            (PRIN1 ' %))
            (TERPRI])
```

(\* either dumping everything, or this is a permanent table which should be dumped in entirety (e.g. templates))

**(DUMPDATABASE1**

(\* rmk%: "24-OCT-79 10:02")

```
[LAMBDA (VALUE FN)
  (COND
    (FN (COND
      ((NOT DUMPEDFLG)
        (SETQ DUMPEDFLG (PRIN2 (CAR DUMPTABLE)))
        (SPACES 1)))
      (PRIN2 FN)
      (SPACES 1)
      (PRIN2 VALUE)
      (SPACES 1]))
```

**(READATABASE**

; Edited 3-Jun-88 12:34 by jrb:

```
[LAMBDA NIL
  [SELECTQ (RATOM)
    ((%[ %()])
    (HELP ' (BAD DATABASE]
  (MSINIT)
  (SETQ MSDBEMPTY)
  (PROG (TAB FN NEWREL NAME)
    (while (SETQ NAME (READ)) do (SELECTQ NAME
      (USERTEMPLATES
        (while (SETQ FN (READ)) do (TEMPLATE FN (READ)
          T)))
      (COND
        ((SETQ TAB (FASOC NAME MSDATABASELST))
          (while (SETQ FN (READ)) do (PUTHASH FN T MSCHANGEDARRAY)
            (SETQ NEWREL (MKLIST (READ)))
            (STORETABLE FN TAB NEWREL)
            (PUTHASH FN NIL MSCHANGEDARRAY)))
        (T (LISPXPRT "*** incompatible MASTERSCOPE data base" T T)
          (while (READ])
```

(RPAQQ DATABASECOMS ((E (DUMPDATABASE))))

(ADDTOVAR GAINSPACEFORMS (MSDATABASELST "erase current Masterscope database" (%. ERASE)))

(RPAQQ MSCHECKBLOCKSCOMS

; blocks checker

```
(FNS MSCHECKBLOCKS MSCHECKBLOCK MSCHECKFNINBLOCK MSCHECKBLOCKBASIC MSCHECKBOUNDFREE GLOBALVARP
  PRINTERROR MSCHECKVARS1 UNECCSPEC NECCSPEC SPECVARP SHORTLST DOERROR MMSGPRINT)
(BLOCKS (MSCHECKBLOCKS MSCHECKBLOCKS MSCHECKBLOCK MSCHECKFNINBLOCK MSCHECKBLOCKBASIC MSCHECKBOUNDFREE
  PRINTERROR MSCHECKVARS1 UNECCSPEC NECCSPEC SPECVARP SHORTLST DOERROR MMSGPRINT
  (LOCALFREEVARS SEEN BLKFNS V ERRORS SFLG LF BLKAPPLYCALLERS U LF1 SHOULDBSPECVARS)
  (NOLINKFNS . T)
  (SPECVARS SPECVARS LOCALVARS RETFNS BLKAPPLYFNS BLKLIBRARY NOLINKFNS LINKFNS
  LOCALFREEVARS DONTCOMPILEFNS ENTRIES)
```

(GLOBALVARS SYSLOCALVARS SYSSPECVARS FILELST MSCRATCHASH)
(GLOBALVARP)))

:: blocks checker

(DEFINEQ

(MSCHECKBLOCKS

; Edited 2-Jun-88 13:45 by jrb:

[LAMBDA (FILES)
(PROG ((LOCALVARS SYSLOCALVARS)
(SPECVARS T)
FNS BLOCKS NOBLOCKFNS TEM)
[COND
((NULL FILES)
(SETQ FILES FILELST))
((NLISTP FILES)
(SETQ FILES (OR (GETP FILES 'FILEGROUP)
(LIST FILES))
[for FL in FILES do (MSNOTICEFILE FL)
(SETQ BLOCKS (NCONC (FILECOMSLST FL 'BLOCKS)
BLOCKS))
;; JRB - Check now gathers everything analyzable, not just FNS
(MAPC [SETQ TEM (for FT in MSFNTYPES join (FILECOMSLST FL (ffetch (MSANALYZABLE
FILEPKGNAME)
of FT]
(FUNCTION UPDATEFN))
(SETQ FNS (NCONC TEM FNS))
(COND
((SETQ TEM (FILECOMSLST FL 'LOCALVARS))
(APPLY (FUNCTION LOCALVARS)
TEM)))
(COND
((SETQ TEM (FILECOMSLST FL 'SPECVARS))
(APPLY (FUNCTION SPECVARS)
TEM)))
(COND
((SETQ TEM (FILECOMSLST FL 'GLOBALVARS))
;; Ordinarily a noop, since noticing the file sets up GLOBALVARS (unlike SPECVARS and LOCALVARS,
;; which are EVAL@COMPILE); however, user might have edited coms since then
(APPLY (FUNCTION ADDTOVAR)
(CONS 'GLOBALVARS TEM]
(UPDATECHANGED)
(TAB 0 0)
(COND
((SETQ NOBLOCKFNS (for FN in FNS unless [OR (MEMB FN DONTCOMPILEFNS)
(for BLOCK in BLOCKS thereis (MEMB FN (CDR BLOCK]
collect FN))
(MSCHECKBLOCK (CONS (COND
[BLOCKS (APPEND ('no block - ")
(COND
[CDDDDR NOBLOCKFNS)
(APPEND (LDIFF NOBLOCKFNS (CDDDDR NOBLOCKFNS))
'("--")
(T NOBLOCKFNS]
(T (CONS "File" FILES)))
NOBLOCKFNS)
FNS BLOCKS)))
(for BLOCK in BLOCKS do (MSCHECKBLOCK BLOCK FNS BLOCKS])

(MSCHECKBLOCK

; Edited 3-Jun-88 10:50 by jrb:

[LAMBDA (BLOCK FNS BLOCKS)
(LET ((SPECVARS SPECVARS)
(LOCALVARS LOCALVARS)
(BLKNAME (CAR BLOCK))
BLKAPPLYFNS ENTRIES LOCALFREEVARS BLKFNS FREEVARS TEM TEM2 BOUNDFREE BLKAPPLYCALLERS ERRORS
SHOULDBESPECVARS (GLOBALVARS GLOBALVARS)
(REFNS RETFNS)
(BLKLIBRARY BLKLIBRARY)
(NOLINKFNS NOLINKFNS)
(LINKFNS LINKFNS)
(DONTCOMPILEFNS DONTCOMPILEFNS))
(DECLARE (SPECVARS GLOBALVARS RETFNS BLKLIBRARY NOLINKFNS LINKFNS DONTCOMPILEFNS))
(COND
((LISTP BLKNAME)
(SETQ BLKNAME NIL)))
[COND
(BLKNAME (SETQ LOCALVARS T)
(SETQ SPECVARS (COND
((NEQ SPECVARS T)
(UNION SYSSPECVARS SPECVARS))
(T SYSSPECVARS)
[for X in (CDR BLOCK) do ; set up declarations and BLKFNS
(COND
((LISTP X)

```

[SETQ TEM (COND
  ((EQ (CADR X)
    '*))
  (EVAL (CADDR X)))
  (T (CDR X]
(SET (CAR X)
  (COND
    ((NLISTP (CDR X))
      (CDR X))
    ([LISTP (SETQ TEM2 (EVALV (CAR X]
      (APPEND TEM TEM2))
      (T TEM)))
  ;; ASSERT: ((REMOTE EVAL) SPECVARS LOCALVARS LOCALFREEVARS GLOBALVARS
  ;; BLKLIBRARY SYSSPECVARS BLKAPPLYFNS ENTRIES LINKFNS NOLINKFNS RETFNS
  ;; SYSLOCALVARS)
  (SELECTQ (CAR X)
    (SPECVARS (COND
      ((EQ TEM T)
        (SETQ LOCALVARS SYSLOCALVARS))))
    (LOCALVARS (COND
      ((EQ TEM T)
        (SETQ SPECVARS SYSSPECVARS))))
    ((LOCALFREEVARS GLOBALVARS BLKLIBRARY SYSSPECVARS BLKAPPLYFNS ENTRIES
      LINKFNS NOLINKFNS RETFNS SYSLOCALVARS))
    (DOERROR (CAR X)
      "unrecognized item in block declaration")))
  ((MEMB X BLKFNS)
    (DOERROR X "on block twice"))
  (T (SETQ BLKFNS (CONS X BLKFNS]
(COND
  (BLKNAME (MSCHECKBLOCKBASIC BLOCK BLKNAME))
  (T (COND
    (BLKAPPLYFNS (DOERROR BLKAPPLYFNS "BLKAPPLYFNS but not a real block" NIL T)))
    (SETQ BLKLIBRARY NIL)))
  (for FN in BLKFNS do (OR (FMEMB FN FNS)
    (FMEMB FN BLKLIBRARY)
    (DOERROR FN "not on the file")))
  (COND
    (BLKNAME ; a real block
      (MSCHECKFNINBLOCK FN BLOCK BLOCKS)))
  [for VAR in (UNION (SETQ TEM (GETRELQ (BIND NOTUSE)
    FN))
    (GETRELQ (USE CL:LOCALLY)
      FN))
    do [OR (FMEMB VAR BOUNDFREE)
      (FMEMB VAR SYSSPECVARS)
      (GLOBALVARP VAR)
      (COND
        ((TESTRELQ (USE FREELY)
          VAR T) ; i.e. it is bound in this block, and used freely by someone else
          (SETQ BOUNDFREE (CONS VAR BOUNDFREE)))
        ((SPECVARP VAR FN)
          (AND (NEQ SPECVARS T)
            (UNECCSPEC FN VAR)))
        ((FMEMB VAR TEM)
          (DOERROR FN "binds and never uses" VAR T]
      (COND
        ((AND (FMEMB VAR (GETRELQ (USE INDIRECTLY)
          FN))
          (NOT (SPECVARP VAR FN)))
          (DOERROR VAR "should be SPECVAR (used in functional arg) in" FN T]
      (SETQ FREEVARS (UNION (GETRELQ (USE FREELY)
        FN)
        FREEVARS)))
  (MSCHECKBOUNDFREE BOUNDFREE BLKNAME)
  [for VAR in FREEVARS unless (OR (FMEMB VAR SHOULDBSPECVARS)
    (FMEMB VAR SYSSPECVARS)
    (FMEMB VAR (LISTP SPECVARS))
    (FMEMB VAR LOCALFREEVARS)
    (FMEMB VAR GLOBALVARS)
    (GETPROP VAR 'GLOBALVAR)
    (CL:CONSTANTP VAR)
    (GET VAR 'GLOBALLY-SPECIAL))
    do (COND
      ((NULL (SETQ TEM (for FN in (GETRELQ (USE FREELY)
        VAR T)
        when (FMEMB FN BLKFNS) collect FN)))
        ; Nobody uses it??
      )
      ((TESTRELQ BIND VAR T)
        (DOERROR VAR "not declared, used freely by " TEM 0))
      ((NOT (BOUNDP VAR))
        (DOERROR VAR "not declared, never bound, no top-level value, used freely by" TEM T))
      (T (DOERROR VAR "not bound, not a GLOBALVAR, used freely by" TEM T]
  (for DEC in BLOCK when (LISTP DEC)

```



```
[COND
  ((AND (EQ BLKNAME (CAR ENTRIES))
        (NULL (CDR ENTRIES))
        (NULL BLKAPPLYFNS))
    (* MKENTRIES treats the case of ENTRIES=NIL specially by
    not setting up a separate BLOCK.)

  (SETQ ENTRIES NIL))
  ((AND (NULL ENTRIES)
        BLKAPPLYFNS)
    (* Above caper only works if no BLKAPPLYFNS)
  (SETQ ENTRIES (LIST BLKNAME])
(COND
  ((MEMB BLKNAME ENTRIES)
   (DOERROR BLKNAME "can't be both entry and block name")))
  (for X in [APPEND BLKAPPLYFNS (OR ENTRIES (SETQ ENTRIES (LIST BLKNAME] do (OR (MEMB X BLKFNS)
   (DOERROR X "on ENTRIES or
   BLKAPPLYFNS but not in
   block"))))

  (for FN in BLKLIBRARY when (AND [NOT (FMEMB FN ' (EQUAL GETPROP GETP NTH TAILP MEMBER]
   (for Y in (GETRELQ (CALL NOTERROR)
   FN T)
   thereis (FMEMB Y BLKFNS)))

  do (COND
    ((NULL (GETPROP FN 'BLKLIBRARYDEF))
     (DOERROR FN "on BLKLIBRARY but no BLKLIBRARYDEF property" NIL T)))
    (SETQ BLKFNS (NCONC1 BLKFNS FN)))
(COND
  ([AND BLKAPPLYFNS (NOT (SETQ BLKAPPLYCALLERS (for X in ' (BLKAPPLY BLKAPPLY*)
   join (for Y in (GETRELQ (CALL NOTERROR)
   X T)
   when (FMEMB Y BLKFNS) collect Y]
   (DOERROR BLKAPPLYFNS "BLKAPPLYFNS but no calls to BLKAPPLY in block" NIL T])
```

**(MSCHECKBOUNDFREE**

```
[LAMBDA (BOUNDFREE BLKNAME)
  (for V in BOUNDFREE do (SCRATCHASH SEEN (PROG ((USERS (GETRELQ (USE FREELY)
   V T))
   (LF (FMEMB V LOCALFREEVARS))
   (BINDERS (GETRELQ BIND V T))
   LF1 SFLG)
  (CLRHASH SEEN)
  (for X in USERS do (PUTHASH X -1 SEEN))
  (for X in BINDERS do (PUTHASH X 1 SEEN))
  (for U in USERS do (COND
    ((FMEMB U BLKFNS)
     (COND
      ((FMEMB U BINDERS)
       (NECCSPEC V U U)))
     (SETQ LF1 LF)))
   (MSCHECKVARS1 U))
(COND
  ((AND (NULL SFLG)
        (OR BLKNAME (EQ LOCALVARS T))
        (NEQ SPECVARS T))
   (for X in BINDERS when (FMEMB X BLKFNS)
    do (SELECTQ (GETHASH X SEEN)
      (2)
      (-1)
      (AND (SPECVARP V X)
           (UNECCSPEC X V]))
```

**(GLOBALVARP**

```
[LAMBDA (X)
  (OR (FMEMB X GLOBALVARS)
      (GETPROP X 'GLOBALVAR])
  (* Imm "31-DEC-78 15:23")
```

**(PRINTERERROR**

```
[LAMBDA (ERR)
  (PROG ((MSG (CAR ERR))
        (VALS (CDDR ERR))
        (NEWPRS PR POS POS2 (LL (IDIFFERENCE (LINELENGTH)
        30))
        POS3)
        (SELECTQ (CAR MSG)
          (0 (SETQ MSG (CDR MSG))
            (PRIN1 "
            (note) "))
          (T (SETQ MSG (CDR MSG))
            (PRIN1 "
            (possible error) "))
          (PRIN1 "
            (probable error) "))
        (COND
          (VALS (for X inside VALS do (PRIN2 X)
                (SPACES 1))
            (PRIN1 '-)
```

```

      (for X inside MSG do (SPACES 1)
        (PRIN1 ' %.)
        (TERPRI)))
[for PRL on (DREVERSE (CADR ERR)) do [COND
  ([NULL (CDDR (SETQ PR (CAR PRL))
    (for ERR in (CDR PRL)
      do (COND
        ((EQUAL (CDR ERR)
          (CDR PR))
          (FRPLACA ERR (CONS (CAR PR)
            (CAR ERR)))
          (FRPLACA PR NIL)
          (RETURN])
        (AND (CAR PR)
          (SETQ NEWPRS (CONS PR NEWPRS])
      (COND
        (NEWPRS (TAB 0 0)
          (SHORTLST (CAAR NEWPRS)
            4)
          (SETQ POS (POSITION))
          (PRIN1 " - ")
          (for X inside MSG do (PRIN1 X)
            (SPACES 1))
          (SETQ POS2 (POSITION))
          [COND
            ((OR (ILESSP POS2 (IDIFFERENCE POS 3))
              (IGREATERP POS2 LL))
              (SETQ POS2 (IPLUS POS 10))
              (SETQ POS3 (IDIFFERENCE (IQUOTIENT (IPLUS POS POS2)
                2)
                4)
              (PRIN1 " -")
              (SHORTLST (CDAR NEWPRS)
                4)
              (PRIN1 ' ".
                ")
              (MAPC (CDR NEWPRS)
                (FUNCTION (LAMBDA (PR)
                  (SHORTLST (CAR PR)
                    4)
                  (TAB POS T)
                  (PRIN1 " -")
                  (TAB POS3 T)
                  (PRIN1 " %"%" " ")
                  (TAB POS2 T)
                  (PRIN1 "-")
                  (SHORTLST (CDR PR)
                    4)
                  (PRIN1 ".
                    "])
                ))
              (SHORTLST (CDR NEWPRS)
                4)
              (PRIN1 ".
                "])
          ]
        ]
      ]
    ]
  ]

```

(MSCHECKVARS1

(\* Imm "16-Jul-84 14:54")

```

[LAMBDA (FN)
  [COND
    ((AND LF1 (FMEMB FN ENTRIES))
      (DOERROR V [CONS "on LOCALFREEVARS" (COND
        ((EQ U FN)
          "but used freely by the entry")
        (T (LIST "but the entry" FN "can reach functions using it
          freely"]
          U T)
        (SETQ LF (SETQ LF1 NIL])
        (PROG ((CALLERS (GETRELQ (CALL NOTERROR)
          FN T))
          (VAL 3))
          [COND
            ((FMEMB FN BLKAPPLYFNS)
              (SETQ CALLERS (UNION BLKAPPLYCALLERS CALLERS])
          ]
          (* interpretation of SEEN codes -
          0 recursive call -
          -1 uses var -
          1 binds var -
          2 binds var, path from it to user -
          T always bound above -
          3 no callers -
          4 not always bound above)
          (for X in CALLERS do (SELECTQ (GETHASH X SEEN)
            ((0 -1 4)
              (3 (SETQ VAL 4)
                (1

```

(\* we have found a path from a user up to a binder -  
 if the path is entirely in the block, then LOCALFREEVAR is ok, -  
 if the path is outside the block, then it doesn't matter, otherwise SPECVAR)

```

(COND
  ((FMEMB X BLKFNS) (* if the binder isn't in this block, ignore)
   (* should just be SPECVAR if not entirely within the block)
   (NECCSPEC v x u))
  (PUTHASH X 2 SEEN)
  (COND
    ((EQ VAL 3)
     (SETQQ VAL T))))
((T 2)
  (COND
    ((EQ VAL 3)
     (SETQQ VAL T))))
(NIL (* now check recursively)
  (PUTHASH X 0 SEEN)
  (PUTHASH X (MSCHECKVARS1 X)
   SEEN))
(SHOULDNT 4))

```

(RETURN VAL)]

**(UNECCSPEC**

```

[LAMBDA (FN VAR) (* lmm "30-AUG-78 03:36")
  (OR (GLOBALVARP VAR)
    (FMEMB VAR (GETRELO (USE INDIRECTLY)
      FN))
    (DOERROR VAR "might not need to be a specvar in" FN T])

```

**(NECCSPEC**

```

[LAMBDA (VAR BINDER) (* lmm "21-SEP-78 04:21")
  (COND
    ((NOT (OR SFLG (SPECVARP VAR BINDER)))
     (SETQ SFLG T)
     (SETQ SHOULDDBESPECVARS (CONS VAR SHOULDDBESPECVARS))
     (DOERROR VAR (LIST "(used freely in)" U "is not a SPECVAR in"
      BINDER T])

```

**(SPECVARP**

```

[LAMBDA (X FN) (* lmm "25-JUN-78 01:15")
  (COND
    ((FMEMB X (GETRELO (DECLARE LOCALVARS)
      FN))
     NIL)
    ((FMEMB X (GETRELO (DECLARE SPECVARS)
      FN))
     T)
    ((NEQ LOCALVARS T)
     (NOT (FMEMB X LOCALVARS)))
    (T (OR (EQ SPECVARS T)
      (FMEMB X SPECVARS)
      (FMEMB X LOCALFREEVARS)
      (FMEMB X GLOBALVARS)
      (GETP X 'GLOBALVAR])

```

**(SHORTLST**

```

[LAMBDA (X N) (* lmm "9-AUG-77 03:18")
  (COND
    ((NULL X)
     N)
    ((LISTP X)
     (SHORTLST (CDR X)
      (SHORTLST (CAR X)
       N)))
    (T (COND
      ((IGREATERP (SETQ N (SUB1 N))
        0)
       (SPACES 1)
       (PRIN2 X))
      ((ZEROP N)
       (PRIN1 " etc"))
      N])

```

**(DOERROR**

```

[LAMBDA (AT MSG ARG QUESTIONABLE) (* lmm "21-Mar-85 08:29")
  [COND
    (QUESTIONABLE (SETQ MSG (CONS QUESTIONABLE MSG))
    (PROG ([L (CDR (OR (SASSOC MSG ERRORS)
      (CAR (SETQ ERRORS (CONS (CONS MSG (CONS))
        ERRORS))
      (AT AT))
    (COND
      (ARG [SETQ AT (OR (FASSOC AT (CAR L))
        (CAAR (FRPLACA L (CONS (CONS AT)

```

```

(CAR L]
(OR (MEMBER ARG (CDR AT))
(NCONC1 AT ARG)))
((NOT (FMEMB AT (CDR L)))
(FRPLACD L (CONS AT (CDR L]))

```

**(MSMSGPRINT**

[LAMBDA (MSG)

; Edited 3-Jun-88 12:39 by jrb:

;; Prints messages Masterscope builds as lists - only atoms get prin2'ed.

```

(COND
((STRINGP MSG)
(PRIN1 MSG))
((CL:CONSP MSG)
(PRIN1 "(")
(MSMSGPRINT (pop MSG))
(while (CL:CONSP MSG) do (PRIN1 " ")
(MSMSGPRINT (pop MSG))))
(if MSG
then (PRIN1 " ." )
(MSMSGPRINT MSG))
(PRIN1 ")")
(T (PRIN2 MSG))
)

```

(DECLARE%: DONTVAL@LOAD DOEVAL@COMPILE DONTCOPY

```

(BLOCK%: MSCHECKBLOCKS MSCHECKBLOCKS MSCHECKBLOCK MSCHECKFNINBLOCK MSCHECKBLOCKBASIC MSCHECKBOUNDFREE PRINTERROR
MSCHECKVARS1 UNECCSPEC NECCSPEC SPECVARP SHORTLST DOERROR MSMSGPRINT
(LOCALFREEVARS SEEN BLKFNS V ERRORS SFLG LF BLKAPPLYCALLERS U LF1 SHOULDBSPECVARS)
(NOLINKFNS . T)
(SPECVARS SPECVARS LOCALVARS RETFNS BLKAPPLYFNS BLKLIBRARY NOLINKFNS LINKFNS LOCALFREEVARS DONTCOMPILEFNS
ENTRIES)
(GLOBALVARS SYSLOCALVARS SYSSPECVARS FILELST MSCRATCHASH)
GLOBALVARP)
)

```

**(RPAQQ MSPATHSCOMS**

```

[
; PATHS
(FNS MSPATHS MSPATHS1 MSPATHS2 MSONPATH MSPATHS4 DASHES DOTABS BELOWMARKER MSPATHSPRINTFN)
(BLOCKS (MSPATHSBLOCK (ENTRIES MSPATHS MSONPATH MSPATHS2)
MSPATHS MSPATHS1 MSPATHS2 MSONPATH MSPATHS4 DASHES DOTABS BELOWMARKER MSPATHSPRINTFN
(LOCALFREEVARS TABS NAMED LINENUM LL BELOWCNT MARKING SEEN INVERTED TO NOTRACE AVOIDING
SEPARATE)
(GLOBALVARS MSBLIP MSCRATCHASH)
(NOLINKFNS . T])
)

```

;; PATHS

(DEFINEQ

**(MSPATHS**

[LAMBDA (FROM TO INVERTED AVOIDING SEPARATE NOTRACE MARKING) ; Edited 3-Jun-88 12:37 by jrb:

;; Display paths; must print all of FROM, with separate tree for all of SEPARATE (considered as a subset of FROM). Stop when you get to a  
;; function in NOTRACE, or if DEPTH is exceeded -- unless TO is NIL, only print paths that eventually reach an element of TO. If INVERTED is not  
;; NIL, print inverted tree. Do not print out functions in AVOIDING

```

(SCRATCHASH SEEN (PROG (TABS (LL (LINELENGTH))
(BELOWCNT 0)
(LINENUM 0)
(FIRST T)
X NAMED TEM (UNDONE (MSLISTSET FROM T)))
(COND
(INVERTED (PRINTOUT T "inverted tree" T))
[MAPC UNDONE (FUNCTION (LAMBDA (X)
(PUTHASH X (COND
((AND NOTRACE (MSMEMBSET X NOTRACE))
-1)
(T 0))
SEEN])
(TAB 0 0)
[RESETVARS ((MSPRINTFLG))
(do (COND
(NAMED (OR FIRST (DASHES (GETHASH (CAR NAMED)
SEEN)))
(SETQ FIRST)
(PUTHASH (CAR NAMED)
0 SEEN)
(MSPATHS1 (CAR NAMED)
NIL T)
(SETQ NAMED (CDR NAMED)))
(UNDONE [COND
([OR (NULL (SETQ TEM (GETHASH (CAR UNDONE)
SEEN)))
(EQ TEM 0)

```



```

      (AND (LISTP TEM)
            (NULL (CAR TEM)]
      (PUTHASH (CAR UNDONE)
               (LIST NIL)
               SEEN)
      (SETQ NAMED (LIST (CAR UNDONE]
                        (SETQ UNDONE (CDR UNDONE)))
      (T (TERPRI)
         (RETURN]
      (RETURN])

```

**(MSPATHS1**

```

[LAMBDA (FROM FIRST LAST) (* Imm " 4-AUG-83 23:45")
  (PROG (TEM THISLINE POS (XT TABS))
    [COND
      ((NOT FIRST)
       (TERPRI)
       (SETQ LINENUM (ADD1 LINENUM)) (* if NOT (EQMEMB (QUOTE NOLINE) PRINTOPTIONS) then)
       (PRIN1 LINENUM)
       (PRIN1 ".")
       (DOTABS (CDR TABS]
      (SETQ THISLINE LINENUM)
      (AND TABS (TAB (CAR TABS)
                    0))
      (AND LAST (SETQ TABS (CDR TABS)))
      (SETQ POS (MSPATHSPRINTFN FROM))
      (MSPATHS2 FROM)
      (COND
        [(NEQ (SETQ TEM (GETHASH FROM SEEN))
              0) (* Already expanded on a previous line -
                 or is a NOTRACE)

          (COND
            ((EQ TEM MSBLIP)
             (SHOULDNT 5))
            ((OR (NOT (NUMBERP TEM))
                 (NOT (MINUSP TEM)))
             (PRIN1 " {")
             (PRIN1 (COND
                     ((NLISTP TEM) (* Either line number or overflow line letter)
                      TEM)
                     [(LISTP TEM) (* A list means that this must be a sub-tree)
                      (COND
                        ((CAR TEM)
                         (T (FRPLACA TEM (BELOWMARKER))
                            (SETQ NAMED (NCONC1 NAMED FROM))
                            (CAR TEM]
                        (T TEM)))
                     (PRIN1 "}")])
            (T (PROG ((TABS TABS)
                    (FIRST T)
                    NEXTLEVEL TEM)
              (PUTHASH FROM (IDIFFERENCE -1 THISLINE)
                          SEEN)
              (OR (SETQ NEXTLEVEL (for Y in (COND
                                           ((NOT INVERTED)
                                            (GETRELQ CALL FROM))
                                           (T (GETRELQ CALL FROM T)))
                    when (MSPATHS2 Y) collect Y) (* AND (SETQ TEM (FASSOC (QUOTE SORT)
                                          PRINTOPTIONS)) (SORT NEXTLEVEL
                                          (CDR TEM)))
                  (RETURN))

                (COND
                  ([AND XT (OR (SETQ TEM (AND SEPARATE (MSMEMBSET FROM SEPARATE)))
                              (SOME NEXTLEVEL (FUNCTION (LAMBDA (FN)
                                                            (IGREATERP (IPLUS (NCHARS FN)
                                                                           POS 6)
                                                                           LL])
                              (* NOT (EQMEMB (QUOTE NOLINE) PRINTOPTIONS))

                    (SETQ NAMED (NCONC1 NAMED FROM))
                    (PRIN1 " {")
                    [PRIN1 (COND
                          (TEM (CAR (PUTHASH FROM (LIST (BELOWMARKER))
                                                       SEEN)))
                          (T (PUTHASH FROM (BELOWMARKER)
                                           SEEN]
                    (PRIN1 "}")
                    (RETURN))
                    (SETQ TABS (CONS POS TABS))
                    (PUTHASH FROM THISLINE SEEN)
                    (for X on NEXTLEVEL do (MSPATHS1 (CAR X)
                                                      FIRST
                                                      (NULL (CDR X)))
                    (SETQ FIRST])

```

**(MSPATHS2**

[LAMBDA (FN FLG) (\* Imm "20-Jul-84 14:36")

(\* Returns T if FN should be PRINTED -
The SEEN table contains one of the following entries for a function -
MSBLIP %: don't print the function at all -
n a number %: don't trace it, it was expanded previously -
-n %: don't trace it, it was printed earlier, though it had no sub-functions -
0 %: yes, print and trace it -
-1 %: yes, print it, but don't trace it -
(NIL)%: it should be given a separate tree, as yet unnamed -
(letter)%: give it a separate tree with this letter name -
letter %: the function is expanded in an OVERFLOW table below)

(\* When below MSPATHS4 for ON PATH sets (and CALL SOMEHOW) the SEEN table contains either 0 %: not traced yet,
MSBLIP %: don't print, -1 print, don't trace, T %: top set (e.g. for CALLED SOMEHOW BY X, X is originally marked T) 1
already seen and traced)

(NEQ [OR (GETHASH FN SEEN)
(PROGN (OR INVERTED (UPDATEFN FN NIL 0))
(COND
((AND AVOIDING (MSMEMBSET FN AVOIDING)) (\* If it is avoiding, then no)
(PUTHASH FN MSBLIP SEEN))
((AND (NULL FLG)
NOTRACE
(MSMEMBSET FN NOTRACE))

(\* Will not be traced%: entry should be either MSBLIP or -1 depending on whether the function should be printed)

(COND
((MSPATHS2 FN T)
(PUTHASH FN -1 SEEN))
(T MSBLIP)))
(NULL TO)
(PUTHASH FN (COND
((AND (NULL INVERTED)
(GETD FN)
(NOT (TESTRELQ KNOWN FN)))
MSBLIP)
(T 0))
SEEN))
((MSMEMBSET FN TO) (\* If it is in the TO set, then definitely YES)
(PUTHASH FN 0 SEEN))
(T (\* Will a path through this function eventually print out an
element of TO?)
(\* assume not)
(PUTHASH FN MSBLIP SEEN)
(COND
((OR (NULL FLG)
(NULL NOTRACE)
(NOT (MSMEMBSET FN NOTRACE))))
(for Y in (COND
((NOT INVERTED)
(GETRELQ CALL FN))
(T (GETRELQ CALL FN T)))
when (MSPATHS2 Y) do (RETURN (PUTHASH FN 0 SEEN)) finally (RETURN MSBLIP)))
(T MSBLIP]

MSBLIP])

(MSONPATH

[LAMBDA (SETREP) ; Edited 9-Jan-87 20:44 by jds

(PROG ((FROM (fetch FROM of (fetch MSPATHOPTIONS of SETREP)))
(TO (fetch TO of (fetch MSPATHOPTIONS of SETREP)))
(AVOIDING (fetch AVOIDING of (fetch MSPATHOPTIONS of SETREP)))
(NOTRACE (fetch NOTRACE of (fetch MSPATHOPTIONS of SETREP)))
INVERTED
(TOPFLG (fetch TOPFLG of (fetch MSPATHOPTIONS of SETREP)))
(SEEN (HASHARRAY 20))
TEM)
(COND
((NULL FROM)
(SETQ INVERTED T)
(SETQ FROM TO)
(SETQ TO NIL)))
(SETQ TEM (MSLISTSET FROM T))
[MAPC TEM (FUNCTION (LAMBDA (X)
(PUTHASH X 0 SEEN)) (\* 0 means yes expand, not expanded yet)
[MAPC TEM (FUNCTION (LAMBDA (X)
(MSPATHS4 X TOPFLG]
(RETURN SEEN])

(MSPATHS4

[LAMBDA (FROM TOP) (\* Imm "25-JUN-78 01:10")

(\* traces paths from FROM. When done, the SEEN array will contain MSBLIP or NIL for entries not expanded, 0 for entries
which should be expanded but weren't for some reason (probably a bug)%, 1 for entries which were below the "top" and T
for entries which were above the top only)

```
(PROG (TEM)
  (COND
    ((MSPATHS2 FROM)
      (COND
        ((EQ (SETQ TEM (GETHASH FROM SEEN))
          0)
          (PUTHASH FROM (COND
            (TOP T)
            (T 1))
            SEEN)
          (for Y in (COND
            (INVERTED (GETRELQ CALL FROM T))
            (T (GETRELQ CALL FROM)))
            do (MSPATHS4 Y))
          ((AND (EQ TEM T)
            (NOT TOP))
            (PUTHASH FROM 1 SEEN]))
```

**(DASHES**

```
[LAMBDA (MARKER) (* Imm "21-JAN-79 14:28")
  (TERPRI)
  (FRPTQ (IDIFFERENCE LL 20)
    (PRIN1 '-))
  (PRIN1 (COND
    ((LISTP MARKER) (* OR (EQMEMB (QUOTE NOLINE) PRINTOPTIONS))
      (PRIN1 "----- ")
      (OR (CAR MARKER)
        '""))
    (T (PRIN1 "--- overflow - "
      MARKER]))
```

**(DOTABS**

```
[LAMBDA (LST) (* Imm%: 19 MAY 75 146)
  (COND
    ((NULL LST)
      NIL)
    (T (DOTABS (CDR LST))
      (TAB (CAR LST)
        0)
      (PRIN1 "|"))
```

**(BELOWMARKER**

```
[LAMBDA NIL (* Imm "22-JUN-78 00:15")
  (* Imm%: 26 MAY 75 1751)
  (PROG1 [COND
    ((ILESSP BELOWCNT 26)
      (FCHARACTER (IPLUS 97 BELOWCNT)))
    (T (PACK* (FCHARACTER (IPLUS 97 (IREMAINDER BELOWCNT 26)))
      (IQUOTIENT BELOWCNT 26)
      (SETQ BELOWCNT (ADD1 BELOWCNT))))]
```

**(MSPATHSPRINTFN**

```
[LAMBDA (FN) (* Imm "16-MAY-78 02:27")
  (AND MARKING (MSMEMBSET FN MARKING)
    (PRIN1 ">"))
  (PRIN2 FN)
  (ADD1 (POSITION])
```

```
)
(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY
(BLOCK%: MSPATHSBLOCK (ENTRIES MSPATHS MSONPATH MSPATHS2)
  MSPATHS MSPATHS1 MSPATHS2 MSONPATH MSPATHS4 DASHES DOTABS BELOWMARKER MSPATHSPRINTFN
  (LOCALFREEVARS TABS NAMED LINENUM LL BELOWCNT MARKING SEEN INVERTED TO NOTRACE AVOIDING SEPARATE)
  (GLOBALVARS MSBLIP MSCRATCHASH)
  (NOLINKFNS . T))
)
(DEFINEQ
```

**(MSFIND**

```
[LAMBDA (IN X) (* Imm "24-JAN-79 15:16")
  (OR (EQ IN X)
    (AND (LISTP IN)
      (OR (MSFIND (CAR IN)
        X)
        (MSFIND (CDR IN)
          X]))
```

**(MSEDITF**

; Edited 31-May-88 17:58 by jrb:

```

[LAMBDA ARGCOUNT
  (LET [(FNAME (ARG ARGCOUNT 1))
        (FEDITCOMS (for X from 2 to ARGCOUNT collect (ARG ARGCOUNT X)
                  (for FPTYPE in MSFNTPES bind FPNAME when (HASDEF FNAME (SETQ FPNAME (ffetch (MSANALYZABLE FILEPKGNAME)
                                                of FPTYPE)))
                    do (if (EQ FPTYPE 'FNS)
                          then (APPLY 'EDITF (CONS FNAME FEDITCOMS))
                          else (EDITE (GETDEF FNAME FPNAME NIL ' (NOERROR NOCOPY EDIT))
                                       FEDITCOMS FNAME FPNAME))
                    (RETURN FNAME]))
        )
  (RPAQQ MSBLIP "sysout and inform Masinter@PARC")
  (RPAQQ MSFNTPES ((FNS FNS GETDEF)))
  ;; SCRATCHASH
  (RPAQ? MSCRATCHASH )
  (DECLARE%: DONTCOPY
  (DECLARE%: EVAL@COMPILE
  (PUTPROPS SCRATCHASH MACRO [(ARRAYNAME . FORMS)
                              ([LAMBDA (ARRAYNAME)
                                (SETQ MSCRATCHASH)
                                (PROG1 (PROGN . FORMS)
                                       (SETQ MSCRATCHASH ARRAYNAME))
                                (COND
                                 (MSCRATCHASH (CLRHASH MSCRATCHASH)
                                               MSCRATCHASH)
                                 (T (HASHARRAY 20 (FUNCTION MSREHASH]))
                                )
                              )
  )
  )

```

;; marking changed

(DEFINEQ

**(MSMARKCHANGED**

; Edited 8-Apr-88 10:17 by jrb:

```

[LAMBDA (NAME TYPE REASON)
  ;; Called from inside MARKASCHANGED
  ;; JRB - Some things need to be MARKASCHANGED strangely (FUNCTIONS that are actually DEFMACROS need to have CHANGEMACRO
  ;; called, but not MSMARKCHANGE1, for instance). If there is a function in the MARKCHANGED-FN slot of the appropriate MSANALYZABLE
  ;; record, and it returns non-NIL, MSMARKCHANGE1 gets called.
  (AND MSDATABASELST (LET (ATYPEREC MSMCFN)
                        (if (OR (NULL TYPE)
                                (SETQ ATYPEREC (ASSOC TYPE MSFNTPES)))
                            then (COND
                                  ((EQ REASON 'DELETED)
                                   (MSERASE (LIST NAME)))
                                  ((OR (NOT (SETQ MSMCFN (ffetch (MSANALYZABLE MARKCHANGED-FN)
                                                                ATYPEREC)))
                                       (APPLY* MSMCFN NAME TYPE REASON))
                                   (MSMARKCHANGE1 NAME 0)))
                                  (SELECTQ TYPE
                                    ((VARS VARIABLES T)
                                     (CHANGEVAR NAME TYPE REASON))
                                    (MACROS (CHANGEMACRO NAME TYPE REASON))
                                    (I.S.OPRS (CHANGEI.S. NAME TYPE REASON))
                                    NIL))
  )

```

**(CHANGEMACRO**

; Edited 3-Jun-88 12:17 by jrb:

```

[LAMBDA (NAME TYPE REASON)
  ;; We don't do anything if the item is being defined. This is a heuristic that compensates for the fact that a database can be loaded from a file
  ;; before the definitions that it knows about come in. We don't want a subsequent LOADFROM of a file to generate all sorts of probably spurious
  ;; messages.
  (AND MSDATABASELST (NEQ REASON 'DEFINED)
        (MSNEEDUNSAVE (GETRELATION NAME ' (CALL DIRECTLY)
                                T)
                      (FILEPKGTYPE TYPE 'DESCRIPTION)
                      (EQ (GETTEMPLATE NAME T)
                          'MACRO))
  )

```

**(CHANGEVAR**

(\* rmk%: "19-FEB-81 15:22")

```

[LAMBDA (NAME TYPE REASON)
  (DECLARE (GLOBALVARS COMPILE.TIME.CONSTANTS))
  (AND MSDATABASELST (FMEMB NAME COMPILE.TIME.CONSTANTS)
        (MSNEEDUNSAVE (GETRELATION NAME ' (USE FREELY)
                                T)
  )

```

"constants"])

**(CHANGEI.S.**

```
[LAMBDA (NAME TYPE REASON) ; Edited 3-Jun-88 12:18 by jrb:
  (AND MSDATABASELST (SELECTQ REASON
    (DEFINED ;; If it has a function definition, then defining it as an i.s.opr has no effect (at least for interpreted code)
      (AND (NOT (GETD NAME))
        (MSNEEDUNSAVE (GETRELATION NAME ' (CALL DIRECTLY)
          T)
          ' (i.s.oprs as functions)
          T)))
    ((CHANGED DELETED)
      (MSNEEDUNSAVE (UNION (GETRELATION NAME ' (USE I.S.OPRS)
        T)
        (AND (U-CASEP NAME)
          (GETRELATION (L-CASE NAME)
            ' (USE I.S.OPRS)
            T)))
        "i.s. oprs" T))
    NIL])
```

**(CHANGERECORD**

```
[LAMBDA (RNAME RFIELDS OLDFLG) ; Edited 3-Jun-88 12:12 by jrb:
  (AND MSDATABASELST OLDFLG (MSNEEDUNSAVE (PROG ((FNLIST (GETRELATION RNAME ' (USE RECORDS)
    T)))
    (for F in RFIELDS
      do (SETQ FNLIST (UNION (GETRELATION F
        ' (USE FIELDS)
        T)
        FNLIST)))
    (RETURN FNLIST))
    "records" MSRECORDTRANFLG])
```

**(MSNEEDUNSAVE**

```
[LAMBDA (FNS MSG MARKCHANGEFLG) (* rmk%: "22-MAY-81 13:23")
  (AND MARKCHANGEFLG (MSMARKCHANGE1 FNS))
  (COND
    ((AND CHECKUNSAVEFLG (SETQ FNS (for FN inside FNS
      when (NOT (OR (EXPRP (OR (GETP FN 'BROKEN)
        (GETP FN 'ADVISED)
        FN))
      (FMEMB FN MSNEEDUNSAVE))))
      collect FN)))
    (COND
      ((EQ CHECKUNSAVEFLG '!))
      (UNSAVEFNS FNS))
      (T (printout T "The functions " .PARA2 0 0 FNS " use " MSG " which have changed." T "Call UNSAVEFNS()
        to load and/or UNSAVEDEF them." T)
        (/SETATOMVAL 'MSNEEDUNSAVE (NCONC FNS MSNEEDUNSAVE]))
```

**(UNSAVEFNS**

```
[LAMBDA (FNS) ; Edited 3-Jun-88 12:24 by jrb:
  (OR FNS (SETQ FNS (APPEND MSNEEDUNSAVE)))
  (for FN in FNS when FN bind FNTYPE
    do [SETQ FNTYPE (for FNREC in MSFNTYPES when (HASDEF FN (fetch (MSANALYZABLE FILEPKGNAME)
      FNREC)
      ' ?
      ' (NOERROR))
      do (RETURN (fetch (MSANALYZABLE FILEPKGNAME)
        FNREC)
        FNREC)
      (OR (EXPRP (OR (GETP FN 'BROKEN)
        (GETP FN 'ADVISED)
        FN))
        (PROG NIL
          (COND
            ((FGETD FN)
              (VIRGINFN FN T)
              (SAVEDEF FN)))
            (SELECTQ RECOMPILEDDEFAULT
              (CHANGES
                (RESETVARS (MSDATABASELST) (* don't mark as changed)
                  (MARKASCHANGED FN FNTYPE)) (* ASSERT%: ((REMOTE CALL) MSMARKCHANGED))
                (EXPRS (for FL in (WHEREIS FN FNTYPE FILELST)
                  unless [OR (FMEMB FL NOTCOMPILEDFILES)
                    (CDR (GETP FL 'FILE)
                  do (/SETATOMVAL 'NOTCOMPILEDFILES (CONS FL NOTCOMPILEDFILES))))
                NIL)
            (COND
              ((HASDEF FN FNTYPE 'SAVED)
                (PRINTOUT T "unsaving " FN T)
                (UNSAVEDEF FN FNTYPE))
              (T (PRINTOUT T "loading " FN T)
```

```

      (LOADDEF FN FNTYPE '?]
      (//SETATOMVAL 'MSNEEDUNSAVE (REMOVE FN MSNEEDUNSAVE)))
      (AND FNS (EQ RECOMPILEDEFAULT 'CHANGES)
      (printout T "WARNING: you must set RECOMPILEDEFAULT to EXPRS in order to have these functions
      recompiled automatically" T])
)

```

```
(ADDTOVAR COMPILE.TIME.CONSTANTS )
```

```
(RPAQQ RECORDCHANGEFN CHANGERECORD)
```

```
(RPAQ? CHECKUNSAVEFLG T)
```

```
(RPAQ? MSNEEDUNSAVE )
```

```
(DECLARE%: EVAL@COMPILE DONTCOPY
```

```
(DECLARE%: EVAL@COMPILE
```

```
(PUTPROPS GETWORDTYPE MACRO [(WORD TYPE)
                              (CDR (FASSOC TYPE (GETHASH WORD MSWORDS)))]
)

```

;; interactive routines

```
(RPAQ MASTERSCOPEDATE "16-Jun-88")
```

```
(ADDTOVAR HISTORYCOMS %.)
```

```
(DEFINEQ
```

```
(%.
  [NLAMBDA MASTERSCOPECOMMAND
    (MASTERSCOPE MASTERSCOPECOMMAND)]
  (* Imm "16-MAY-78 01:07")

```

**(MASTERSCOPE**

```
[LAMBDA (MASTERSCOPECOMMAND TOPFLG) ; Edited 5-Dec-86 06:08 by Imm
```

;; Top level entry. If given a sentence, parse it and return; otherwise, enter into USEREXEC-like loop

```
(COND
  (MASTERSCOPECOMMAND (MSINTERPRET (MKLIST MASTERSCOPECOMMAND)
                                     (NOT TOPFLG)))
  (T (PRINTOUT T "Type Masterscope commands to the exec using the . command, e.g." T ". WHO CALLS
    'MASTERSCOPE" T]))

```

**(MASTERSCOPE1**

```
[LAMBDA NIL ; Edited 28-Jan-88 11:28 by jrb:
              ; merged from smL Loops Masterscope by JRB
```

```
(printout T "Masterscope " MASTERSCOPEDATE T)
(PROG (X (*PACKAGE* (CL:FIND-PACKAGE "IL")))
  ERLP
    [ERSETQ (PROGN (PROMPTCHAR "_." T LISPXHISTORY)
                  (SELECTQ (SETQ X (LISPXREAD T T))
                           ((E _)
                            (LISPX (LISPXREAD T T)
                                     '_)
                           ((OK STOP)
                            (RETFROM (FUNCTION MASTERSCOPE1)))
                           (LISPX X '_ NIL (FUNCTION MASTERSCOPEEXEC)

```

```
(GO ERLP)])
```

**(MASTERSCOPEEXEC**

```
[LAMBDA (X LINE) ; Edited 17-Jun-87 16:57 by jrb:
                  (* Called via the LISPX in MASTERSCOPE)
                  ; Merged from smL Loops Masterscope by JRB
```

```
(PROG (MASTERSCOPECOMMAND)
  (AND [OR [COND
        ((NULL LINE)
         (OR (NOT (LITATOM X))
              (OR (NEQ (EVALV X)
                       'NOBIND)
                  (STRPOSL CLISPCHARRAY X)
         (AND (LITATOM X)
               (FGETD X)
               (LISTP LINE)
               (OR [COND
                   ((NULL (CDR LINE))
                    (OR (NULL (CAR LINE))
                        (LISTP (CAR LINE)
                   (EQ (ARGTYPE X)
                        3]
        (* "EDITF ] " OR SETQ (A B) TYPE ENTRY)

```

```
(RETURN))
```

(\* If MASTERSCOPEEXEC returns NIL, then LISPX will handle

the event as a normal typin)

```

(SETQ MASTERSCOPECOMMAND (CONS X LINE))
(SELECTQ (CAR MASTERSCOPECOMMAND)
  ((OK STOP BYE ok stop)
   (RETFROM 'MASTERSCOPE1 NIL T))
  NIL)
LISPXVALUE
[AND (LISTP LISPXHIST)
  (FRPLACA LISPXHIST (CONS '%. (CAR LISPXHIST)
    (SETQ LISPXVALUE (MSINTERPRET MASTERSCOPECOMMAND))
    (RETURN T])
  )

```

(\* Make sure the event shows up with a % in it)

:: Interpreting commands

(DEFINEQ

**(MSINTERPRETSET**

[LAMBDA (SET OP ARG)

(\* DECLARATIONS%: (RECORDS SENTENCE MSSETPHRASE))  
; Edited 3-Jun-88 12:42 by jrb:

```

(PROG (TEM TYPE (REP (fetch (MSSETPHRASE REP) of SET)))
  START
  [COND
    [(NLISTP REP)
     (OR (NULL REP)
        (SHOULDNT (LIST REP TEM TYPE ARG)))]
    [COND
      ((fetch (MSSETPHRASE TYPE) of SET)
       [replace (MSSETPHRASE REP) of SET with (create

```

```

INRELATION
HTABLES _ (for TYPE
            inside (fetch (MSSETPHRASE TYPE)
                          of SET)
            join (GETVERBTABLES 'IS TYPE))
OSET _ (create MSSETPHRASE
              DET _ 'ANY]

```

```

(RETURN (MSINTERPRETSET SET OP ARG])
(RETURN (SELECTQ OP
  (CHECK ARG)
  (LIST MSBLIP)
  (HARD (LISTHARD SET))
  (MEMB T)
  (SHOULDNT 8])
(T
  (RETURN
    (SELECTQ (fetch (SENTENCE ID) of REP)
      (APPLY (SELECTQ OP
        (CHECK ARG)
        (HARD (LISTHARD SET))
        (LIST MSBLIP)
        (MEMB (APPLY* (fetch (APPLY PRED) of REP)
                      ARG))
        (SHOULDNT 9)))
      (NOT (SELECTQ OP
        (CHECK (MSINTERPRETSET (fetch NEGATED of REP)
                               'CHECK ARG))
        (HARD (LISTHARD SET))
        (LIST MSBLIP)
        (MEMB (NOT (MSINTERPRETSET (fetch NEGATED of REP)
                                   'MEMB ARG)))
        (SHOULDNT 10)))
      (INRELATION (SELECTQ OP
        (CHECK ARG)
        ((LIST HARD)

```

:: got a list of dotted pairs of hash tables and another set; want to know the set of all things which  
:: have the given relation to any in the other set

```

(PROG ((HTABS (fetch (INRELATION HTABLES) of REP))
      (INVERTED (fetch (INRELATION INVERTED) of REP))
      (OTHERSET (fetch (INRELATION OSET) of REP))
      V SET2VAL)
  (SETQ SET2VAL (MSINTERPRETSET OTHERSET 'LIST))
  [COND
    [(EQ SET2VAL MSBLIP)
     (for R in HTABS do (MAPTABLE (COND
       (INVERTED (CDR R))
       (T (CAR R)))
       (FUNCTION MSINTERPA]

```

```

(T
  (for R in HTABS
    do (for X in SET2VAL
      do (SETQ V (UNION [GETTABLE X (COND
        (INVERTED
         (CAR R))
        (T (CDR R]

```





```

                                with (create QUOTE
                                        QUOTED _ S1)))
                                MSBLIP)
                                (T (UNION S1 S2])
(MEMB (OR (MSINTERPRETSET (fetch (CSET SET1) of REP)
                            'MEMB ARG)
          (MSINTERPRETSET (fetch (CSET SET2) of REP)
                            'MEMB ARG)))
(SHOULDNT 14)))
(AND ; I.e. WHO ON MYFILE AND @ EXPRP CALL Z -- only if both
      ; sets are vague does the world need updating
(SELECTQ OP
  (CHECK ([LAMBDA (X)
    (OR (MSINTERPRETSET (fetch (CSET SET2) of REP)
                        'CHECK ARG)
        X]
      (MSINTERPRETSET (fetch (CSET SET1) of REP)
                        'CHECK ARG)))
  ((HARD LIST)
   [PROG (S1 S2)
    (RETURN (COND
      [[EQ MSBLIP (SETQ S1 (MSINTERPRETSET (fetch (CSET SET1)
                                                of REP)
                                            'LIST]
      (COND
        ((EQ MSBLIP (SETQ S2 (MSINTERPRETSET (fetch (CSET SET2)
                                                    of REP)
                                                OP)))
        MSBLIP)
        (T (SUBSET S2 (FUNCTION (LAMBDA (X)
          (MSINTERPRETSET
            (fetch (CSET SET1) of REP)
            'MEMB X]
          (T (SUBSET S1 (FUNCTION (LAMBDA (X)
            (MSINTERPRETSET
              (fetch (CSET SET2) of REP)
              'MEMB X]))
          (MEMB (AND (MSINTERPRETSET (fetch (CSET SET1) of REP)
                                    'MEMB ARG)
                  (MSINTERPRETSET (fetch (CSET SET2) of REP)
                                    'MEMB ARG)))
      (SHOULDNT 15)))
(ANDNOT (replace (SENTENCE ID) of REP with 'AND)
[replace (MSSETPHRASE REP) of (fetch SET2 of REP)
 with (create NOT
        NEGATED _ (create MSSETPHRASE using (fetch SET2 of REP)
        REP _ (fetch (MSSETPHRASE REP)
                    of (fetch SET2 of REP)
        (GO RETRY))
(IN [SETQ REP (create QUOTE
                    QUOTED _ (MKLIST (CL:EVAL (fetch (IN EXPRESSION) of REP]
(GO RETRY))
(BLOCKS ; Block set
  (SELECTQ OP
    (CHECK [[LAMBDA (X Y)
      (OR X Y]
      (AND (fetch FNS of REP)
           (MSINTERPRETSET (fetch FNS of REP)
                           'CHECK))
      (AND (fetch FILES of REP)
           (MSINTERPRETSET (fetch FILES of REP)
                           'CHECK))
    (PROGN [SETQ REP (create QUOTE
                        QUOTED _ (MSGETBLOCKDEC (fetch TYPES of REP)
                                                (fetch FNS of REP)
                                                (AND (fetch FILES of REP)
                                                    (MSINTERPRETSET (fetch FILES
                                                                    of REP)
                                                                    'HARD]
      (GO RETRY))))
(FIELDS (SELECTQ OP
  (CHECK (MSINTERPRETSET (fetch RECS of REP)
                        OP))
  (PROGN [SETQ REP (create QUOTE
                        QUOTED _
                        (PROG (VAL)
                          (for X in (MSLISTSET (fetch RECS of REP)
                                                T)
                            do (SETQ VAL (UNION (RECORDFIELDNAMES X)
                                                VAL)))
      (RETURN VAL]
  (GO RETRY))))
(THAT (PROG (TABLES (MSVERB (fetch (THAT MSVERB) of REP))
  VALUE
  (OS (fetch (THAT OTHERSET) of REP)))
  (SELECTQ (fetch (MSVERB ROOT) of MSVERB)
  ((AND OR ANDNOT)

```

```

[SETQ REP (create CSET
ID _ (fetch C of (fetch VPART of MSVERB))
SET1 _
(create MSSETPHRASE
using SET REP _
(create THAT
MSVERB _
(create MSVERB
TENSE _ (fetch TENSE of MSVERB)
VPART _
(fetch VB1
of (fetch VPART of MSVERB)))
OTHERSET _ OS))
SET2 _
(create MSSETPHRASE
using SET REP _
(create THAT
MSVERB _
(create MSVERB
TENSE _ (fetch TENSE of MSVERB)
VPART _
(fetch VB2
of (fetch VPART of MSVERB)))
OTHERSET _ OS]
(GO RETRY))
(CALL (COND
((EQ (fetch (MSVERB MODIFIER) of MSVERB)
'SOMEHOW)
[SETQ REP
(create PATHS
MSPATHOPTIONS _
(COND
((EQ (fetch TENSE of MSVERB)
'ED)
(create PATHOPTIONS
FROM _ OS
TO _ (create MSSETPHRASE)
TOPFLG _ T))
(T (create PATHOPTIONS
TO _ OS
TOPFLG _ T]
(GO RETRY))))
(CONTAIN (COND
((EQ (fetch DET of OS)
'WHICH)
(SHOULDNT 16)))
;; JRB - Default types on files are now ignored - removed ((fetch| (MSSETPHRASE
;; DEFAULTTYPE) |of| SET) from ORs below.
[SETQ REP (create QUOTE
QUOTED _
(SELECTQ (fetch TENSE of MSVERB)
(ED (ONFILE (MSINTERPRETSET OS 'HARD)
(OR (fetch (MSVERB MODIFIER)
of MSVERB)
(fetch (MSSETPHRASE TYPE)
of SET)
'ALL))))
(ONFILE NIL (OR (fetch (MSVERB MODIFIER)
of MSVERB)
(fetch (MSSETPHRASE TYPE)
of OS)
'ALL)
(OR (MSINTERPRETSET OS 'HARD)
T]
(GO RETRY))
NIL)
(SELECTQ OP
(CHECK (SETQ VALUE (MSINTERPRETSET OS 'CHECK (fetch KNOWN of OS))))
NIL)
(SETQ TABLES (GETVERTABLES (fetch (MSVERB ROOT) of MSVERB)
(fetch (MSVERB MODIFIER) of MSVERB)))
(replace REP of SET with (SETQ REP (create INRELATION
INVERTED _ (EQ (fetch TENSE of MSVERB)
'ED)
HTABLES _ TABLES
OSET _ OS)))
OUT (RETURN (OR (MSINTERPRETSET SET OP ARG)
VALUE))))
(PATHS (COND
((EQ OP 'CHECK)
(CHECKPATHS (fetch MSPATHOPTIONS of REP)))
(T (SETQ REP (create GETHASH
HTABLE _ (LIST (MSONPATH REP))
BADMARKS _ T))
(GO RETRY))))
(SHOULDNT 17]

```

```

RETRY
  (replace REP of SET with REP)
  (GO START])

```

**(MSINTERPA**

```

[LAMBDA (VAL KEY)
  (AND (NOT (FMEMB KEY V))
    [COND
      ((AND (NULL (fetch (MSSETPHRASE TYPE) of OTHERSET))
        (NULL (fetch REP of OTHERSET)))
        VAL)
      (T (find Z in VAL suchthat (MSINTERPRETSET OTHERSET 'MEMB Z]
        (SETQ V (CONS KEY V]))
    ]
  )
  (* DECLARATIONS%: (RECORDS SETPHRASE))
  ; Edited 12-Jan-87 01:20 by jds

```

**(MSGETBLOCKDEC**

```

[LAMBDA (TYPE FNSET FILES)
  (PROG (VAL)
    [for FILE inside (OR FILES FILELST)
      do ([for BLOCK in (FILECOMSLST FILE 'BLOCKS) when (OR (NULL FNSET)
        (SOME BLOCK (FUNCTION (LAMBDA (FILE)
          (AND (LITATOM FILE)
            (MSMEMBSET
              FILE FNSET]
          )
        ]
      ]
    ]
    do ([SELECTQ TYPE
      ((BLKFNS BLOCK NIL)
        (for FILE in (CDR BLOCK) when (AND (LITATOM FILE)
          (NOT (FMEMB FILE VAL))))
          do (SETQ VAL (CONS FILE VAL)))
        (for Y in BLOCK when (AND (LISTP Y)
          (EQMEMB (CAR Y)
            TYPE))
          do (SETQ VAL (UNION (COND
            ((EQ (CADR Y)
              '*))
            (EVAL (CADDR Y)))
            (T (CDR Y)))
            VAL]
          (COND
            ((AND (EQ TYPE 'ENTRIES)
              (CAR BLOCK)
              (FMEMB (CAR BLOCK)
                (CDR BLOCK))
              (NOT (FMEMB (CAR BLOCK)
                VAL)))
              (SETQ VAL (CONS (CAR BLOCK)
                VAL]
            (OR FNSET (SETQ VAL (UNION (FILECOMSLST FILE (SELECTQ TYPE
              (BLKFNS 'FNS)
                TYPE))
                VAL]
          )
        (RETURN VAL])
    ]
  )
  (* Imm "24-FEB-79 20:50")

```

**(LISTHARD**

```

[LAMBDA (SET)
  (PROG (VAL)
    [for TYPE inside (OR (fetch (MSSETPHRASE TYPE) of SET)
      (fetch (MSSETPHRASE DEFAULTTYPE) of SET))
      do (for TABLE in (GETVERBTABLES 'IS (COND
        ((AND (EQ TYPE 'FNS)
          (fetch (MSSETPHRASE KNOWN) of SET))
          'KNOWN)
        (T TYPE)))
        do (SETQ VAL (MSHASHLIST (CAR TABLE)
          VAL SET]
      )
    (RETURN VAL])
  )
  (* DECLARATIONS%: (RECORDS MSSETPHRASE))
  ; Edited 12-Jan-87 00:59 by jds

```

**(MSMEMBSET**

```

[LAMBDA (ITEM SET)
  (MSINTERPRETSET SET 'MEMB ITEM])
  (* Imm%: 25-JAN-76 2 20)

```

**(MSLISTSET**

```

[LAMBDA (SET TRYHARD TYPE)
  (* Interpret set as List -
  return list of elements in set S, or MSBLIP if can't)
  (MSINTERPRETSET SET (COND
    (TRYHARD 'HARD)
    (T 'LIST))
  TYPE])
  (* Imm " 8-JUL-78 02:11")

```

**(MSHASHLIST**

[LAMBDA (HTABLE PREVVALUE OTHERSET BADMARKS) (\* Imm " 8-AUG-77 15:17")  
(MAPTABLE HTABLE (FUNCTION MSHASHLIST1))  
PREVVALUE]]

**(MSHASHLIST1**

[LAMBDA (VAL KEY) (\* Imm " 8-AUG-77 15:16")  
(AND (NEQ VAL MSBLIP)  
(NOT (EQMEMB VAL BADMARKS))  
(NOT (FMEMB KEY PREVVALUE))  
(OR (NULL OTHERSET)  
(MSMEMBSET KEY OTHERSET))  
(SETQ PREVVALUE (CONS KEY PREVVALUE))

**(CHECKPATHS**

[LAMBDA (OPTIONS VAL) (\* Imm "20-DEC-78 20:03")  
(PROG (VAL)  
(for PR in OPTIONS when (FMEMB (CAR PR)  
'(FROM TO AVOIDING NOTRACE MARKING SEPARATE))  
do (AND (MSINTERPRETSET (CDR PR)  
'CHECK  
(EQ (CAR PR)  
'FROM))  
(SETQ VAL T)))  
(RETURN (OR VAL (NULL (FASSOC 'FROM OPTIONS))

**(ONFILE**

[LAMBDA (FILES TYPES FINDITEMS) ; Edited 2-Jun-88 11:19 by jrb:  
; MSHASHFILE uses cause GETRELATION barfs if CONTAINS  
; table doesn't exist.

(PROG (VAL)  
;; JRB - TYPES of 'ALL means gather all types Masterscope knows about  
[AND (EQ TYPES 'ALL)  
(SETQ TYPES (for FT in MSFNITYPES collect (fetch (MSANALYZABLE FILEPKGNAME) of FT]  
[for FILE (FNSONLY \_ (AND MSHASHFILE (SELECTQ (COND  
(AND (LISTP TYPES)  
(NULL (CDR TYPES))  
(CAR TYPES)))  
(T TYPES))  
((FNS KNOWN NIL)  
T)  
NIL))]  
inside (OR FILES FILELST)  
do ; Don't notice the file if we only care about FNS and the file is  
; known to the database.  
(COND  
[(AND FNSONLY (NOT (MEMB FILE FILELST))  
(GETRELATION FILE 'CONTAINS]  
(T (MSNOTICEFILE FILE)))  
(for TYPE inside TYPES do (SETQ TYPE (SELECTQ TYPE  
((FNS KNOWN NIL)  
'FNS)  
TYPE))  
(COND  
[FINDITEMS (OR (FMEMB FILE VAL)  
(AND (find X inside FINDITEMS  
suchthat (INFILECOMS? X TYPE (FILECOMS FILE)))  
(SETQ VAL (CONS FILE VAL]  
(T (SETQ VAL (UNION (FILECOMSLST FILE TYPE)  
VAL]  
[COND  
(AND MSHASHFILE (NULL VAL)  
(find TYPE inside TYPES suchthat (SELECTQ TYPE  
((FNS KNOWN NIL)  
T)  
NIL)) ; Didn't find it in core; perhaps the CONTAINS table knows  
(COND  
[FILES (for FILE inside FILES do (COND  
(FINDITEMS (for X inside FINDITEMS  
when (TESTRELATION X 'CONTAINS FILE T)  
do (pushnew VAL FILE))]  
(T (SETQ VAL (UNION (GETRELATION FILE 'CONTAINS)  
VAL]  
(FINDITEMS  
;; No files: should use all known files, but that information isn't explicitly kept by MSHASH. Soooo, we'll only do the  
;; case where FINDITEMS is given  
(for X inside FINDITEMS do (SETQ VAL (UNION (GETRELATION X 'CONTAINS T)  
VAL]  
(RETURN VAL])

)

(DEFINEQ

**(MSINTERPRET**

; Edited 8-Apr-88 10:23 by jrb:

```
[LAMBDA (COMMAND SUBROUTINE)
  (PROG (VAL EDITQUIETFLG)
    (SELECTQ (CAR COMMAND)
      ((; * -)
        (RETURN))
      NIL)
    (SETQ VAL (MSPARSE COMMAND))
    (COND
      ((EQ MSPRINTFLG T)
        (PRINT VAL T)))
    (COND
      ((EQ (CAR VAL)
        'OUTPUT)
        (MSOUTPUT (CADR VAL))
        (SETQ VAL (CDDR VAL))
        (MAPRINT COMMAND NIL ". " "
          ")))
      (* Now to interpret)
    (COND
      ((AND (EQ (CAR VAL)
        'ERASE)
        (NULL (CDR VAL)))
        (MSERASE T)
        (RETURN 'ok]
    (MSINIT)
    (RETURN
      (SELECTQ (fetch (SENTENCE ID) of VAL)
        (REANALYZE
```

; Definitely don't want to CHECKFORCHANGED before the  
; ANALYZE is done

:: From Lanning's Loops changes for Masterscope...

:: (MAPC (MSLISTSET (CDR VAL) T 'KNOWN) (FUNCTION (LAMBDA (X) (UPDATEFN X T)))) (CL:VALUES)

:: JRB - The MSANALYZEFNS hashtable hook is hereby flushed.

```
(LET* [[SETTYPE (COND
  ((fetch (MSSETPHRASE TYPE) of (CDR VAL)))
  ((fetch (MSSETPHRASE DEFAULTTYPE) of (CDR VAL))
  (SET (MSLISTSET (CDR VAL)
    T
    'KNOWN]
```

:: SETTYPE is allowed to be NIL here...

```
(if [AND SETTYPE (NULL (for MT in MSFNTPES
  thereis (EQ SETTYPE (fetch (MSANALYZABLE SETNAME)
    of MT]
  then (PRINTOUT T "Sorry, can't analyze " SETTYPE T)
    (ERROR!))
  else (for X in SET do (UPDATEFN X T))
    (CL:VALUES)))]
```

```
(ANALYZE (CHECKFORCHANGED (SETQ VAL (CDR VAL)))
```

:: From Lanning's Loops changes for Masterscope...

:: (COND ((EQ (SETQ VAL (MSLISTSET VAL NIL 'KNOWN)) MSBLIP) (|printout| T "Sorry, can't figure out which  
:: functions you mean." T) (ERROR!))) (MAPC VAL (FUNCTION UPDATEFN)) (CL:VALUES)

```
[LET* [[SETTYPE (COND
  ((fetch (MSSETPHRASE TYPE) of VAL))
  ((fetch (MSSETPHRASE DEFAULTTYPE) of VAL)]
  (SET (MSLISTSET VAL T 'KNOWN]
```

:: SETTYPE is allowed to be NIL here...

```
(COND
  ([AND SETTYPE (NULL (for MT in MSFNTPES thereis (EQ SETTYPE (fetch (MSANALYZABLE
    SETNAME)
    of MT]
    (PRINTOUT T "Sorry, can't analyze " SETTYPE T)
    (ERROR!))
  ((EQ SET MSBLIP)
  (PRINTOUT T "Sorry, can't figure out which items you mean. " T)
  (ERROR!))
  (T (for X in SET do (UPDATEFN X T NIL SETTYPE))
    (CL:VALUES])
```

((EDIT SHOW)

```
[PROG (DONE NEEDUPDATE UPDATEALL TYPE (EDIT (fetch (SENTENCE ID) of VAL))
  REL SHOWSET (EDITCOMS (fetch OTHERSTUFF of VAL))
  (SUBJECT (fetch (SENTENCE SUBJECT) of VAL))
  (MSPRED (fetch (SENTENCE MSPRED) of VAL))
  REP)
```

(**DECLARE** (SPECVARS TYPE SHOWSET EDIT EDITCOMS DONE))

```
[COND
  ((NULL MSPRED)
  (CHECKFORCHANGED SUBJECT)
  (RETURN (MAPC (MSLISTSET SUBJECT T)
    (FUNCTION (LAMBDA (FN)
```

; EDIT ANY CALLING FOO -- just call EDITFNS

```

(PRIN2 FN T)
(PRIN1 " :
" T)
(OR (NLSETQ (PRINT (APPLY 'MSEDITF (CONS FN EDITCOMS)
T))
(PRINT "failed" T)
[SETQ REL (fetch (THAT MSVERB) of (SETQ REP (fetch (MSSETPHRASE REP) of MSPRED)
[SETQ SHOWSET (fetch (THAT OTHERSET) of REP)
[COND
(EQ (fetch TENSE of REL)
'ED)
(replace TENSE of REL with 'S)
(SETQ MSPRED (create MSSETPHRASE
REP _ (create THAT
MSVERB _ REL
OTHERSET _ (SETQ SHOWSET (PROG1 SUBJECT
(SETQ SUBJECT
SHOWSET]
(SETQ TYPE (VERBNOTICELIST (fetch VPART of REL)))
(SETQ UPDATEALL (MSINTERPRETSET SUBJECT 'CHECK T))
[for FN in NEEDUPDATE do (COND
((GETHASH FN MSCHANGEDARRAY)
(MSSHOWUSE FN TYPE SHOWSET EDIT NIL EDITCOMS)
(SETQ DONE (CONS FN DONE)))
(T (UPDATEFN FN)
(COND
(UPDATEALL [MAPHASH MSCHANGEDARRAY (FUNCTION (LAMBDA (VAL KEY)
(AND (OR (EQ VAL T)
(TESTRELQ KNOWN KEY)
(TESTRELQ (CALL NOTERROR)
KEY T))
(COND
((MSSHOWUSE KEY TYPE
SHOWSET EDIT
'CHANGED EDITCOMS)
(SETQ DONE (CONS KEY DONE)
(MSCHECKEMPTY)))
(MAPC (MSLISTSET (MSJOINSET 'AND MSPRED SUBJECT)
T)
(FUNCTION (LAMBDA (AT)
(AND (NOT (FMEMB AT DONE))
(MSSHOWUSE AT TYPE SHOWSET EDIT NIL EDITCOMS]
(CL:VALUES))
(? [CHECKFORCHANGED (SETQ VAL (MSJOINSET 'AND (fetch MSPRED of VAL)
(fetch SUBJECT of VAL]
(OR SUBROUTINE (TAB 0 0))
(MSSOLVE VAL))
(PATHS (PROG ([INVERTED
(for X on (CDR VAL) bind FROMFOUND
do (SELECTQ (CAAR X)
(FROM (SETQ FROMFOUND T))
(TO (RETURN (NOT FROMFOUND)))
NIL)
finally (RETURN (COND
(FROMFOUND NIL)
(T (FRPLACD VAL
(CONS [CONS 'FROM
(create MSSETPHRASE
REP _
(create THAT
MSVERB _
(create MSVERB
ROOT _ 'IS
MODIFIER _
' KNOWN)
OTHERSET _ (create
MSSETPHRASE
]
(CDR VAL)))
NIL]
NEEDUPDATE UPDATEALL TEM)
(SETQ UPDATEALL (CHECKPATHS (fetch MSPATHOPTIONS of VAL)))
(for X in NEEDUPDATE do (UPDATEFN X))
(COND
(UPDATEALL (UPDATECHANGED)
(MSCHECKEMPTY)))
(COND
((SETQ TEM (fetch OUTPUT of (CDR VAL)))
(MSOUTPUT TEM)))
(AND (SETQ TEM (fetch (PATHOPTIONS LINELENGTH) of (CDR VAL)))
(RESETSAVE (LINELENGTH TEM)))
;; Display paths; must print all of FROM, with separate tree for all of SEPARATE (considered as a subset of FROM).
;; Stop when you get to a function in NOTRACE, -- unless TO is NIL, only print paths that eventually reach an element of
;; TO. If INVERTED is not NIL, print inverted tree. Do not print out functions in AVOIDING
[SETQ MSTHOSE (MSPATHS [COND
(INVERTED (fetch TO of (fetch MSPATHOPTIONS of VAL)))

```

```

(T (fetch FROM of (fetch MSPATHOPTIONS of VAL)
[COND
  (INVERTED (fetch FROM of (fetch MSPATHOPTIONS of VAL)))
  (T (fetch TO of (fetch MSPATHOPTIONS of VAL)
INVERTED
(fetch AVOIDING of (fetch MSPATHOPTIONS of VAL))
(fetch SEPARATE of (fetch MSPATHOPTIONS of VAL))
(fetch NOTRACE of (fetch MSPATHOPTIONS of VAL))
(fetch MARKING of (fetch MSPATHOPTIONS of VAL)
(RETURN (CL:VALUES))))
(ERASE (MSERASE (MSLISTSET (CDR VAL)
T
'KNOWN))
(PRIN1 "Erased." T)
(CL:VALUES))
(DESCRIBE (CHECKFORCHANGED (CDR VAL)
NIL T)
(TAB 0 0)
(MAPC (MSLISTSET (CDR VAL)
T)
(FUNCTION MSDESCRIBE)))
(FOR (CHECKFORCHANGED (CADDR VAL)
(FRPLACA (CADDR VAL)
(KWOTE (MSLISTSET (CADDR VAL)
T)))
(EVAL VAL))
(CHECK (CHECKFORCHANGED (CDR VAL)
[MSCHECKBLOCKS (AND (CDR VAL)
(MSLISTSET (CDR VAL)
'HARD
'FILES])
(SHOULDNT 18])
; case of plain ERASE taken care of earlier
; Need to update the world since will print out CALLED BY:

```

(VERBNOTICELIST

```

[LAMBDA (VPART) ; Edited 12-Jun-87 16:37 by jrb:

```

;;; NOTE: The call to MSVBTABLES used to be a call to the macro MSVBNOTICED. The macro, however, existed only on the file MSANALYZE.  
 ;;; Further, there was an EXPR definition for the fn MSVBNOTICED, but BvM could find no reference to it in any file. What is going on? - smL

```

(COND
  [(type? CVERB VPART)
  (UNION (VERBNOTICELIST (fetch VB1 of VPART))
  (VERBNOTICELIST (fetch VB2 of VPART)
  (T (OR (MSVBTABLES (fetch (VPART ROOT) of VPART)
(fetch (VPART MODIFIER) of VPART))
(PROGN (printout T "can't SHOW or EDIT where things " (fetch (VPART ROOT) of VPART)
%,
(OR (fetch (VPART MODIFIER) of VPART)
"")
"! " T)
(ERROR!))

```

(MSOUTPUT

```

[LAMBDA (FILE) (* Imm "13-AUG-77 12:16")
(* OUTPUT is already RESETSAVE'd)
[COND
  ((OPENP FILE 'OUTPUT)
  (OUTPUT FILE))
  (T (OUTFILE FILE)
  (RESETSAVE NIL (LIST 'CLOSEF FILE)
  (LINELENGTH FILELINELENGTH]))
(* output to file, reset LINELENGTH)

```

(MSCHECKEMPTY

```

[LAMBDA NIL (* Imm "20-JAN-79 14:08")
(PROG (Q CF)
(COND
  (MSDBEMPTY (printout T "No functions have been analyzed!" T)
  (UPDATEFILES)
  (SETQ CF (FILEPKGCHANGES 'FNS))
  [COND
    ((AND [SETQ Q (APPEND (AND FILELST (LIST 'ON '%' FILELST))
    (AND CF FILELST ' (OR))
    (AND CF (LIST 'IN '%' CF)
    (EQ [ASKUSER (AND (FIXP DWIMWAIT)
    (ITIMES 10 DWIMWAIT))
    ' (Y)
    (CONS "want to ." (SETQ Q (APPEND ' (ANALYZE THE FNS)
    Q)))
    ' ((Y "es
    ")
    (N "o
    "]
    ' Y))

```

```

(MASTERSCOPE Q)
(COND
  (MSDBEMPTY (printout T "Sorry, no functions were found to analyze!" T))
  (T (RETURN]
(ERROR!])

```

**(CHECKFORCHANGED**

```

[LAMBDA (SET NOTTHISONE UPDATEALL) (* Imm "25-JUN-78 01:03")
  (PROG (NEEDUPDATE)
    (SETQ UPDATEALL (OR (MSINTERPRETSET SET 'CHECK (AND (NOT NOTTHISONE)
      (fetch KNOWN of SET)))
      UPDATEALL))
    (for X in NEEDUPDATE do (UPDATEFN X))
    (COND
      (UPDATEALL (UPDATECHANGED)
        (MSCHECKEMPTY]))

```

**(MSSOLVE**

; Edited 12-Jan-87 13:43 by jds

```

[LAMBDA (SET)
  (SETQ MSTHOSE (MSLISTSET SET T))
  (PROG (ND QT OSET REP)
    (SETQ REP (fetch REP of SET))
    [OR (SELECTQ (fetch (SENTENCE ID) of REP)
      (AND (SETQ ND (fetch SET2 of REP))
        (AND (EQ [fetch (SENTENCE ID) of (SETQ REP (fetch REP of (fetch SET1 of REP]
          'INRELATION)
          (EQ (fetch DET of (SETQ OSET (fetch OSET of REP)))
            'WHICH)))
        (INRELATION (EQ (fetch DET of (SETQ OSET (fetch OSET of REP)))
          'WHICH))
      NIL)
    (RETURN (COND
      ((EQ (fetch (MSSETPHRASE DET) of SET)
        'WHICH)
        MSTHOSE)
      (T (NOT (NULL MSTHOSE]
    (replace REP of SET with REP)
    (replace INVERTED of REP with (NOT (fetch INVERTED of REP)))
    [replace OSET of REP with (create MSSETPHRASE
      REP _ (create QUOTE
        QUOTED _ (SETQ QT (LIST NIL]
    [MAPC MSTHOSE (FUNCTION (LAMBDA (FN)
      (PRIN2 FN)
      (PRIN1 " -- ")
      (FRPLACA QT FN)
      (PRINT (SUBSET (MSLISTSET SET T)
        (FUNCTION (LAMBDA (X)
          (MSMEMBSET X OSET]
    (RETURN (CL:VALUES])

```

```

)
(DECLARE%: DONTCOPY
(DECLARE%: EVAL@COMPILE
(RECORD GETHASH (ID HTABLE . BADMARKS)
  ID _ 'GETHASH)
(RECORD INRELATION (ID (INVERTED . HTABLES) . OSET)
  ID _ 'INRELATION)
(ASSOCRECORD PATHOPTIONS (TO FROM AVOIDING SEPARATE NOTRACE TOPFLG OUTPUT LINELENGTH MARKING)
  (* CHECKPATHS assumes that this is an ASSOCRECORD)
)
(RECORD MSANALYZABLE (FILEPKGNAME SETNAME GETDEF-FN MARKCHANGED-FN))
)
)
(FILESLoad MSCOMMON)
(DECLARE%: DONTCOPY
(RPAQQ MSCOMPILETIME
  [[P (MAPC ' (GETRELQ TESTRELQ SCRATCHASH)
    (FUNCTION (LAMBDA (X)
      (PUTHASH X 'MACRO USERTEMPLATES]
    (BLOCKS (NIL % . MSMARKCHANGE1 MSFIND (LOCALVARS . T))
      (MSSTOREDATA MSSTOREDATA MSCOLLECTDATA (LOCALFREEVARS FNDATA)
        (NOLINKFNS . T))
      (MASTERSCOPEBLOCK MSINTERPRETSET CHANGEI.S. CHANGERECD CHANGEVAR CHECKFORCHANGED CHECKPATHS
        DUMPDATABASE DUMPDATABASE1 FMAPRINT GETRELATION GETTEMPLATE GETVERBTABLES LISTHARD
        MAPRELATION MASTERSCOPE MASTERSCOPE1 MASTERSCOPEEXEC MSCHECKEMPTY MSCLOSEFILES MSDESCRIBE
        MSDESCRIBE1 MSERASE MGETBLOCKDEC MSHASHLIST MSHASHLIST1 MSINIT MSINTERPA MSINTERPRET

```



```

MSLISTSET MSMARKCHANGED MSMEMBSET MSNEEDUNSAVE MSNLAMBDAHECK MSNOTICEFILE MSOUTPUT
MSPRINHELPFILE MSSHOWUSE MSSOLVE MSUPDATE MSUPDATEFN1 ONFILE PARSERELATION PARSERELATION1
READATABASE SETTEMPLATE TEMPLATE TESTRELATION UNSAVEFNS UPDATECHANGED UPDATECHANGED1
UPDATEFN VERBNOTICELIST ADDTEMPLATEWORD MSADDANALYZE MSADDMODIFIER MSADDRELATION MSADDDTYPE
(ENTRIES CHANGERECD DUMPDATABASE DUMPDATABASE1 GETRELATION GETTEMPLATE MAPRELATION
MASTERSCOPE MASTERSCOPEEXEC MSCLOSEFILES MSHASHLIST1 MSINTERPA MSMARKCHANGED
MSMEMBSET MSLISTSET MSNEEDUNSAVE MSNOTICEFILE MSSHOWUSE PARSERELATION READATABASE
SETTEMPLATE TESTRELATION UNSAVEFNS UPDATECHANGED UPDATECHANGED1 UPDATEFN MSLISTSET
MSDESCRIBE ADDTEMPLATEWORD MSADDANALYZE MSADDMODIFIER MSADDRELATION MSADDDTYPE)

```

```

(RETURNS MASTERSCOPE1)
(SPECVARS ANYFOUND BADMARKS FNDA NEEDUPDATE OTHERSET PREVVALUE SHOWFN V VARS)
(NOLINKFNS . T))

```

```

(GLOBALVARS CHECKUNSAVEFLG CLISPCHARRAY CLISPIFYPRETTYFLG DWIMIFYCOMPFLG DWIMWAIT FILELINELENGTH FILELST
FILERDTBL LISPXISTORY MASTERSCOPEDATE MSBLIP MSCHANGEDARRAY MSDATABASEINIT NODUMPRELATIONS
MSDBEMPTY MSERRORFN MSFILELST MSHELPPFILE MSNEEDUNSAVE MSOPENFILES MSPRINTCNT MSPRINTFLG
MSRECORDTRANFLG MSTEMPLATES MSTHOSE NOTCOMPILEDFILES RECOMPILEDEFAULT TABLE.TO.NOTICED
USERTEMPLATES MSDATABASELST MSHASHFILE ANALYZEUSERFNS)
(DECLARE%: EVAL@COMPILE (P (CLISPDEC 'FAST)))

```

```

[MAPC ' (GETRELQ TESTRELQ SCRATCHASH)
(FUNCTION (LAMBDA (X)
(PUTHASH X 'MACRO USERTEMPLATES)

```

```

(DECLARE%: DONTVAL@LOAD DOEVAL@COMPILE DONTCOPY

```

```

(BLOCK%: NIL %. MSMARKCHANGE1 MSFIND (LOCALVARS . T))

```

```

(BLOCK%: MSSTOREDATA MSSTOREDATA MSCOLLECTDATA (LOCALFREEVARS FNDA)
(NOLINKFNS . T))

```

```

(BLOCK%: MASTERSCOPEBLOCK MSINTERPRETSET CHANGEI.S. CHANGERECD CHANGEVAR CHECKFORCHANGED CHECKPATHS
DUMPDATABASE DUMPDATABASE1 FMAPRINT GETRELATION GETTEMPLATE GETVERBTABLES LISTHARD MAPRELATION
MASTERSCOPE MASTERSCOPE1 MASTERSCOPEEXEC MSCHECKEMPTY MSCLOSEFILES MSDESCRIBE MSDESCRIBE1 MSERASE
MSGETBLOCKDEC MSHASHLIST MSHASHLIST1 MSINIT MSINTERPA MSINTERPRET MSLISTSET MSMARKCHANGED MSMEMBSET
MSNEEDUNSAVE MSNLAMBDAHECK MSNOTICEFILE MSOUTPUT MSPRINHELPFILE MSSHOWUSE MSSOLVE MSUPDATE MSUPDATEFN1
ONFILE PARSERELATION PARSERELATION1 READATABASE SETTEMPLATE TEMPLATE TESTRELATION UNSAVEFNS UPDATECHANGED
UPDATECHANGED1 UPDATEFN VERBNOTICELIST ADDTEMPLATEWORD MSADDANALYZE MSADDMODIFIER MSADDRELATION MSADDDTYPE
(ENTRIES CHANGERECD DUMPDATABASE DUMPDATABASE1 GETRELATION GETTEMPLATE MAPRELATION MASTERSCOPE
MASTERSCOPEEXEC MSCLOSEFILES MSHASHLIST1 MSINTERPA MSMARKCHANGED MSMEMBSET MSLISTSET MSNEEDUNSAVE
MSNOTICEFILE MSSHOWUSE PARSERELATION READATABASE SETTEMPLATE TESTRELATION UNSAVEFNS UPDATECHANGED
UPDATECHANGED1 UPDATEFN MSLISTSET MSDESCRIBE ADDTEMPLATEWORD MSADDANALYZE MSADDMODIFIER
MSADDRELATION MSADDDTYPE)
(RETURNS MASTERSCOPE1)
(SPECVARS ANYFOUND BADMARKS FNDA NEEDUPDATE OTHERSET PREVVALUE SHOWFN V VARS)
(NOLINKFNS . T))
)

```

```

(DECLARE%: DOEVAL@COMPILE DONTCOPY

```

```

(GLOBALVARS CHECKUNSAVEFLG CLISPCHARRAY CLISPIFYPRETTYFLG DWIMIFYCOMPFLG DWIMWAIT FILELINELENGTH FILELST
FILERDTBL LISPXISTORY MASTERSCOPEDATE MSBLIP MSCHANGEDARRAY MSDATABASEINIT NODUMPRELATIONS MSDBEMPTY
MSERRORFN MSFILELST MSHELPPFILE MSNEEDUNSAVE MSOPENFILES MSPRINTCNT MSPRINTFLG MSRECORDTRANFLG MSTEMPLATES
MSTHOSE NOTCOMPILEDFILES RECOMPILEDEFAULT TABLE.TO.NOTICED USERTEMPLATES MSDATABASELST MSHASHFILE
ANALYZEUSERFNS)
)

```

```

(DECLARE%: EVAL@COMPILE

```

```

(CLISPDEC 'FAST)
)
)

```

```

(DECLARE%: DONTVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS

```

```

(ADDOVAR NLAMA %.)

```

```

(ADDOVAR NLAML )

```

```

(ADDOVAR LAMA MSEDITF MSEDITF)
)

```

```

(PUTPROPS MASTERSCOPE COPYRIGHT ("Xerox Corporation" 1983 1984 1985 1986 1987 1988))

```

FUNCTION INDEX

%. . . . .	30	GETTEMPLATE . . . . .	8	MSCLOSEFILES . . . . .	5	MSNEEDUNSAVE . . . . .	29	ONFILE . . . . .	36
ADDDHASH . . . . .	13	GETVERBTABLES . . . . .	11	MSCOLLECTDATA . . . . .	5	MSNLAMBDAHECK . . . . .	4	PARSERELATION . . . . .	12
ADDTEMPLATEWORD . . . . .	8	GLOBALVARP . . . . .	21	MSDESCRIBE . . . . .	6	MSNOTICEFILE . . . . .	2	PARSERELATION1 . . . . .	12
BELOWMARKER . . . . .	27	LISTHARD . . . . .	35	MSDESCRIBE1 . . . . .	6	MSONPATH . . . . .	26	PRINTERERROR . . . . .	21
BUILDGETRELQ . . . . .	16	MAKEHASH . . . . .	14	MSEDITF . . . . .	2, 27	MSOUTPUT . . . . .	39	READATABASE . . . . .	17
BUILDTESTRELQ . . . . .	16	MAPRELATION . . . . .	13	MSERASE . . . . .	16	MSPATHS . . . . .	24	SETTEMPLATE . . . . .	8
CHANGEI.S. . . . .	29	MASTERSCOPE . . . . .	30	MSFIND . . . . .	27	MSPATHS1 . . . . .	25	SHORTLST . . . . .	23
CHANGEMACRO . . . . .	28	MASTERSCOPE1 . . . . .	30	MSGETBLOCKDEC . . . . .	35	MSPATHS2 . . . . .	25	SPECVARP . . . . .	23
CHANGERECD . . . . .	29	MASTERSCOPEEXEC . . . . .	30	MSGETDEF . . . . .	2	MSPATHS4 . . . . .	26	STORETABLE . . . . .	11
CHANGEVAR . . . . .	28	MSADDANALYZE . . . . .	9	MSHASHLIST . . . . .	36	MSPATHSPRINTFN . . . . .	27	SUBHASH . . . . .	14
CHECKFORCHANGED . . . . .	40	MSADDMODIFIER . . . . .	9	MSHASHLIST1 . . . . .	36	MSPRINTHELPPFILE . . . . .	7	TEMPLATE . . . . .	7
CHECKPATHS . . . . .	36	MSADDRELATION . . . . .	9	MSINIT . . . . .	10	MSREHASH . . . . .	14	TESTRELATION . . . . .	13
DASHES . . . . .	27	MSADDTYPE . . . . .	9	MSINTERPA . . . . .	35	MSSHOWUSE . . . . .	3	UNECCSPEC . . . . .	23
DOERROR . . . . .	23	MSCHECKBLOCK . . . . .	18	MSINTERPRET . . . . .	37	MSSOLVE . . . . .	40	UNSAVEFNS . . . . .	29
DOTABS . . . . .	27	MSCHECKBLOCKBASIC . . . . .	20	MSINTERPRETSET . . . . .	31	MSSTOREDATA . . . . .	11	UPDATECHANGED . . . . .	5
DUMPDATABASE . . . . .	17	MSCHECKBLOCKS . . . . .	18	MSLISTSET . . . . .	35	MSUPDATE . . . . .	4	UPDATECHANGED1 . . . . .	5
DUMPDATABASE1 . . . . .	17	MSCHECKBOUNDFREE . . . . .	21	MSMARKCHANGE1 . . . . .	10	MSUPDATEFN1 . . . . .	4	UPDATEFN . . . . .	1
EQMEMHASH . . . . .	14	MSCHECKEMPTY . . . . .	39	MSMARKCHANGED . . . . .	28	MSUSERVBTABLES . . . . .	15	VERBNOTICELIST . . . . .	39
FMAPRINT . . . . .	7	MSCHECKFNINBLOCK . . . . .	20	MSMEMBSET . . . . .	35	MSVBTABLES . . . . .	14		
GETRELATION . . . . .	13	MSCHECKVARS1 . . . . .	22	MSMSGPRINT . . . . .	24	NECCSPEC . . . . .	23		

VARIABLE INDEX

ANALYZEUSERFNS . . . . .	6	MSAUXCOMS . . . . .	6	MSDATABASELST . . . . .	11	MSPATHSCOMS . . . . .	24
CHECKUNSAVEFLG . . . . .	30	MSBLIP . . . . .	28	MSDBCOMS . . . . .	10	MSPRINTCNT . . . . .	6
COMPILE.TIME.CONSTANTS . . . . .	30	MSCHANGEDARRAY . . . . .	11	MSDBEMPTY . . . . .	11	MSPRINTFLG . . . . .	5
DATABASECOMS . . . . .	17	MSCHECKBLOCKSCOMS . . . . .	17	MSFILELST . . . . .	5	MSUSERVBTABLES . . . . .	16
DESCRIBELST . . . . .	7	MSCHECKFNS . . . . .	10	MSFNTPES . . . . .	28	NODUMPRELATIONS . . . . .	12
GAINSPACEFORMS . . . . .	17	MSCOMPILETIME . . . . .	40	MSHASHFILE . . . . .	6	RECORDCHANGEFN . . . . .	30
HISTORYCOMS . . . . .	30	MSCRATCHASH . . . . .	28	MSHELPPFILE . . . . .	7	TABLE.TO.NOTICED . . . . .	5
MASTERSCOPEDATE . . . . .	30	MSDATABASECOMS . . . . .	1	MSNEEDUNSAVE . . . . .	30		
MSANALYZEFNS . . . . .	16	MSDATABASEINIT . . . . .	11	MSOPENFILES . . . . .	5		

RECORD INDEX

GETHASH . . . . .	40	INRELATION . . . . .	40	MSANALYZABLE . . . . .	40	PATHOPTIONS . . . . .	40
-------------------	----	----------------------	----	------------------------	----	-----------------------	----

MACRO INDEX

GETRELQ . . . . .	16	GETWORDTYPE . . . . .	30	SCRATCHASH . . . . .	28	TESTRELQ . . . . .	16
-------------------	----	-----------------------	----	----------------------	----	--------------------	----

PROPERTY INDEX

MASTERSCOPE . . . . .	1
-----------------------	---