File created: 14-Jul-86 14:48:47 **{ERINYES}<LISPMANUAL>LISP>IMNAME.;2**

changes to: (VARS IMNAMECOMS)
(FNS IMNAME.UPDATE.SEND.INFO)

previous date: 30-Sep-85 13:02:01 {ERINYES}<LISPMANUAL>LISP>IMNAME.;1

Read Table: OLD-INTERLISP-FILE

Package: INTERLISP

Format: XCCS


```
(RPAQQ IMNAMECOMS
       ((FNS DELETE.UNDER.MENUS DISPLAY.UNDER.MENU GET.IM.NAME.LIST IMNAME IMNAME.UPDATE.HASHFILE
             IMNAME.UPDATE.REF#TOPROG IMNAME.UPDATE.REFS IMNAME.UPDATE.SEND.INFO INSPECT.IM MAKE.IM.INSPECTOR
             MOVE.UNDER.MENUS OPEN.IM.NAME.HASHFILE REDISPLAY.IM.NAME.MENU REDISPLAY.IM.REF.MENU
             REDISPLAY.IM.TYPE.MENU SELECT.IM.MENU.ITEM TEDIT.IM.FILE)
        [INITVARS (IM.NAME.MAX.DISPLAY 5)
               (IM.NAME.DEFAULT.HASHFILE NIL)
               (IM.NAME.HASHFILE.ABBREVS (QUOTE ((INTERLISP . {ERINYES}<LISPMANUAL>INTERLISP.IMNAMEHASH)
                                                 (LOOPS . {INDIGO}<LOOPS>MANUAL>LOOPS.IMNAMEHASH]
        (FILES HASH)
        (MACROS IMNAME.RESETSAVE.MOVD)))

(DEFINEQ

(DELETE.UNDER.MENUS
  [LAMBDA (TOP.MENU.OR.WINDOW)                                   (* mjs "27-OCT-83 10:53")
    (PROG (TOP.WINDOW UNDER.MENU UNDER.WINDOW)
          (if [AND (SETQ TOP.WINDOW (if (WINDOWP TOP.MENU.OR.WINDOW)
                                        else (WFROMMENU TOP.MENU.OR.WINDOW)))
                   (SETQ UNDER.MENU (WINDOWPROP TOP.WINDOW (QUOTE UNDER.MENU]
              then (WINDOWPROP TOP.WINDOW (QUOTE UNDER.MENU)
                          NIL)
                   (DELETE.UNDER.MENUS UNDER.MENU)
                   (DELETEMENU UNDER.MENU NIL (SETQ UNDER.WINDOW (WFROMMENU UNDER.MENU)))
                   (CLOSEW UNDER.WINDOW])

(DISPLAY.UNDER.MENU
  [LAMBDA (TOP.MENU NEW.MENU)                                    (* mjs "24-Jul-85 15:19")
    (PROG (NEW.WINDOW (TOP.WINDOW (WFROMMENU TOP.MENU)))
          (if TOP.WINDOW
              then (DELETE.UNDER.MENUS TOP.MENU)
                   (SETQ NEW.WINDOW (CREATEW (CREATEREGION (fetch (REGION LEFT) of (WINDOWPROP TOP.WINDOW
                                                                                         (QUOTE REGION)))
                                                           (IDIFFERENCE (fetch (REGION BOTTOM)
                                                                           of (WINDOWPROP TOP.WINDOW (QUOTE REGION)))
                                                                  (fetch (MENU IMAGEHEIGHT) of NEW.MENU))
                                                           (fetch (MENU IMAGEWIDTH) of NEW.MENU)
                                                           (fetch (MENU IMAGEHEIGHT) of NEW.MENU))
                                             NIL 0))
                   (WINDOWPROP NEW.WINDOW (QUOTE UNDER.MENU)
                          NIL)
                   (ADDMENU NEW.MENU NEW.WINDOW)
                   (WINDOWADDPROP NEW.WINDOW (QUOTE CLOSEFN)
                          (FUNCTION DELETE.UNDER.MENUS))
                   (WINDOWADDPROP NEW.WINDOW (QUOTE MOVEFN)
                          (FUNCTION MOVE.UNDER.MENUS))
                   (WINDOWPROP TOP.WINDOW (QUOTE UNDER.MENU)
                          NEW.MENU)
            else (MENU NEW.MENU])

(GET.IM.NAME.LIST
  [LAMBDA (HASHFILE.NAME)                                        (* mjs "16-Dec-83 14:34")
    (PROG (NAME.LIST HASHFILE.PTR)
          (SETQ HASHFILE.PTR (OPEN.IM.NAME.HASHFILE HASHFILE.NAME))
          (if (NULL HASHFILE.PTR)
              then (RETURN))
          (SETQ NAME.LIST (GETHASHFILE (QUOTE namelist)
                                 HASHFILE.PTR))
          (CLOSEHASHFILE HASHFILE.PTR)
          (RETURN NAME.LIST])

(IMNAME
  [LAMBDA (HASHFILE.NAME)                                        (* mjs "15-Jan-84 17:22")
    (MAKE.IM.INSPECTOR HASHFILE.NAME])

(IMNAME.UPDATE.HASHFILE
  [LAMBDA (OLD.HASHFILE.NAME ADD.FILES DELETE.FILES FLUSH.DISAPPEARED.FLG)
```

(* mjs "29-Sep-85 14:08")

(* * this function updates an IMNAME database hashfile. First, it looks at the list of files referenced in the hashfile <saved under the hash name "file/date/info" >, and determines which of these files have been updated, by searching for the files, and checking creation dates. Next, for every updated file, IMTRAN is called with several of its subprocedures modified so it will not output anything, and only put index information into an in-core hasharray.
Finally, the entries in the old hashfile are read in, merged with info from the in-core hasharray, and written out to a new hashfile.)

(* * OLD.HASHFILE.NAME is the name of the hashfile to be updated.
<Note that this file must be named explicitely --- no file searches are done, so that the user will not inadvertantly start updating the main manual database>. The new hashfile will be created as the new version of the same file name.
ADD.FILES is a list of files that will be analyzed, and added to the database.
DELETE.FILES is a list of files that will be deleted from the database.
ADD.FILES and DELETE.FILES can be used to "manage" a database, as new files are added to a document, and old ones are removed, split up,or renamed. FLUSH.DISAPPEARED.FLG determines what IMNAME.UPDATE.HASHFILE will do if it finds that some of the files in the database have disappeared <and they are not named on DELETE.FILES>.
If FLUSH.DISAPPEARED.FLG = T, the info for those files will simply be deleted.
If FLUSH.DISAPPEARED.FLG = ERROR, IMNAME.UPDATE.HASHFILE will return without doing anything if files have disappeared. If FLUSH.DISAPPEARED.FLG = <anything else>, the info on the disappeared files will simply be retained.)

(* * to create a new IMNAME hashfile, pass a non-existant file name as OLD.HASHFILE.NAME, and give a list of files as ADD.FILES. In this case, a new hashfile will be created just from the internal hasharray info.)

```
(PROG ((DISAPPEARED.IM.FILES NIL)
       (REDO.IM.FILE.NAMES NIL)
       (FLUSH.FILES NIL)
       (OLD.HASHFILE NIL)
       (OLD.IM.NAME.LIST NIL)
       (OLD.IM.FILE.INFO NIL)
       (OLD.IM.FILE.NAMES NIL)
       NEW.HASHFILE.NAME NEW.HASHFILE NEW.IM.NAME.LIST NEW.IM.FILE.NAMES NEW.IM.FILE.INFO NEW.IM.HASH
       DELETE.FILE.NAMES ERRFILE ERRFILE.NAME)
      (DECLARE (SPECVARS ERRFILE ERRFILE.NAME))        (* make sure that IMTRAN is loaded, so that we can analyze
                                                       updated files.)
      (FILESLOAD IMTRAN)                               (* U-CASE all file names, because we will be comparing them to
 database file names)
      (SETQ OLD.HASHFILE.NAME (U-CASE OLD.HASHFILE.NAME))
      (SETQ ADD.FILES (U-CASE ADD.FILES))
      (SETQ DELETE.FILES (U-CASE DELETE.FILES))
      (SETQ NEW.HASHFILE.NAME (PACKFILENAME (QUOTE VERSION)
                                            NIL
                                            (QUOTE BODY)
                                            OLD.HASHFILE.NAME))
      (SETQ ERRFILE.NAME (PACKFILENAME (QUOTE EXTENSION)
                                       (QUOTE IMERR)
                                       (QUOTE BODY)
                                       NEW.HASHFILE.NAME))
      [if (INFILEP OLD.HASHFILE.NAME)
          then                                         (* if old hashfile exists, open it and get namelist and filelist)
               (IM.WARNING "Opening old hashfile " OLD.HASHFILE.NAME)
               (SETQ OLD.HASHFILE (OPENHASHFILE OLD.HASHFILE.NAME (QUOTE INPUT)))
               (SETQ OLD.IM.NAME.LIST (GETHASHFILE (QUOTE namelist)
                                                   OLD.HASHFILE))
               (SETQ OLD.IM.FILE.INFO (GETHASHFILE (QUOTE fileinfo)
                                                   OLD.HASHFILE))
               (SETQ OLD.IM.FILE.NAMES (for X in OLD.IM.FILE.INFO collect (CAR X]
      (if (NULL OLD.HASHFILE)
          then (IM.WARNING "Will construct new hashfile named: " NEW.HASHFILE.NAME))

      (* push on REDO.IM.FILE.NAMES the full names of all files on ADD.FILES that can be found.)

      (for FILE in ADD.FILES bind COMPLETEFILENAME do (SETQ COMPLETEFILENAME (FINDFILE FILE T))
                                                      (if COMPLETEFILENAME
                                                          then [SETQ FILE (PACKFILENAME (QUOTE NAME)
                                                                                        (FILENAMEFIELD FILE
                                                                                               (QUOTE NAME))
                                                                                        (QUOTE EXTENSION)
                                                                                        (FILENAMEFIELD FILE
                                                                                               (QUOTE EXTENSION]
                                                               (push REDO.IM.FILE.NAMES FILE)
                                                               (IM.WARNING "Adding file " COMPLETEFILENAME)
                                                               (if (MEMB FILE OLD.IM.FILE.NAMES)
                                                                   then (IM.WARNING "(updating version of file
                                                                                    in old database)"))
                                                        else (IM.WARNING "Can't find file " FILE " --
                                                                       ignored")))
                                                               (* collect the normal names of all deleted files, minus version
                                                               numbers)
      [SETQ DELETE.FILE.NAMES (for FILE in DELETE.FILES collect (PACKFILENAME (QUOTE NAME)
                                                                              (FILENAMEFIELD FILE (QUOTE NAME))
                                                                              (QUOTE EXTENSION)
                                                                              (FILENAMEFIELD FILE (QUOTE EXTENSION]
```

(* analyze all of the files referenced in the old hashfile. There are four cases: <1> if the file is on DELETE.FILE.NAMES, all references to it should be flushed; otherwise <2> if the file does not exist, put it on DISAPPEARED.IM.FILES;
otherwise <3> if the file DOES exist, but it has a different version number, flush the old version of the file, and reanalyze the new version; otherwise <4> the current version of the file is correct, so you don't have to do anything.)

```
(for FILE in OLD.IM.FILE.NAMES bind STANDARD.FILE.NAME LATEST.VERSION.FILE
   do (if (MEMB FILE DELETE.FILE.NAMES)
          then                                       (* if the file is on DELETE.FILE.NAMES, just flush it)
               (IM.WARNING "will delete info for: " FILE)
               (push FLUSH.FILES FILE)
        elseif (MEMB FILE REDO.IM.FILE.NAMES)
          then

(* if an old file is already on the list to redo, flush it immediately.
It must have been an added file)

               (push FLUSH.FILES FILE)
        elseif (NULL (SETQ LATEST.VERSION.FILE (FINDFILE FILE T)))
          then                                       (* if this file does not exist, put it on DISAPPEARED.IM.FILES)
               (IM.WARNING "can't find old file " FILE)
               (push DISAPPEARED.IM.FILES FILE)
        elseif [NOT (EQUAL (GETFILEINFO LATEST.VERSION.FILE (QUOTE CREATIONDATE))
                           (CDR (ASSOC FILE OLD.IM.FILE.INFO]
          then

(* if this file DOES exist, but it has a different creationdate, flush the old version of the file, and reanalyze the new version)

               (IM.WARNING "old file " FILE " has been updated -- will re-analyze" "  [author="
                      (GETFILEINFO LATEST.VERSION.FILE (QUOTE AUTHOR))
                      " , date="
                      (GETFILEINFO LATEST.VERSION.FILE (QUOTE CREATIONDATE))
                      "]")
               (push FLUSH.FILES FILE)
               (push REDO.IM.FILE.NAMES FILE)))

(* if any files referenced in the old hashfile have disappeared, take different actions depending on
FLUSH.DISAPPEARED.FLG: If FLUSH.DISAPPEARED.FLG = T, just flush the file info.
If FLUSH.DISAPPEARED.FLG = ERROR, close the hashfile and stop the program.
Otherwise, just leave the disappeared file info intact.)

(if DISAPPEARED.IM.FILES
    then (IM.WARNING "the following files have disappeared: " DISAPPEARED.IM.FILES)
         (SELECTQ FLUSH.DISAPPEARED.FLG
              (T (IM.WARNING "Will delete info for disappeared files")
                 (SETQ FLUSH.FILES (APPEND DISAPPEARED.IM.FILES FLUSH.FILES)))
              (ERROR (IM.WARNING "--- returning ---")
                     (CLOSEHASHFILE OLD.HASHFILE)
                     (if (OPENP ERRFILE)
                         then (CLOSEF ERRFILE)
                              (IM.WARNING T "IMTRAN Error File: " (FULLNAME ERRFILE)
                                     T))
                     (RETURN))
              (IM.WARNING "Will keep info for disappeared files")))
                                                     (* initialize new file list and name list, and in-core hasharray for
                                                     re-analyzed file info)
(SETQ NEW.IM.FILE.NAMES (LDIFFERENCE OLD.IM.FILE.NAMES FLUSH.FILES))
(SETQ NEW.IM.FILE.INFO (for X in OLD.IM.FILE.INFO when (MEMB (CAR X)
                                                             NEW.IM.FILE.NAMES)
                          collect X))
(SETQ NEW.IM.NAME.LIST NIL)
(SETQ NEW.IM.HASH (HASHARRAY 2000))

(* * analyze updated IM files, by running each one through IMTRAN)

(RESETLST                                            (* make sure that IMTRAN does not dump anything, or include
                                                     any files.)
    (IMNAME.RESETSAVE.MOVD (FUNCTION NILL)
           (FUNCTION DUMP))
    (IMNAME.RESETSAVE.MOVD (FUNCTION NILL)
           (FUNCTION INCLUDE.FILE))                  (* use modified SEND.INFO program that will dump info in
                                                     in-core hasharray)
    (IMNAME.RESETSAVE.MOVD (FUNCTION IMNAME.UPDATE.SEND.INFO)
           (FUNCTION SEND.INFO))
    (IMNAME.RESETSAVE.MOVD (FUNCTION IMNAME.UPDATE.REF#TOPROG)
           (FUNCTION REF#TOPROG))
    (PROG ((IMNAME.UPDATE.SEND.INFO.HASH NEW.IM.HASH)
           (IMNAME.UPDATE.SEND.INFO.NEW.WORDS NIL)
           IMNAME.UPDATE.SEND.INFO.FILENAME)
          (DECLARE (SPECVARS UPDATE.SEND.INFO.HASH UPDATE.SEND.INFO.FILENAME))

(* the single file ERRFILE is used to save error messages from all invokations of IMTRAN.
UPDATE.SEND.INFO.HASH and UPDATE.SEND.INFO.FILENAME are SPECVARS used to communicate with the special
version of SEND.INFO which puts index info in the in-core hash array)

          (for FILE in REDO.IM.FILE.NAMES bind COMPLETE.FILE.NAME
             do (SETQ COMPLETE.FILE.NAME (FINDFILE FILE T))
                (if (NULL COMPLETE.FILE.NAME)
                    then (SHOULDNT "Could find file before, but not now"))
                (IM.WARNING "Retranslating file: " COMPLETE.FILE.NAME)
                (push NEW.IM.FILE.NAMES FILE)
                [push NEW.IM.FILE.INFO (CONS FILE (GETFILEINFO FILE (QUOTE CREATIONDATE]
```

```
                                                                  (* UPDATE.SEND.INFO.FILENAME is the file name in a
                                                                  standard format <name and ext only>)
                       (SETQ IMNAME.UPDATE.SEND.INFO.FILENAME FILE)
                       (PROG ((GLOBAL.CHAPTER.NUMBER 0)
                              (IM.NOTE.FLG NIL)
                              (IM.REF.FLG NIL)
                              (IM.INDEX.FILE.FLG T))
                             (DECLARE (SPECVARS GLOBAL.CHAPTER.NUMBER IM.NOTE.FLG IM.REF.FLG
                                             IM.INDEX.FILE.FLG))
                             (IMTRAN COMPLETE.FILE.NAME)))   (* set new word list to all words collected while reanalyzing files)
                  (SETQ NEW.IM.NAME.LIST IMNAME.UPDATE.SEND.INFO.NEW.WORDS)))
         (SETQ NEW.IM.NAME.LIST (UNION NEW.IM.NAME.LIST OLD.IM.NAME.LIST))
         [SETQ NEW.HASHFILE (CREATEHASHFILE NEW.HASHFILE.NAME NIL NIL (TIMES 1.3 (LENGTH NEW.IM.NAME.LIST]
         (for NAM in NEW.IM.NAME.LIST bind NEW.REFS (FLUSH.IM.NAMES _ NIL)
            do (if (SETQ NEW.REFS (IMNAME.UPDATE.REFS (if OLD.HASHFILE
                                                          then (GETHASHFILE NAM OLD.HASHFILE)
                                                        else NIL)
                                    (GETHASH NAM NEW.IM.HASH)
                                    FLUSH.FILES))
                   then (PUTHASHFILE NAM NEW.REFS NEW.HASHFILE)
                 else (push FLUSH.IM.NAMES NAM))
            finally (SETQ NEW.IM.NAME.LIST (LDIFFERENCE NEW.IM.NAME.LIST FLUSH.IM.NAMES)))
         (PUTHASHFILE (QUOTE namelist)
                NEW.IM.NAME.LIST NEW.HASHFILE)
         (PUTHASHFILE (QUOTE fileinfo)
                NEW.IM.FILE.INFO NEW.HASHFILE)
         (if OLD.HASHFILE
             then (CLOSEHASHFILE OLD.HASHFILE))
         (CLOSEHASHFILE NEW.HASHFILE)
         (if (OPENP ERRFILE)
             then (IM.WARNING "IMTRAN Error File: " (FULLNAME ERRFILE))
                  (CLOSEF ERRFILE))
         (RETURN NEW.HASHFILE.NAME])
```

(IMNAME.UPDATE.REF#TOPROG

```
  [LAMBDA NIL                                                     (* mjs "30-Sep-85 13:01")
    (PROG (FILEPTR SAV REF.STRING TYPE ARGS TEMP NAME TYPE.AS.STRING INFO.WORD NEW.HASH.INFO)
          (SETQ FILEPTR (GETFILEPTR IM.INFILE))
          (SETQ SAV (SAVE.ARG))
          (SETQ TEMP (PARSE.INDEX.SPEC SAV NIL))
          (if (OR (NULL TEMP)
                  (NULL (CAR TEMP)))
              then (IM.WARNING "null index --- ignored")
                   (RETURN))
          (SETQ ARGS (CAR TEMP))
          [SETQ TYPE (if (EQ TO.NAME (QUOTE FIGUREREF))
                         then                                     (* for FIGUREREF, ignore specified type ---
                                                                  use TAG)
                              (QUOTE TAG)
                       else (U-CASE (CDR TEMP]
          [SETQ TYPE.AS.STRING (if (NLISTP TYPE)
                                   then (MKSTRING (L-CASE TYPE T))
                                 else (LIST.TO.STRING (L-CASE TYPE T]
          [SETQ NAME (U-CASE (MKATOM (LIST.TO.STRING ARGS]
          (SETQ INFO.WORD (L-CASE TO.NAME T))
          (SETQ NEW.HASH.INFO (GETHASH NAME IMNAME.UPDATE.SEND.INFO.HASH))
          (if (NULL NEW.HASH.INFO)
              then (push IMNAME.UPDATE.SEND.INFO.NEW.WORDS NAME))
          (PUTHASH NAME (CONS (LIST TYPE.AS.STRING (LIST IMNAME.UPDATE.SEND.INFO.FILENAME INFO.WORD FILEPTR))
                              NEW.HASH.INFO)
                 IMNAME.UPDATE.SEND.INFO.HASH])
```

(IMNAME.UPDATE.REFS

```
  [LAMBDA (OLD.REFS NEW.REFS FLUSH.FILES)                         (* mjs "15-Jan-84 17:18")

          (* merge the refs in OLD.REFS with the refs in NEW.REFS, flushing any references to files on FLUSH.FILES)

    (PROG [(NEW.REF NIL)
           (TYPES (for X in OLD.REFS collect (CAR X]
          (for X in NEW.REFS unless (MEMBER (CAR X)
                                            TYPES)
             do (push TYPES (CAR X)))

          (* * now, TYPES contains a list of all of the types in both the old and new refs)

          [for TYPE in TYPES bind OLD.FILEREFS NEW.FILEREFS
             do                                                   (* first, collect file refs from OLD.REFS, flushing files on
                                                                  FLUSH.FILES)
                (SETQ OLD.FILEREFS (for X in (CDR (SASSOC TYPE OLD.REFS)) unless (MEMB (CAR X)
                                                                                       FLUSH.FILES)
                                      collect X))

          (* next, collect all file refs in NEW.REFS for this type. Note that each ref in NEW.REFS contains exactly one file ref)
                                                                  (* implicite assumption: the files in OLD.REFS and NEW.REFS
                are completely disjoint)
```

```
            (SETQ NEW.FILEREFS (for X in NEW.REFS when (EQUAL TYPE (CAR X)) collect (CADR X)))
            [SETQ NEW.FILEREFS (for FILEREFS in (PARTITION.LIST NEW.FILEREFS NIL (FUNCTION CAR))
                                   collect                   (* FILEREFS is a list of the filerefs for a single file)
                                                             (* sort FILEREFS by file pointers)
                                         [SORT FILEREFS (FUNCTION (LAMBDA (A B)
                                                                    (ILESSP (CADDR A)
                                                                           (CADDR B]
                                                             (* put all of the file refs in one list, headed by the file name)
                                         (CONS (CAAR FILEREFS)
                                               (for X in FILEREFS join (CDR X]
            (SETQ NEW.FILEREFS (SORT (NCONC NEW.FILEREFS OLD.FILEREFS)
                                     T))
            (if NEW.FILEREFS
                then (SETQ NEW.REF (CONS (CONS TYPE NEW.FILEREFS)
                                         NEW.REF]    (* finally, sort all of the references by type)
        (RETURN (SORT NEW.REF T])
```

(IMNAME.UPDATE.SEND.INFO

```
  [LAMBDA (NAME TYPE SAV INFO PLIST)                          (* mjs "14-Jul-86 14:42")

        (* * substitute version of SEND.INFO that puts index info into IMNAME.UPDATE.SEND.INFO.HASH instead of spitting out
        an index object.)

    (PROG ((FILEPTR (GETFILEPTR IM.INFILE))
           (INFO.WORD (if (MEMB (QUOTE *PRIMARY*)
                                INFO)
                          then (QUOTE Primary)
                        elseif (MEMB (QUOTE *DEF*)
                                     INFO)
                          then (QUOTE Definition)
                        else NIL))
           [TYPE.AS.STRING (if (NLISTP TYPE)
                               then (MKSTRING (L-CASE TYPE T))
                             else (LIST.TO.STRING (L-CASE TYPE T]
           (NEW.HASH.INFO (GETHASH NAME IMNAME.UPDATE.SEND.INFO.HASH)))

        (* * pack index subentries and subsubentries)

        (if (LISTGET PLIST (QUOTE SUBNAME))
            then (if (LISTGET PLIST (QUOTE SUBSUBNAME))
                     then (SETQ INFO.WORD (PACK* " -> " (LISTGET PLIST (QUOTE SUBSUBNAME))
                                                 "/"
                                                 (LISTGET PLIST (QUOTE SUBSUBTYPE))
                                                 "/" INFO.WORD)))
                 (SETQ INFO.WORD (PACK* "sub -> " (LISTGET PLIST (QUOTE SUBNAME))
                                        "/"
                                        (LISTGET PLIST (QUOTE SUBTYPE))
                                        "/" INFO.WORD)))
        (if (MEMB (QUOTE *BEGIN*)
                  INFO)
            then (SETQ INFO.WORD (PACK* INFO.WORD "/begin")))
        (if (MEMB (QUOTE *END*)
                  INFO)
            then (SETQ INFO.WORD (PACK* INFO.WORD "/end")))
        (if (NULL NEW.HASH.INFO)
            then (push IMNAME.UPDATE.SEND.INFO.NEW.WORDS NAME))
        (PUTHASH NAME (CONS (LIST TYPE.AS.STRING (LIST IMNAME.UPDATE.SEND.INFO.FILENAME INFO.WORD FILEPTR))
                            NEW.HASH.INFO)
               IMNAME.UPDATE.SEND.INFO.HASH])
```

(INSPECT.IM

```
  [LAMBDA (NAM HASHFILE.NAME)                                 (* mjs "27-OCT-83 18:54")
    (PROG (REFS HASHFILE.PTR HASHFILE.DEFAULT.DIRECTORY TYP FILE.POS.PTR)
          (SETQ HASHFILE.PTR (OPEN.IM.NAME.HASHFILE HASHFILE.NAME))
          (if (NULL HASHFILE.PTR)
              then (RETURN))
          [SETQ HASHFILE.DEFAULT.DIRECTORY (PACKFILENAME (QUOTE HOST)
                                                  (FILENAMEFIELD (HASHFILENAME HASHFILE.PTR)
                                                         (QUOTE HOST))
                                                  (QUOTE DIRECTORY)
                                                  (FILENAMEFIELD (HASHFILENAME HASHFILE.PTR)
                                                         (QUOTE DIRECTORY]
          (SETQ REFS (GETHASHFILE (U-CASE NAM)
                            HASHFILE.PTR))
          (CLOSEHASHFILE HASHFILE.PTR)
          (if (NULL REFS)
              then (CLRPROMPT)
                   (PROMPTPRINT (CONCAT NAM " has no references"))
                   (RETURN))
          (REDISPLAY.IM.TYPE.MENU NIL (LIST NAM HASHFILE.DEFAULT.DIRECTORY REFS])
```

(MAKE.IM.INSPECTOR

```
  [LAMBDA (HASHFILE.NAME MENU.REGION)                         (* mjs "24-Jul-85 16:55")
    (PROG (HASHFILE.PTR HASHFILE.WINDOW.STRING HASHFILE.DEFAULT.DIRECTORY WINDOW MENU)
```

```
        (SETQ HASHFILE.PTR (OPEN.IM.NAME.HASHFILE HASHFILE.NAME))
        (if (NULL HASHFILE.PTR)
            then (RETURN))
        [SETQ HASHFILE.WINDOW.STRING (U-CASE (FILENAMEFIELD (HASHFILENAME HASHFILE.PTR)
                                                  (QUOTE NAME]
        [SETQ HASHFILE.DEFAULT.DIRECTORY (PACKFILENAME (QUOTE HOST)
                                              (FILENAMEFIELD (HASHFILENAME HASHFILE.PTR)
                                                     (QUOTE HOST))
                                              (QUOTE DIRECTORY)
                                              (FILENAMEFIELD (HASHFILENAME HASHFILE.PTR)
                                                     (QUOTE DIRECTORY]
        (SETQ WINDOW (CREATEW (if MENU.REGION
                                  else (GETBOXREGION 106 37 NIL NIL NIL (CONCAT "Please position the IM Name
                                                                                Inspector Window for "
                                                                                HASHFILE.WINDOW.STRING)))
                              NIL 0))
        (WINDOWPROP WINDOW (QUOTE UNDER.MENU)
               NIL)
        (WINDOWPROP WINDOW (QUOTE IM.NAME.ASSOC)
               NIL)
        (WINDOWPROP WINDOW (QUOTE IM.NAME.HASHFILE)
               HASHFILE.PTR)
        (WINDOWPROP WINDOW (QUOTE IM.NAME.HASHFILE.WINDOW.STRING)
               HASHFILE.WINDOW.STRING)
        (WINDOWPROP WINDOW (QUOTE IM.NAME.DEFAULT.DIRECTORY)
               HASHFILE.DEFAULT.DIRECTORY)
        [WINDOWADDPROP WINDOW (QUOTE CLOSEFN)
               (FUNCTION (LAMBDA (WINDOW)
                           (CLOSEHASHFILE (WINDOWPROP WINDOW (QUOTE IM.NAME.HASHFILE]
        (WINDOWADDPROP WINDOW (QUOTE CLOSEFN)
               (FUNCTION DELETE.UNDER.MENUS))
        (WINDOWADDPROP WINDOW (QUOTE MOVEFN)
               (FUNCTION MOVE.UNDER.MENUS))
        (SETQ MENU (create MENU))
        (ADDMENU MENU WINDOW)
        (REDISPLAY.IM.NAME.MENU MENU])
```

(MOVE.UNDER.MENUS

```
  [LAMBDA (TOP.MENU.OR.WINDOW NEW.POS)                          (* mjs "28-OCT-83 13:21")
    (PROG (TOP.WINDOW UNDER.MENU UNDER.WINDOW UNDER.WINDOW.NEW.POS)
          (COND
             ([AND (SETQ TOP.WINDOW (if (WINDOWP TOP.MENU.OR.WINDOW)
                                        else (WFROMMENU TOP.MENU.OR.WINDOW)))
                   (SETQ UNDER.MENU (WINDOWPROP TOP.WINDOW (QUOTE UNDER.MENU]
              (SETQ UNDER.WINDOW (WFROMMENU UNDER.MENU))
              [SETQ UNDER.WINDOW.NEW.POS (create POSITION
                                             XCOORD _ (fetch (POSITION XCOORD) of NEW.POS)
                                             YCOORD _ (IDIFFERENCE (fetch (POSITION YCOORD) of NEW.POS)
                                                             (fetch (REGION HEIGHT) of (WINDOWPROP UNDER.WINDOW
                                                                                              (QUOTE REGION]
              (MOVEW UNDER.WINDOW UNDER.WINDOW.NEW.POS])
```

(OPEN.IM.NAME.HASHFILE

```
  [LAMBDA (HASHFILE.NAME)                                       (* edited: "11-Jul-84 14:51")
    (PROG ([DEFAULT.HASHFILE.NAME (if IM.NAME.DEFAULT.HASHFILE
                                      elseif (EQ (FILENAMEFIELD LOGINHOST/DIR (QUOTE HOST))
                                                 (QUOTE IVY))
                                        then (CDR (ASSOC (QUOTE LOOPS)
                                                         IM.NAME.HASHFILE.ABBREVS))
                                      else (CDR (ASSOC (QUOTE INTERLISP)
                                                       IM.NAME.HASHFILE.ABBREVS]
           (ABBREV.HASHFILE.NAME (CDR (ASSOC HASHFILE.NAME IM.NAME.HASHFILE.ABBREVS)))
           FULL.HASHFILE.NAME)
          [SETQ FULL.HASHFILE.NAME (if ABBREV.HASHFILE.NAME
                                       elseif (NULL HASHFILE.NAME)
                                         then (INFILEP DEFAULT.HASHFILE.NAME)
                                       elseif (FINDFILE HASHFILE.NAME T)
                                       elseif (FINDFILE (PACKFILENAME (QUOTE BODY)
                                                               HASHFILE.NAME
                                                               (QUOTE EXTENSION)
                                                               (FILENAMEFIELD DEFAULT.HASHFILE.NAME (QUOTE EXTENSION
                                                                                                        )))
                                                        T)
                                       else (INFILEP (PACKFILENAME (QUOTE BODY)
                                                            HASHFILE.NAME
                                                            (QUOTE BODY)
                                                            DEFAULT.HASHFILE.NAME]
          (if FULL.HASHFILE.NAME
              then (printout T "opening data base file " FULL.HASHFILE.NAME T)
                   (RETURN (OPENHASHFILE FULL.HASHFILE.NAME (QUOTE INPUT)))
            else (printout T "data base file " HASHFILE.NAME " not found" T)
                 (RETURN NIL])
```

(REDISPLAY.IM.NAME.MENU

```
  [LAMBDA (OLDMENU)                                                   (* mjs " 6-Aug-85 14:10")
                                                                      (* updates IM name menu OLDMENU.)
    (PROG ((WINDOW (WFROMMENU OLDMENU))
           NAME.ASSOC HASHFILE.PTR HASHFILE.WINDOW.STRING HASHFILE.DEFAULT.DIRECTORY MENU)
          (SETQ NAME.ASSOC (WINDOWPROP WINDOW (QUOTE IM.NAME.ASSOC)))
          (SETQ HASHFILE.PTR (WINDOWPROP WINDOW (QUOTE IM.NAME.HASHFILE)))
          (SETQ HASHFILE.WINDOW.STRING (WINDOWPROP WINDOW (QUOTE IM.NAME.HASHFILE.WINDOW.STRING)))
          (SETQ HASHFILE.DEFAULT.DIRECTORY (WINDOWPROP WINDOW (QUOTE IM.NAME.DEFAULT.DIRECTORY)))
          [SETQ MENU
           (create MENU
                  ITEMS _ [PROG [(MENU.ITEMS (for X in NAME.ASSOC as C from 1 to IM.NAME.MAX.DISPLAY
                                                collect (LIST (CAR X)
                                                              (CAR X)
                                                              "Reselect an old IM name"]
                                (RETURN (CONS HASHFILE.WINDOW.STRING (CONS (QUOTE (Type% an% IM% name
                                                                                   Type% an% IM% name
                                                                                   "The user is prompted
                                                                                   to type in a new IM
                                                                                   name"))
                                                                           MENU.ITEMS]
                  TITLE _ "IM Name Inspector"
                  MENUBORDERSIZE _ 1
                  WHENSELECTEDFN _
                  (FUNCTION (LAMBDA (ITEM MENU MOUSEKEY)
                              (if (LISTP ITEM)
                                  then (SELECT.IM.MENU.ITEM (if (EQ (CADR ITEM)
                                                                    (QUOTE Type% an% IM% name))
                                                                then (CLRPROMPT)
                                                                     [MKATOM (U-CASE (PROMPTFORWORD
                                                                                      "Type an IM name: " NIL
                                                                                      "Type a name to be looked
                                                                                      up in the Interlisp Manual
                                                                                      Index" PROMPTWINDOW NIL NIL
                                                                                      (CHARCODE (EOL ESCAPE LF]
                                                              else (CADR ITEM))
                                                            MENU]
          (DELETE.UNDER.MENUS OLDMENU)
          (DELETEMENU OLDMENU NIL WINDOW)
          (SHAPEW WINDOW (CREATEREGION (fetch (REGION LEFT) of (WINDOWPROP WINDOW (QUOTE REGION)))
                                (fetch (REGION BOTTOM) of (WINDOWPROP WINDOW (QUOTE REGION)))
                                (fetch (MENU IMAGEWIDTH) of MENU)
                                (fetch (MENU IMAGEHEIGHT) of MENU)))
          (ADDMENU MENU WINDOW)
          (SHADEITEM HASHFILE.WINDOW.STRING MENU BLACKSHADE WINDOW)
          (RETURN MENU])

(REDISPLAY.IM.REF.MENU
  [LAMBDA (NAME.OR.TYPE.MENU TYPE.NAME.DIR.REFS)                      (* mjs "28-OCT-83 11:42")
                                                                      (* TYPE.NAME.DIR.REFS is a list of <selected-type
                                                                      selected-name default-directory refs>)
    (PROG ((SELECTED.TYPE (CAR TYPE.NAME.DIR.REFS))
           (SELECTED.NAME (CADR TYPE.NAME.DIR.REFS))
           (DEFAULT.DIR (CADDR TYPE.NAME.DIR.REFS))
           (REFS (CADDDR TYPE.NAME.DIR.REFS)))
          (DISPLAY.UNDER.MENU NAME.OR.TYPE.MENU
                 (create MENU
                        ITEMS _ [for PTRS.TO.ONE.FILE in (CDR (ASSOC SELECTED.TYPE REFS))
                                   join (CONS (LIST (CONCAT "from: " (CAR PTRS.TO.ONE.FILE))
                                                    (CONS (PACKFILENAME (QUOTE BODY)
                                                                 (CAR PTRS.TO.ONE.FILE)
                                                                 (QUOTE BODY)
                                                                 DEFAULT.DIR)
                                                          NIL)
                                                    "creates/finds TEDIT window into the file")
                                              (for PTR on (CDR PTRS.TO.ONE.FILE) by (CDDR PTR)
                                                 collect (LIST (CONCAT (if (CAR PTR)
                                                                           else "index")
                                                                       "   ("
                                                                       (CADR PTR)
                                                                       ")")
                                                               (CONS (PACKFILENAME (QUOTE BODY)
                                                                            (CAR PTRS.TO.ONE.FILE)
                                                                            (QUOTE BODY)
                                                                            DEFAULT.DIR)
                                                                     (CADR PTR))
                                                               "creates/finds TEDIT window into the file, and
                                                               positions the cursor at the selected reference"]
                        TITLE _ (CONCAT "Refs for " SELECTED.NAME " (type " SELECTED.TYPE ")")
                        MENUBORDERSIZE _ 1
                        WHENSELECTEDFN _ (FUNCTION (LAMBDA (ITEM MENU MOUSEKEY)
                                                     (if (NLISTP ITEM)
                                                         then NIL
                                                       else (TEDIT.IM.FILE (CAR (CADR ITEM))
                                                                   (CDR (CADR ITEM])
```

```
(REDISPLAY.IM.TYPE.MENU
  [LAMBDA (NAME.MENU NAME.DIR.REFS)                                    (* mjs "27-OCT-83 17:04")
                                                                       (* NAME.DIR.REFS is a list of <selected-name default-directory
    refs>)
    (PROG ((NAME.WINDOW (WFROMMENU NAME.MENU))
           (SELECTED.NAME (CAR NAME.DIR.REFS))
           (REFS (CADDR NAME.DIR.REFS)))
          (if (EQLENGTH REFS 1)
              then                                                     (* if only one type, skip type menu)
                   (REDISPLAY.IM.REF.MENU NAME.MENU (CONS (CAAR REFS)
                                                          NAME.DIR.REFS))
            else (DISPLAY.UNDER.MENU NAME.MENU (create MENU
                                                   ITEMS _ (for X in REFS collect (LIST (CAR X)
                                                                                        NAME.DIR.REFS "Select
                                                                                        which type you want the
                                                                                        references for"))
                                                   TITLE _ (CONCAT "ref types for '" SELECTED.NAME "'")
                                                   MENUBORDERSIZE _ 1
                                                   WHENSELECTEDFN _
                                                   (FUNCTION (LAMBDA (ITEM MENU MOUSEKEY)
                                                               (if (NLISTP ITEM)
                                                                   then NIL
                                                                 else (REDISPLAY.IM.REF.MENU
                                                                        MENU
                                                                        (CONS (CAR ITEM)
                                                                              (CADR ITEM])

(SELECT.IM.MENU.ITEM
  [LAMBDA (NAM MENU)                                                   (* mjs "24-Jul-85 15:19")
    (PROG (NAME.ASSOC WINDOW NAM.DATA REFS HASHFILE.PTR)
          (SETQ WINDOW (WFROMMENU MENU))
          (SETQ NAME.ASSOC (WINDOWPROP WINDOW (QUOTE IM.NAME.ASSOC)))
          (SETQ HASHFILE.PTR (WINDOWPROP WINDOW (QUOTE IM.NAME.HASHFILE)))
          (if (EQ NAM (CAR (CAR NAME.ASSOC)))
              then                                                     (* selected first item, so don't need to do any updating)
                   (RETURN))
          (SETQ NAM.DATA (ASSOC NAM NAME.ASSOC))
          (if NAM.DATA
              then (SETQ REFS (CDR NAM.DATA))
                   (SETQ NAME.ASSOC (CONS NAM.DATA (REMOVE NAM.DATA NAME.ASSOC)))
            else (SETQ REFS (GETHASHFILE (U-CASE NAM)
                                         HASHFILE.PTR))
                 (if REFS
                     then (SETQ NAME.ASSOC (CONS (CONS NAM REFS)
                                                 NAME.ASSOC))
                   else (CLRPROMPT)
                        (PROMPTPRINT (CONCAT NAM " has no references"))
                        (RETURN)))
          (WINDOWPROP WINDOW (QUOTE IM.NAME.ASSOC)
                 NAME.ASSOC)
          (SETQ MENU (REDISPLAY.IM.NAME.MENU MENU))
          (REDISPLAY.IM.TYPE.MENU MENU (LIST NAM (WINDOWPROP WINDOW (QUOTE IM.NAME.DEFAULT.DIRECTORY))
                                             REFS])

(TEDIT.IM.FILE
  [LAMBDA (IM.FILE.NAME IM.FILE.PTR)                                   (* mjs "24-Jul-85 15:26")
    (PROG [(TEDIT.TEXT.OBJECT NIL)
           (NORMAL.FILE.NAME (PACKFILENAME (QUOTE NAME)
                                           (FILENAMEFIELD IM.FILE.NAME (QUOTE NAME))
                                           (QUOTE EXTENSION)
                                           (FILENAMEFIELD IM.FILE.NAME (QUOTE EXTENSION]
          (if (NULL IM.FILE.NAME)
              then (RETURN))
          [for X in (OPENWINDOWS) bind POSS.TOBJ POSS.FILENAME when (SETQ POSS.TOBJ (WINDOWPROP X (QUOTE TEXTOBJ))
                                                                          )
             repeatuntil TEDIT.TEXT.OBJECT do (SETQ POSS.FILENAME (FULLNAME (fetch (TEXTOBJ TXTFILE) of POSS.TOBJ)))
                                              (COND
                                                 ([OR (NOT (LITATOM POSS.FILENAME))
                                                      (NEQ (FILENAMEFIELD POSS.FILENAME (QUOTE NAME))
                                                           (FILENAMEFIELD NORMAL.FILE.NAME (QUOTE NAME)))
                                                      (NEQ (FILENAMEFIELD POSS.FILENAME (QUOTE EXTENSION))
                                                           (FILENAMEFIELD NORMAL.FILE.NAME (QUOTE EXTENSION]
                                                  (SETQ TEDIT.TEXT.OBJECT NIL))
                                                 (T (SETQ TEDIT.TEXT.OBJECT POSS.TOBJ]
          (if TEDIT.TEXT.OBJECT
              then (if IM.FILE.PTR
                       then (TEDIT.SETSEL TEDIT.TEXT.OBJECT (IMAX 1 (IDIFFERENCE (ADD1 IM.FILE.PTR)
                                                                                25))
                                   0
                                   (QUOTE LEFT))
                            (TEDIT.NORMALIZECARET TEDIT.TEXT.OBJECT)
                            (TEDIT.SETSEL TEDIT.TEXT.OBJECT (ADD1 IM.FILE.PTR)
                                   0
                                   (QUOTE LEFT))
                            (TEDIT.NORMALIZECARET TEDIT.TEXT.OBJECT))
```

```
                (TTY.PROCESS (WINDOWPROP (CAR (fetch (TEXTOBJ \WINDOW) of TEDIT.TEXT.OBJECT))
                                         (QUOTE PROCESS)))
          else (PROG [(FULL.FILE.NAME (if (FINDFILE NORMAL.FILE.NAME T)
                                          else (INFILEP IM.FILE.NAME]
                      (if (NULL FULL.FILE.NAME)
                          then (CLRPROMPT)
                               (printout PROMPTWINDOW NORMAL.FILE.NAME " not found" T)
                               (RETURN))
                      (CLRPROMPT)
                      (printout PROMPTWINDOW "Please specify a TEDIT window for " FULL.FILE.NAME T)
                      (TEDIT FULL.FILE.NAME NIL NIL (if IM.FILE.PTR
                                                        then (LIST (QUOTE SEL)
                                                                   (ADD1 IM.FILE.PTR))
                                                      else NIL])
)

(RPAQ? IM.NAME.MAX.DISPLAY 5)

(RPAQ? IM.NAME.DEFAULT.HASHFILE NIL)

(RPAQ? IM.NAME.HASHFILE.ABBREVS (QUOTE ((INTERLISP . {ERINYES}<LISPMANUAL>INTERLISP.IMNAMEHASH)
                                         (LOOPS . {INDIGO}<LOOPS>MANUAL>LOOPS.IMNAMEHASH))))

(FILESLOAD HASH)

(DECLARE: EVAL@COMPILE

(PUTPROPS IMNAME.RESETSAVE.MOVD MACRO [X (BQUOTE (RESETSAVE (MOVD , (CAR X)
                                                                  ,
                                                                  (CADR X)
                                                                  ,
                                                                  (CADDR X))
                                                        (LIST [QUOTE (LAMBDA (FN DEF)
                                                                                (PUTD FN DEF]
                                                              ,
                                                              (CADR X)
                                                              (GETD , (CADR X])
)

(PUTPROPS IMNAME COPYRIGHT ("Xerox Corporation" 1983 1984 1985 1986))
```

FUNCTION INDEX

VARIABLE INDEX

MACRO INDEX