File created: 15-Apr-87 10:54:01 **{IVY}<BLOOMBERG>LISP>MATHSERVER.;1**

changes to: (METHODS Server.ExecuteCommandFile)
(FNS Server.ExecuteCommandFile MS.TopLevel MS.ExpandFilename MS.SubmitBatchJob
MS.RunInteractiveJob MS.Compile MS.Link MS.CompileLink MS.CompileLinkRun MS.StartDefaultFE)
(VARS MATHSERVERCOMS)

previous date: 12-Dec-86 19:13:28 {PHYLUM}<LISPUSERS>KOTO>MATHSERVER.;1

Read Table: OLD-INTERLISP-FILE

Package: INTERLISP

Format: XCCS


```
(RPAQQ MATHSERVERCOMS
       ((* * SERVER MENU - Sets up the main Server Free Menu)
        (* MENU AND WINDOW FUNCTIONS)
        (FNS MS.TopLevel MS.CreateFreeMenu MS.SelectHost MS.ExpandFilename MS.CloseErrorWindow
             MS.CleanupErrorFile MS.AttachErrorWindow MS.MostRoom MS.GetMessageWindow MS.MakeIconWindow PrintMsg
             )
        (* SERVER METACLASS FUNCTIONS)
        (FNS MS.MakeMenuOfKnownHosts MS.DestroyInstances)
        (* MATH SERVER FUNCTIONS)
        (FNS MS.SubmitBatchJob MS.AbortBatchJob MS.Status MS.DisplayStatus MS.RunInteractiveJob MS.Compile
             MS.Link MS.CompileLink MS.CLR.Check MS.CLR.NoCheck MS.CompileLinkRun)
        (* FORTRAN EDITOR FUNCTIONS)
        (FNS MS.StartNewFE MS.StartDefaultFE MS.FindFortranEdit MS.CheckForDirtyFile)
        (* ERROR HANDLING FUNCTIONS)
        (FNS MS.BatchErrors? MS.BatchLog)
        (* Icon BITMAPS)
        (BITMAPS MS.Icon MS.IconMask)
        (* VARS)
        (ADDVARS (BackgroundMenuCommands (Server% Menu (QUOTE (MS.TopLevel))
                                                 "Start the Server Menu")))
        (VARS (BackgroundMenu NIL))
        (GLOBALVARS MS.HostPopMenu)
        (* * FORTRAN EDIT - Sets up a Fortran Edit Process)
        (* MAIN FUNCTIONS)
        (FNS FE.TopLevel FE.AdjustProps FE.CaretPosition TEDIT.PARA&CHAR FE.CharFn FE.GetEditProps
             FE.GetSourceFileName FE.LoopFn)
        (* WINDOW FUNCTIONS)
        (FNS FE.GetPositionWindow FE.GetEditWindow FE.GetMessageWindow FE.ReshapeFn FE.ShadeWindow)
        (* LOCALMENU FUNCTIONS)
        (FNS FE.CreateLocalMenu FE.SetHost FE.SetDirectory FE.MyGet FE.MyPut FE.StripVersion FE.Compile FE.Link
             FE.CompileLinkRun FE.RunInteractive)
        (* SERVER METACLASS FUNCTIONS)
        (FNS FE.ValidHostname FE.GetServer)
        (* ICON STUFF)
        (FNS FE.ShrinkIconCreate)
        (BITMAPS FE.Icon FE.IconMask)
        (INITVARS [FE.defaultFont (FONTCLASS (QUOTE FORTRANEDITFONT)
                                             (QUOTE (1 (GACHA 12)
                                                       (GACHA 10)
                                                       (GACHA 10]
               (FE.iconFont (FONTCREATE (QUOTE HELVETICA)
                                        8
                                        (QUOTE BOLD)))
               (FE.iconTitleRegion (create REGION LEFT _ 8 BOTTOM _ 8 WIDTH _ 110 HEIGHT _ 40))
               (FE.titledIconTemplate (create TITLEDICON ICON _ FE.Icon MASK _ FE.IconMask TITLEREG _
                                              FE.iconTitleRegion)))
        (* VARS)
        (ADDVARS (BackgroundMenuCommands (Fortran% Edit (QUOTE (FE.TopLevel))
                                                 "Start a Fortran Edit")))
        (VARS (BackgroundMenu NIL))
        (GLOBALVARS FE.defaultFont FE.iconFont FE.titledIconTemplate TEDIT.READTABLE)
        (* * SERVERS -- Defines the Loops MathServer objects)
        (CLASSES Cray FortranServer MathServer Server VMSServer)
        (METHODS FortranServer.Compile FortranServer.Compiled? FortranServer.Link FortranServer.Linked?
               MathServer.AlertManager Server.AbortJob Server.CommandFileExtension Server.Description
               Server.Error? Server.ErrorFile Server.ErrorString Server.ExecuteCommandFile
               Server.ExtractFilename Server.GetQueues Server.GetTime Server.Host Server.MakeError
               Server.MakeFullName Server.MakePartialName Server.Name Server.PutErrorInWindow
               Server.PutTextInWindow Server.Result Server.RunFile Server.RunJob Server.ServerDirectory
               Server.SourceExtension Server.Status Server.SubmitJob Server.UserDirectory
               VMSServer.MakeCommandString)
        (FNS MS.MakeInstances StripPA)
        (P (MS.DestroyInstances)
           (MS.MakeInstances))
        (* * PROGRAMCHAT - Windowless CHAT for communication)
        (FNS OPENCHATSTREAM PROGRAMCHAT PROGRAMCHAT.LOGIN PROGRAMCHAT.OUTPUT)
        (* VARS for our site)
        (VARS NETWORKLOGINFO)
        [P (pushnew NETWORKOSTYPES (QUOTE (GSLVAX . VMS))
                  (QUOTE (SITKA . VMS))
```

```
                (QUOTE (MADVAX . VMS]
        (* * PROGRAMMER'S INTERFACE - use remote servers with LISP calls)
        (FNS PRIN.RunRemote PRIN.ValidateHost PRIN.ValidateFilename PRIN.Error)))
```

(* * SERVER MENU -
Sets up the main Server Free Menu)

(* * MENU AND WINDOW FUNCTIONS)

```
(DEFINEQ
```

(MS.TopLevel

```
  [LAMBDA NIL                                                    (* DSB "15-Apr-87 10:19")
                                                                 (* Sets up the ServerFreeMenu, with PopUpMenu for host
                                                                 selection and with attached messageWindow)
    (PROG (menuWindow menuRegion messageWindow side)
          (SETQ menuWindow (MS.CreateFreeMenu 470 470))
          (FM.CHANGESTATE (FM.ITEMFROMID menuWindow (QUOTE QUEUE))
                 menuWindow)
          (SETQ menuRegion (WINDOWPROP menuWindow (QUOTE REGION)))
```

(* * Create PopUp menu for selection of Host)

```
          (SETQ MS.HostPopMenu (MS.MakeMenuOfKnownHosts))
```

(* * create, attach and save pointer to messageWindow)

```
          (SETQ messageWindow (CREATEW (CREATEREGION 0 0 200 150)
                                       "Message Window" NIL T))
          (SETQ side (QUOTE LEFT))
          (COND
             ((EQ (QUOTE LEFT)
                  (MS.MostRoom menuWindow))
              (SETQ side (QUOTE RIGHT)))
             (T NIL))
          (ATTACHWINDOW messageWindow menuWindow side (QUOTE JUSTIFY))
          (WINDOWPROP menuWindow (QUOTE MessageWindow)
                 messageWindow)
          (WINDOWPROP menuWindow (QUOTE ICONFN)
                 (FUNCTION MS.MakeIconWindow))
          (OPENW menuWindow])
```

(MS.CreateFreeMenu

```
  [LAMBDA (LEFT BOTTOM)                                          (* DSB " 9-Dec-86 15:50")
                                                                 (* returns a free menu window at specified position)
    (FM.FORMATMENU (BQUOTE (((TYPE TITLE LABEL FortranEdit-Commands: FONT (MODERN 12 BOLD))
                             (LABEL StartNew SELECTEDFN MS.StartNewFE)
                             (LABEL StartWithDefault SELECTEDFN MS.StartDefaultFE))
                            ((TYPE TITLE LABEL PlotMenu-Commands: FONT (MODERN 12 BOLD))
                             (LABEL SimplePlot SELECTEDFN MAPL.Simple.TopLevel)
                             (LABEL Gen.Plot SELECTEDFN MAPL.Gen.TopLevel)
                             (LABEL MetaPlot SELECTEDFN MAPL.Meta.TopLevel))
                            ((TYPE TITLE LABEL Compiler-Commands: FONT (MODERN 12 BOLD))
                             (LABEL Compile SELECTEDFN MS.Compile)
                             (LABEL Link SELECTEDFN MS.Link)
                             (LABEL C/L SELECTEDFN MS.CompileLink))
                            ((TYPE TITLE LABEL Run-Commands: FONT (MODERN 12 BOLD))
                             (LABEL RunInteractive SELECTEDFN MS.RunInteractiveJob)
                             (LABEL C/L/R SELECTEDFN MS.CLR.NoCheck)
                             (LABEL C?/L?/R SELECTEDFN MS.CLR.Check))
                            ((TYPE TITLE LABEL Batch-Commands: FONT (MODERN 12 BOLD))
                             (LABEL Submit SELECTEDFN MS.SubmitBatchJob)
                             (LABEL Status SELECTEDFN MS.Status)
                             (LABEL Errors? SELECTEDFN MS.BatchErrors?)
                             (LABEL Abort SELECTEDFN MS.AbortBatchJob)
                             (LABEL Log SELECTEDFN MS.BatchLog))
                            ((TYPE TITLE LABEL "COMPUTE SERVER FILE INFO" FONT (MODERN 12 BOLD)))
                            ((TYPE EDITSTART LABEL Filename: FONT (MODERN 12 BOLD)
                                   ITEMS
                                   (FILENAME))
                             (TYPE EDIT ID FILENAME LABEL ""))
                            ((TYPE EDITSTART LABEL DefaultDirectory: FONT (MODERN 12 BOLD)
                                   ITEMS
                                   (DEFAULTDIRECTORY))
                             (TYPE EDIT ID DEFAULTDIRECTORY LABEL ""))
                            ((TYPE EDITSTART LABEL JobParameters: FONT (MODERN 12 BOLD)
                                   ITEMS
                                   (PARAMETERSTRING))
                             (TYPE EDIT ID PARAMETERSTRING LABEL ""))
                            ((TYPE EDITSTART LABEL LinkParameters: FONT (MODERN 12 BOLD)
                                   ITEMS
                                   (LINKSTRING))
                             (TYPE EDIT ID LINKSTRING LABEL ""))
                            ((TYPE TITLE LABEL "COMPUTE SERVER HOST INFO" FONT (MODERN 12 BOLD)))
                            ((TYPE TITLE LABEL HostName: FONT (MODERN 12 BOLD)
```

```
                                SELECTEDFN MS.SelectHost)
                             (TYPE TITLE ID HOST LABEL ""))
                            ((TYPE TITLE LABEL Queue: FONT (MODERN 12 BOLD))
                             (TYPE NWAY ID QUEUE LABEL Fast CLASSNAME FastQueue)
                             (TYPE NWAY ID QUEUE LABEL Medium CLASSNAME MediumQueue)
                             (TYPE NWAY ID QUEUE LABEL Slow CLASSNAME SlowQueue))
                            ((TYPE EDITSTART LABEL JobNumber: FONT (MODERN 12 BOLD)
                                   ITEMS
                                   (JOBNUMBER))
                             (TYPE EDIT ID JOBNUMBER LABEL ""))
                            ((TYPE TITLE LABEL SERVERBROWSER-Command: FONT (MODERN 12 BOLD))
                             (LABEL MakeBrowser SELECTEDFN MS.MakeInstances))
                            (WINDOWPROPS TITLE "Server Menu" LEFT , LEFT BOTTOM , BOTTOM])
```

(MS.SelectHost

```
[LAMBDA (ITEM WINDOW BUTTONS)                                    (* DSB "10-Jun-86 18:17")
```

(* Uses the pop-up menu MS.HostPopMenu to return a label and a pointer to the Host Server.)

```
  (PROG ((promptW (GETPROMPTWINDOW WINDOW))
         (sItem (FM.ITEMFROMID WINDOW (QUOTE HOST)))
         server)
```

(* * Opens the PopUp menu. Returns the object name of the selected server.)

```
        (CLEARW promptW)
        (PRIN1 "Select host." promptW)
        (SETQ server (MENU MS.HostPopMenu))
        (CLEARW promptW)
```

(* * if the server exists, set the Host prop of the "HOST" item in the menu to point to the Server object.
Then change the item label to be the name of the Server object.)

```
        (COND
           (server (FM.ITEMPROP sItem (QUOTE Host)
                           server)
                   (FM.CHANGELABEL sItem WINDOW (_ server Name)))
           (T
```

(* * otherwise, set both the Host prop and the label of the "HOST" item in the menu to nil.)

```
                 (FM.ITEMPROP sItem (QUOTE Host)
                        NIL)
                 (FM.CHANGELABEL sItem WINDOW ""])
```

(MS.ExpandFilename

```
[LAMBDA (ITEM WINDOW BUTTONS)                                    (* DSB "15-Apr-87 10:22")
                                                                 (* if filename contains a directory, expand it into separate slots)
                                                                 (* this is a shortened version of MAPL.ExpandFilename)
  (PROG ((state (FM.READSTATE WINDOW))
         filename defaultDirectory name)
        (SETQ filename (LISTGET state (QUOTE FILENAME)))
```

(* * strip off any extensions and version numbers)

```
        (SETQ name (UNPACKFILENAME filename (QUOTE NAME)))
        (FM.CHANGELABEL (FM.ITEMFROMID WINDOW (QUOTE FILENAME))
               WINDOW name)
```

(* * if there is a directory, place it in the menu)

```
        (SETQ defaultDirectory (UNPACKFILENAME filename (QUOTE DIRECTORY)))
        (COND
           (defaultDirectory (FM.CHANGELABEL (FM.ITEMFROMID WINDOW (QUOTE DEFAULTDIRECTORY))
                                    WINDOW defaultDirectory)))
```

(* * update "state" and return)

```
        (SETQ state (FM.READSTATE WINDOW))
        (RETURN state])
```

(MS.CloseErrorWindow

```
[LAMBDA (mainWindow)                                             (* DSB "15-Aug-86 15:27")
```

(* * Check if an errorWindow already exists. If so, delete the {core} file behind it, clean up props, and finally close the window.)

```
  (LET (oldWindow oldFile)
       [SETQ oldWindow (find item in (ATTACHEDWINDOWS mainWindow) suchthat (WINDOWPROP item (QUOTE ERRORFILE]
       (COND
          (oldWindow (MS.CleanupErrorFile oldWindow)
                 (CLOSEW oldWindow])
```

(MS.CleanupErrorFile

```
[LAMBDA (errorWindow)                                    (* DSB "15-Aug-86 15:30")

          (* deletes the error file that resides in {core}, if it exists, and resets errorWindow prop to NIL)

    (LET [(oldFile (WINDOWPROP errorWindow (QUOTE ERRORFILE]
         (COND
            (oldFile (CLOSEF? oldFile)
                   (DELFILE oldFile)
                   (WINDOWPROP errorWindow (QUOTE ERRORFILE)
                          NIL)))
         NIL])
```

(MS.AttachErrorWindow

```
[LAMBDA (mainWindow title)                               (* DSB "15-Aug-86 15:22")
                                                         (* Attaches an error window to the main menu window)

          (* * MS.CloseErrorWindow should have been called prior to this.
          Nevertheless, we check for an old errorWindow, and if it exists, we call MS.CloseErrorWindow)

          (* * Make the error window and attach it to the appropriate side of the main window)

    (LET (oldWindow errorWindow)
         [SETQ oldWindow (find item in (ATTACHEDWINDOWS mainWindow) suchthat (WINDOWPROP item (QUOTE ERRORFILE]
         (AND oldWindow (MS.CloseErrorWindow mainWindow))
         (SETQ errorWindow (CREATEW (QUOTE (0 0 470 300))
                                 title NIL T))
         (ATTACHWINDOW errorWindow mainWindow (MS.MostRoom mainWindow)
                (QUOTE JUSTIFY)
                (QUOTE LOCALCLOSE))
        errorWindow])
```

(MS.MostRoom

```
[LAMBDA (WINDOW)                                         (* DSB " 7-Aug-86 11:55")
                                                         (* determines if attached window should be on right or left side
                                                         of main window)
    (LET ((region (WINDOWPROP WINDOW (QUOTE REGION)))
          leftSpace rightSpace)
         (SETQ leftSpace (fetch (REGION LEFT) of region))
         [SETQ rightSpace (DIFFERENCE 1025 (PLUS leftSpace (fetch (REGION WIDTH) of region]
         (COND
            ((GEQ leftSpace rightSpace)
             (QUOTE LEFT))
            (T (QUOTE RIGHT])
```

(MS.GetMessageWindow

```
[LAMBDA (WINDOW)                                         (* DSB " 6-Jun-86 15:57")

          (* maybe later, we'll check if the message window exists and if not, will make it first)

    (WINDOWPROP WINDOW (QUOTE MessageWindow])
```

(MS.MakeIconWindow

```
[LAMBDA (WINDOW OLDICON)                                 (* DSB " 9-Dec-86 15:44")

          (* * Creates a window with an icon formed by two bit maps.)

    (OR OLDICON (ICONW MS.Icon MS.IconMask])
```

(PrintMsg

```
[LAMBDA (place msg)                                      (* thh: " 6-Nov-85 11:04")
                                                         (* prints message in appropriate prompt window)
                                                         (* no message if place = DON'T)

    (COND
       ((Object? place)
        (_ place ClearPromptWindow)
        (_ place PromptPrint msg))
       ((TYPENAMEP place (QUOTE PLOT))
        (LET [(w (PLOTPROP place (QUOTE PLOTPROMPTWINDOW]
             (CLEARW w)
             (PRIN1 msg w)))
       ((WINDOWP place)
        (LET ((w (GETPROMPTWINDOW place 2)))
             (CLEARW w)
             (PRIN1 msg w)))
       ((EQ place (QUOTE DON'T)))
       (T (CLRPROMPT)
          (PROMPTPRINT msg])

)
```

```
          (* * SERVER METACLASS FUNCTIONS)

(DEFINEQ

(MS.MakeMenuOfKnownHosts
  [LAMBDA NIL                                                    (* DSB "19-Aug-86 17:05")
                                                                 (* makes the MS.HostPopMenu)

    (create MENU
           ITEMS _ (for server in (_ ($ Server)
                                    AllInstances!)
                      collect (LIST (_ server Name)
                                    server
                                    (_ server Description])

(MS.DestroyInstances
  [LAMBDA NIL                                                    (* DSB "19-Aug-86 15:37")
                                                                 (* obvious! Use MS.MakeInstances after this.)

    (for instance in (_ ($ Server)
                        AllInstances!)
       do (_ instance Destroy])

)


          (* * MATH SERVER FUNCTIONS)

(DEFINEQ

(MS.SubmitBatchJob
  [LAMBDA (ITEM WINDOW BUTTONS)                                  (* DSB "15-Apr-87 10:28")
                                                                 (* Creates the SubmitJob message to be sent to the appropriate
    host)
    (PROG (state (promptW (GETPROMPTWINDOW WINDOW))
                 (messageW (MS.GetMessageWindow WINDOW))
                 filename defaultDirectory host validQueues queue parameterString result)

          (* * check that all required data is specified)

          (SETQ state (MS.ExpandFilename ITEM WINDOW BUTTONS))
          (SETQ filename (LISTGET state (QUOTE FILENAME)))
          (COND
             ((EQUAL filename "")
              (PrintMsg WINDOW "Unspecified file name.")
              (RETURN)))
          (SETQ defaultDirectory (LISTGET state (QUOTE DEFAULTDIRECTORY)))
          (COND
             ((EQUAL defaultDirectory "")
              (PrintMsg WINDOW "Unspecified default directory.")
              (RETURN)))
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                                  (QUOTE Host)))
          (COND
             ((NOT host)
              (PrintMsg WINDOW "Unspecified host.")
              (RETURN)))
          (SETQ validQueues (_ host GetQueues))
          (SETQ queue (LISTGET state (QUOTE QUEUE)))
          (COND
             ((NOT queue)
              (PrintMsg WINDOW "Unspecified queue.")
              (RETURN))
             ((NOT (MEMBER queue validQueues))
              (CLEARW promptW)
              (PRIN1 "Not a valid queue for this server" promptW)
              (TERPRI promptW)
              (PRINTOUT promptW "Valid queues:  " validQueues)
              (RETURN)))
          (SETQ parameterString (LISTGET state (QUOTE PARAMETERSTRING)))
          (COND
             ((STRPOS "," parameterString)
              (PrintMsg WINDOW "Remove comma(s) from JobParameters")
              (RETURN)))

          (* * return the SubmitJob message)

          (PrintMsg WINDOW "Submitting Batch Job ...")
          (CLEARW messageW)
          (MS.CloseErrorWindow WINDOW)
          [COND
             ((EQUAL parameterString "")
              (SETQ result (_ host SubmitJob (PACKFILENAME (QUOTE DIRECTORY)
                                                           defaultDirectory
                                                           (QUOTE BODY)
                                                           filename)
                              queue)))
             (T (SETQ result (_ host SubmitJob (PACKFILENAME (QUOTE DIRECTORY)
```

```
                                                    defaultDirectory
                                                    (QUOTE BODY)
                                                    filename)
                                queue
                                (LIST (MKSTRING parameterString]
          (PRIN1 result messageW)
          (CLEARW promptW)
          (PRIN1 "Done" promptW)
          (FM.CHANGELABEL (FM.ITEMFROMID WINDOW (QUOTE JOBNUMBER))
                 WINDOW
                 (CAR result])
```

(MS.AbortBatchJob

```
  [LAMBDA (ITEM WINDOW BUTTONS)                                    (* DSB "12-Aug-86 09:52")
                                                                   (* Creates the AbortJob message to be sent to the appropriate
                                                                   host)
    (PROG ((state (FM.READSTATE WINDOW))
           (promptW (GETPROMPTWINDOW WINDOW))
           (messageW (MS.GetMessageWindow WINDOW))
           host jobNumber queue validQueues result)

          (* * check that the host name is specified)

          (CLEARW promptW)
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                                  (QUOTE Host)))
          (COND
             ((NOT host)
              (PrintMsg WINDOW "Unspecified host.")
              (RETURN)))

          (* * check that the job number is specified)

          (SETQ jobNumber (LISTGET state (QUOTE JOBNUMBER)))
          (COND
             ((EQUAL jobNumber "")
              (SETQ jobNumber NIL)
              (PrintMsg WINDOW "Unspecified job number.")
              (RETURN)))

          (* * check that the queue is specified and valid)

          (SETQ validQueues (_ host GetQueues))
          (SETQ queue (LISTGET state (QUOTE QUEUE)))
          (COND
             ((NOT queue)
              (PrintMsg WINDOW "Unspecified queue.")
              (RETURN))
             ((NOT (MEMBER queue validQueues))
              (CLEARW promptW)
              (PRIN1 "Not a valid queue for this server" promptW)
              (TERPRI promptW)
              (PRINTOUT promptW "Valid queues:  " validQueues)
              (RETURN)))

          (* * abort the job and return the result)

          (CLEARW promptW)
          (PRINTOUT promptW "Job " jobNumber " on queue " queue " is being aborted ...")
          (SETQ result (_ host AbortJob jobNumber queue))
          (TERPRI promptW)
          (PRIN1 "Done" promptW)
          (CLEARW messageW)
          (PRIN1 result messageW])
```

(MS.Status

```
  [LAMBDA (ITEM WINDOW BUTTONS)                                    (* DSB "12-Aug-86 17:53")
                                                                   (* Creates the Status message to be sent to the appropriate
                                                                   host)
    (PROG ((state (FM.READSTATE WINDOW))
           (promptW (GETPROMPTWINDOW WINDOW))
           (messageW (MS.GetMessageWindow WINDOW))
           host jobNumber result)

          (* * check that the host name is specified)

          (CLEARW promptW)
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                                  (QUOTE Host)))
          (COND
             ((NOT host)
              (PrintMsg WINDOW "Unspecified host.")
              (RETURN)))

          (* * give notice if no job number is supplied)
```

```
          (SETQ jobNumber (LISTGET state (QUOTE JOBNUMBER)))
          (COND
             ((NUMBERP (MKATOM jobNumber))
              (PRIN1 "Finding Status ..." promptW))
             (T (SETQ jobNumber NIL)
                (PRIN1 "Unspecified job number." promptW)
                (TERPRI promptW)
                (PRIN1 "Finding status of all batch jobs ..." promptW)))
```

(* * return the Status message)

```
          (CLEARW messageW)
          (SETQ result (_ host Status jobNumber))
          (MS.DisplayStatus result messageW)
          (CLEARW promptW)
          (PRIN1 "Done" promptW])
```

(MS.DisplayStatus

```
  [LAMBDA (result messageW)                                    (* DSB "12-Aug-86 18:00")
                                                               (* Displays the status in nice format in messageWindow)
    (PROG (number time)
          (TERPRI messageW)
          (PRIN1 "     JOB            CPU" messageW)
          (TERPRI messageW)
          (PRIN1 "- - - - - - - - - - - - - - - - - -" messageW)
```

(* * is it a single data item (JOB in CAAR) or...)

```
          (COND
             ((EQ (CAAR result)
                  (QUOTE JOB))
              (SETQ number (CADAR result))
              (SETQ time (CADADR result))
              (TERPRI messageW)
              (PRIN1 (CONCAT "     " number "       " time)
                     messageW))
             (T (for item in result do (SETQ number (CADAR item))
                                       (SETQ time (CADADR item))
                                       (TERPRI messageW)
                                       (PRIN1 (CONCAT "     " number "       " time)
                                              messageW])
```

(MS.RunInteractiveJob

```
  [LAMBDA (ITEM WINDOW BUTTONS)                                (* DSB "15-Apr-87 10:30")
                                                               (* Creates the RunJob message to be sent to the appropriate
                                                               host)
    (PROG (state (promptW (GETPROMPTWINDOW WINDOW))
                 (messageW (MS.GetMessageWindow WINDOW))
                 filename defaultDirectory host parameterString result errorFile errorWindow)
```

(* * check that all required data is specified)

```
          (SETQ state (MS.ExpandFilename ITEM WINDOW BUTTONS))
          (SETQ filename (LISTGET state (QUOTE FILENAME)))
          (COND
             ((EQUAL filename "")
              (PrintMsg WINDOW "Unspecified file name.")
              (RETURN)))
          (SETQ defaultDirectory (LISTGET state (QUOTE DEFAULTDIRECTORY)))
          (COND
             ((EQUAL defaultDirectory "")
              (PrintMsg WINDOW "Unspecified default directory.")
              (RETURN)))
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                                  (QUOTE Host)))
          (COND
             ((NOT host)
              (PrintMsg WINDOW "Unspecified host.")
              (RETURN)))
          (SETQ parameterString (LISTGET state (QUOTE PARAMETERSTRING)))
          (COND
             ((STRPOS "," parameterString)
              (PrintMsg WINDOW "Remove comma(s) from JobParameters")
              (RETURN)))
```

(* * return the RunJob message)

```
          (CLEARW promptW)
          (PRIN1 "Running interactive job ..." promptW)
          (CLEARW messageW)
          (MS.CloseErrorWindow WINDOW)
          [COND
             [(EQUAL parameterString "")
              (SETQ result (_ host RunJob (PACKFILENAME (QUOTE DIRECTORY)
```

```
                                                defaultDirectory
                                                (QUOTE BODY)
                                                filename]
            (T (SETQ result (_ host RunJob (PACKFILENAME (QUOTE DIRECTORY)
                                                  defaultDirectory
                                                  (QUOTE BODY)
                                                  filename)
                                   (LIST (MKSTRING parameterString]
         (PRIN1 result messageW)
         (CLEARW promptW)
         (COND
            ((EQUAL (CAR result)
                    (QUOTE ERROR))
             (PRIN1 "Run-time warning or error" promptW)
             (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Run-time Errors"))
             (SETQ errorFile (_ host ExtractFilename result))
             (_ host PutErrorInWindow errorFile errorWindow WINDOW))
            (T (PRIN1 "Done" promptW])
```

(MS.Compile

```
  [LAMBDA (ITEM WINDOW BUTTONS)                                    (* DSB "15-Apr-87 10:14")
                                                                   (* Creates the Compile message to be sent to the appropriate
                                                                   host)
    (PROG (state (promptW (GETPROMPTWINDOW WINDOW))
                 (messageW (MS.GetMessageWindow WINDOW))
                 filename defaultDirectory host result errorFile errorWindow)
          (CLEARW messageW)

          (* * check that all required data is specified)

          (SETQ state (MS.ExpandFilename ITEM WINDOW BUTTONS))
          (SETQ filename (LISTGET state (QUOTE FILENAME)))
          (COND
             ((EQUAL filename "")
              (PrintMsg WINDOW "Unspecified file name.")
              (RETURN)))
          (SETQ defaultDirectory (LISTGET state (QUOTE DEFAULTDIRECTORY)))
          (COND
             ((EQUAL defaultDirectory "")
              (PrintMsg WINDOW "Unspecified default directory.")
              (RETURN)))
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                                  (QUOTE Host)))
          (COND
             ((NOT host)
              (PrintMsg WINDOW "Unspecified host.")
              (RETURN)))

          (* * check for dirty file in a Fortran Edit)

          (COND
             ((MS.CheckForDirtyFile filename promptW messageW)
              (RETURN)))

          (* * send the Compile message)

          (CLEARW promptW)
          (PRIN1 "Compiling source file ..." promptW)
          (MS.CloseErrorWindow WINDOW)
          (SETQ result (_ host Compile (PACKFILENAME (QUOTE DIRECTORY)
                                                     defaultDirectory
                                                     (QUOTE BODY)
                                                     filename)))
          (CLEARW messageW)
          (PRIN1 result messageW)
          (CLEARW promptW)
          (COND
             ((EQUAL (CAR result)
                     (QUOTE ERROR))
              (PRIN1 "Compilation Error" promptW)
              (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Compilation Errors"))
              (SETQ errorFile (_ host ExtractFilename result))
              (_ host PutErrorInWindow errorFile errorWindow WINDOW))
             (T (PRIN1 "Successful Compilation" promptW])
```

(MS.Link

```
  [LAMBDA (ITEM WINDOW BUTTONS)                                    (* DSB "15-Apr-87 10:31")
                                                                   (* Creates the Link message to be sent to the appropriate host)
    (PROG (state (promptW (GETPROMPTWINDOW WINDOW))
                 (messageW (MS.GetMessageWindow WINDOW))
                 filename defaultDirectory host linkString result errorFile errorWindow)

          (* * check that all required data is specified)

          (SETQ state (MS.ExpandFilename ITEM WINDOW BUTTONS))
```

```
          (SETQ filename (LISTGET state (QUOTE FILENAME)))
          (COND
             ((EQUAL filename "")
              (PrintMsg WINDOW "Unspecified file name.")
              (RETURN)))
          (SETQ defaultDirectory (LISTGET state (QUOTE DEFAULTDIRECTORY)))
          (COND
             ((EQUAL defaultDirectory "")
              (PrintMsg WINDOW "Unspecified default directory.")
              (RETURN)))
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                                  (QUOTE Host)))
          (COND
             ((NOT host)
              (PrintMsg WINDOW "Unspecified host.")
              (RETURN)))
          (SETQ linkString (LISTGET state (QUOTE LINKSTRING)))
          (COND
             ((STRPOS " " linkString)
              (PrintMsg WINDOW "Remove spaces from LinkParameters")
              (RETURN)))
```

(* * return the Link message)

```
          (CLEARW promptW)
          (PRIN1 "Linking ..." promptW)
          (CLEARW messageW)
          (MS.CloseErrorWindow WINDOW)
          [COND
             [(EQUAL linkString "")
              (SETQ result (_ host Link (PACKFILENAME (QUOTE DIRECTORY)
                                                      defaultDirectory
                                                      (QUOTE BODY)
                                                      filename]
             (T (SETQ result (_ host Link (PACKFILENAME (QUOTE DIRECTORY)
                                                        defaultDirectory
                                                        (QUOTE BODY)
                                                        filename)
                                 (LIST (MKSTRING linkString]
          (PRIN1 result messageW)
          (CLEARW promptW)
          (COND
             ((EQUAL (CAR result)
                     (QUOTE ERROR))
              (PRIN1 "Linking Error" promptW)
              (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Link Errors"))
              (SETQ errorFile (_ host ExtractFilename result))
              (_ host PutErrorInWindow errorFile errorWindow WINDOW))
             (T (PRIN1 "Done:  successful link" promptW])
```

(MS.CompileLink

```
  [LAMBDA (ITEM WINDOW BUTTONS)                                   (* DSB "15-Apr-87 10:31")
                                                                  (* Creates the Compile and Link messages to be sent to the
                                                                  appropriate host)
    (PROG (state (promptW (GETPROMPTWINDOW WINDOW))
                 (messageW (MS.GetMessageWindow WINDOW))
                 filename defaultDirectory host linkString result errorFile errorWindow)
          (CLEARW messageW)
```

(* * check that all required data is specified)

```
          (SETQ state (MS.ExpandFilename ITEM WINDOW BUTTONS))
          (SETQ filename (LISTGET state (QUOTE FILENAME)))
          (COND
             ((EQUAL filename "")
              (PrintMsg WINDOW "Unspecified file name.")
              (RETURN)))
          (SETQ defaultDirectory (LISTGET state (QUOTE DEFAULTDIRECTORY)))
          (COND
             ((EQUAL defaultDirectory "")
              (PrintMsg WINDOW "Unspecified default directory.")
              (RETURN)))
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                                  (QUOTE Host)))
          (COND
             ((NOT host)
              (PrintMsg WINDOW "Unspecified host.")
              (RETURN)))
          (SETQ linkString (LISTGET state (QUOTE LINKSTRING)))
          (COND
             ((STRPOS " " linkString)
              (PrintMsg WINDOW "Remove spaces from LinkParameters")
              (RETURN)))
```

(* * check for dirty file in a Fortran Edit)

```
          (COND
             ((MS.CheckForDirtyFile filename promptW messageW)
              (RETURN)))

          (* * send the Compile message)

          (CLEARW promptW)
          (PRIN1 "Compiling source file ..." promptW)
          (MS.CloseErrorWindow WINDOW)
          (SETQ result (_ host Compile (PACKFILENAME (QUOTE DIRECTORY)
                                                     defaultDirectory
                                                     (QUOTE BODY)
                                                     filename)))
          (CLEARW messageW)
          (PRIN1 result messageW)
          (CLEARW promptW)
          (COND
             ((EQUAL (CAR result)
                     (QUOTE ERROR))
              (PRIN1 "Compilation error" promptW)
              (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Compilation Errors"))
              (SETQ errorFile (_ host ExtractFilename result))
              (_ host PutErrorInWindow errorFile errorWindow WINDOW)
              (RETURN))
             (T (PRIN1 "Compile finished.  Now linking..." promptW)))

          (* * return the Link message)

          [COND
             [(EQUAL linkString "")
              (SETQ result (_ host Link (PACKFILENAME (QUOTE DIRECTORY)
                                                      defaultDirectory
                                                      (QUOTE BODY)
                                                      filename]
             (T (SETQ result (_ host Link (PACKFILENAME (QUOTE DIRECTORY)
                                                        defaultDirectory
                                                        (QUOTE BODY)
                                                        filename)
                                 (LIST (MKSTRING linkString]
          (CLEARW messageW)
          (PRIN1 result messageW)
          (CLEARW promptW)
          (COND
             ((EQUAL (CAR result)
                     (QUOTE ERROR))
              (PRIN1 "Linking Error" promptW)
              (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Link Errors"))
              (SETQ errorFile (_ host ExtractFilename result))
              (_ host PutErrorInWindow errorFile errorWindow WINDOW))
             (T (PRIN1 "Done:  successful link" promptW])
```

(MS.CLR.Check

```
  [LAMBDA (ITEM WINDOW BUTTONS)                          (* DSB "12-Aug-86 10:17")
                                                         (* This functin calls MS.CompileLinkRun with the check flag T)
    (MS.CompileLinkRun ITEM WINDOW BUTTONS T])
```

(MS.CLR.NoCheck

```
  [LAMBDA (ITEM WINDOW BUTTONS)                          (* DSB "12-Aug-86 10:18")
                                                         (* This functin calls MS.CompileLinkRun with the check flag NIL)
    (MS.CompileLinkRun ITEM WINDOW BUTTONS NIL])
```

(MS.CompileLinkRun

```
  [LAMBDA (ITEM WINDOW BUTTONS checkFlag)                (* DSB "15-Apr-87 10:34")

          (* Sequentially creates the Compile, Link and RunJob messages, to be sent to the appropriate host)

    (PROG (state (promptW (GETPROMPTWINDOW WINDOW))
                 (messageW (MS.GetMessageWindow WINDOW))
                 filename defaultDirectory host parameterString linkString result errorFile errorWindow)
          (CLEARW messageW)

          (* * check that all required data is specified)

          (SETQ state (MS.ExpandFilename ITEM WINDOW BUTTONS))
          (SETQ filename (LISTGET state (QUOTE FILENAME)))
          (COND
             ((EQUAL filename "")
              (PrintMsg WINDOW "Unspecified file name.")
              (RETURN)))
          (SETQ defaultDirectory (LISTGET state (QUOTE DEFAULTDIRECTORY)))
          (COND
             ((EQUAL defaultDirectory "")
              (PrintMsg WINDOW "Unspecified default directory.")
              (RETURN)))
```

```
(SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                  (QUOTE Host)))
(COND
   ((NOT host)
    (PrintMsg WINDOW "Unspecified host.")
    (RETURN)))
(SETQ linkString (LISTGET state (QUOTE LINKSTRING)))
(COND
   ((STRPOS " " linkString)
    (PrintMsg WINDOW "Remove spaces from LinkParameters")
    (RETURN)))
(SETQ parameterString (LISTGET state (QUOTE PARAMETERSTRING)))
(COND
   ((STRPOS "," parameterString)
    (PrintMsg WINDOW "Remove comma(s) from JobParameters")
    (RETURN)))
```

(* * check for dirty file in a Fortran Edit)

```
(COND
   ((MS.CheckForDirtyFile filename promptW messageW)
    (RETURN)))
```

(* * send the Compile message)

```
(CLEARW promptW)
(MS.CloseErrorWindow WINDOW)
[COND
   ((AND checkFlag (_ host Compiled? host defaultDirectory filename))
    (PRIN1 "Source file already compiled. " promptW))
   (T (PRIN1 "Compiling source file ..." promptW)
      (SETQ result (_ host Compile (PACKFILENAME (QUOTE DIRECTORY)
                                          defaultDirectory
                                          (QUOTE BODY)
                                          filename)))
      (CLEARW messageW)
      (PRIN1 result messageW)
      (CLEARW promptW)
      (COND
         ((EQUAL (CAR result)
                 (QUOTE ERROR))
          (PRIN1 "Compilation error" promptW)
          (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Compilation Errors"))
          (SETQ errorFile (_ host ExtractFilename result))
          (_ host PutErrorInWindow errorFile errorWindow WINDOW)
          (RETURN))
         (T (PRIN1 "Compile finished.  " promptW]
```

(* * return the Link message)

```
[COND
   ((AND checkFlag (_ host Linked? host defaultDirectory filename))
    (TERPRI promptW)
    (PRIN1 "    Exec. file exists.  Now running it..." promptW))
   (T (PRIN1 "   Now linking..." promptW)
      [COND
         [(EQUAL linkString "")
          (SETQ result (_ host Link (PACKFILENAME (QUOTE DIRECTORY)
                                          defaultDirectory
                                          (QUOTE BODY)
                                          filename]
         (T (SETQ result (_ host Link (PACKFILENAME (QUOTE DIRECTORY)
                                            defaultDirectory
                                            (QUOTE BODY)
                                            filename)
                            (LIST (MKSTRING linkString]
      (CLEARW messageW)
      (PRIN1 result messageW)
      (CLEARW promptW)
      (COND
         ((EQUAL (CAR result)
                 (QUOTE ERROR))
          (PRIN1 "Linking Error" promptW)
          (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Link Errors"))
          (SETQ errorFile (_ host ExtractFilename result))
          (_ host PutErrorInWindow errorFile errorWindow WINDOW)
          (RETURN))
         (T (PRIN1 "Link finished.  Now running job ..." promptW]
```

(* * return the RunJob message)

```
[COND
   [(EQUAL parameterString "")
    (SETQ result (_ host RunJob (PACKFILENAME (QUOTE DIRECTORY)
                                       defaultDirectory
                                       (QUOTE BODY)
                                       filename]
```

```
            (T (SETQ result (_ host RunJob (PACKFILENAME (QUOTE DIRECTORY)
                                                 defaultDirectory
                                                 (QUOTE BODY)
                                                 filename)
                                   (LIST (MKSTRING parameterString]
         (CLEARW messageW)
         (PRIN1 result messageW)
         (CLEARW promptW)
         (COND
            ((EQUAL (CAR result)
                    (QUOTE ERROR))
             (PRIN1 "Run-time warning or error" promptW)
             (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Run-time Errors"))
             (SETQ errorFile (_ host ExtractFilename result))
             (_ host PutErrorInWindow errorFile errorWindow WINDOW))
            (T (PRIN1 "Done" promptW])

)
```

(* * FORTRAN EDITOR FUNCTIONS)

```
(DEFINEQ
```

(MS.StartNewFE

```
  [LAMBDA NIL                                                    (* DSB "21-Aug-86 15:38")
                                                                 (* starts a new Fortran Edit process without setting defaults or
                                                                 getting a file)
    (FE.TopLevel])
```

(MS.StartDefaultFE

```
  [LAMBDA (ITEM WINDOW BUTTONS)                                  (* DSB "15-Apr-87 10:35")
```

(* Starts a FE, sets defaults according to the values in the ServerMenu, and gets the source file named in the server menu, if it exists.)

```
    (PROG (state textStream host hostname defaultDirectory directory filename name getFilename)
          (SETQ textStream (FE.TopLevel))
          (SETQ state (MS.ExpandFilename ITEM WINDOW BUTTONS))
          (SETQ filename (LISTGET state (QUOTE FILENAME)))
          (SETQ defaultDirectory (LISTGET state (QUOTE DEFAULTDIRECTORY)))
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                                  (QUOTE Host)))
          [COND
             (host (TEXTPROP textStream (QUOTE MS.HOST)
                             host)
                   (SETQ hostname (_ host Name))
                   (TEXTPROP textStream (QUOTE MS.HOSTNAME)
                             hostname)
                   (TEXTPROP textStream (QUOTE MS.DIRECTORY)
                             (_ host UserDirectory]
          (COND
             ((NOT (EQUAL defaultDirectory ""))
              (SETQ directory (MKATOM defaultDirectory))
              (TEXTPROP textStream (QUOTE MS.DIRECTORY)
                        directory)))
          [COND
             ((NOT (EQUAL filename ""))
              (SETQ name (MKATOM filename]
          (COND
             ((AND host directory name)
              (SETQ getFilename (PACKFILENAME (QUOTE HOST)
                                              hostname
                                              (QUOTE DIRECTORY)
                                              directory
                                              (QUOTE NAME)
                                              name
                                              (QUOTE EXTENSION)
                                              (_ host SourceExtension)))
              (COND
                 ((INFILEP getFilename)
                  (TEDIT.GET (TEXTOBJ textStream)
                             getFilename)
                  (TEXTPROP textStream (QUOTE FILENAME)
                             getFilename)
                  (TEDIT.PROMPTPRINT textStream (CONCAT "Retrieved file " getFilename)))
                 (T (TEDIT.PROMPTPRINT textStream (CONCAT "File " getFilename " not found"])
```

(MS.FindFortranEdit

```
  [LAMBDA (filename)                                             (* DSB "22-Aug-86 09:36")
```

(* * searches through all open windows and returns a stream number of a Fortran Edit process whose filename matches the parameter filename If no such process exists, it returns NIL.)

```
    (LET (textStream fullEditFilename editFilename)
```

```
        (for window in (OPENWINDOWS) do (COND
                                        ((WINDOWPROP window (QUOTE FORTRANEDIT))
                                         (SETQ textStream (TEXTSTREAM window))
                                         (SETQ fullEditFilename (TEXTPROP textStream (QUOTE FILENAME)))
                                         (SETQ editFilename (UNPACKFILENAME fullEditFilename (QUOTE NAME)))
                                         (COND
                                            ((EQUAL (U-CASE editFilename)
                                                    (U-CASE (MKATOM filename)))
                                             (RETURN textStream))
                                            (T NIL])
```

(MS.CheckForDirtyFile

```
  [LAMBDA (filename promptW messageW)                    (* DSB "22-Aug-86 09:42")
```

(* * returns T (to abort the operation) only if there exists a dirty file with the same name and the user does not click the left button.)

```
    (PROG (textStream dirty)
```

(* * see if there is an active FE with filename)

```
          (SETQ textStream (MS.FindFortranEdit filename))
          (COND
             ((NOT textStream)
              (RETURN NIL)))
```

(* * if the file has been changed, give the user the option to abort the operation)

```
          (SETQ dirty (TEDIT.STREAMCHANGEDP textStream))
          (COND
             [dirty (CLEARW promptW)
                    (COND
                       ((MOUSECONFIRM "Not saved yet; LEFT to use previous version." T promptW)
                        (PRIN1 "Using old version on the server" messageW)
                        (RETURN NIL))
                       (T (PRIN1 "Operation aborted.  Put file in Fortran Edit to server." messageW)
                          (RETURN T]
             (T (PRIN1 "File in Fortran Editor has not been changed.  Operation proceeds" messageW)
                (RETURN NIL])

)
```

(* * ERROR HANDLING FUNCTIONS)

```
(DEFINEQ
```

(MS.BatchErrors?

```
  [LAMBDA (ITEM WINDOW BUTTONS)                          (* DSB "15-Aug-86 18:07")
                                                         (* If Batch errors have occurred, it displays them in the error window)
    (PROG ((state (FM.READSTATE WINDOW))
           (promptW (GETPROMPTWINDOW WINDOW))
           (messageW (MS.GetMessageWindow WINDOW))
           host defaultDirectory jobNumber errorFile localErrorFile resultFile stream result errorWindow)
```

(* * check that all required data is specified)

```
          (CLEARW promptW)
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                                  (QUOTE Host)))
          (COND
             ((NOT host)
              (PrintMsg WINDOW "Unspecified host.")
              (RETURN)))
          (SETQ defaultDirectory (LISTGET state (QUOTE DEFAULTDIRECTORY)))
          (COND
             ((EQUAL defaultDirectory "")
              (PrintMsg WINDOW "Unspecified default directory.")
              (RETURN)))
          (SETQ jobNumber (LISTGET state (QUOTE JOBNUMBER)))
          (COND
             ((EQUAL jobNumber "")
              (SETQ jobNumber NIL)
              (PrintMsg WINDOW "Unspecified job number.")
              (RETURN)))
```

(* * get the error file)

```
          (CLEARW promptW)
          (CLEARW messageW)
          (MS.CloseErrorWindow WINDOW)
          [SETQ resultFile (INFILEP (PACKFILENAME (QUOTE HOST)
                                                  (_ host Name)
                                                  (QUOTE DIRECTORY)
                                                  defaultDirectory
```

```
                                        (QUOTE NAME)
                                        jobNumber
                                        (QUOTE EXTENSION)
                                        (QUOTE RES]
          [SETQ errorFile (INFILEP (PACKFILENAME (QUOTE HOST)
                                         (_ host Name)
                                         (QUOTE DIRECTORY)
                                         defaultDirectory
                                         (QUOTE NAME)
                                         jobNumber
                                         (QUOTE EXTENSION)
                                         (QUOTE ERR]
          (COND
             (errorFile (PRIN1 "Retrieving error messages..." promptW)
                    (SETQ stream (OPENFILE resultFile (QUOTE INPUT)))
                    (SETQ result (READ stream))
                    (CLOSEF stream)
                    (DELFILE resultFile)
                    (PRIN1 result messageW)
                    (SETQ localErrorFile (COPYFILE errorFile (QUOTE {core}localErrorFile)))
                    (DELFILE errorFile)
                    (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Batch Errors"))
                    (_ host PutErrorInWindow localErrorFile errorWindow WINDOW))
             (T (PRIN1 "No batch errors found" promptW])
```

(MS.BatchLog

```
  [LAMBDA (ITEM WINDOW BUTTONS)                                    (* DSB "15-Aug-86 18:07")
                                                                   (* opens a window to display the batch log file)
    (PROG ((state (FM.READSTATE WINDOW))
           (promptW (GETPROMPTWINDOW WINDOW))
           (messageW (MS.GetMessageWindow WINDOW))
           filename host logfile localLogfile errorWindow)

          (* * check that all required data is specified)

          (SETQ filename (LISTGET state (QUOTE FILENAME)))
          (COND
             ((EQUAL filename "")
              (PrintMsg WINDOW "Unspecified file name.")
              (RETURN)))
          (SETQ host (FM.ITEMPROP (FM.ITEMFROMID WINDOW (QUOTE HOST))
                           (QUOTE Host)))
          (COND
             ((NOT host)
              (PrintMsg WINDOW "Unspecified host.")
              (RETURN)))

          (* * get the logfile)

          (CLEARW promptW)
          (CLEARW messageW)
          (MS.CloseErrorWindow WINDOW)
          [SETQ logfile (INFILEP (PACKFILENAME (QUOTE HOST)
                                         (_ host Name)
                                         (QUOTE DIRECTORY)
                                         (_ host UserDirectory)
                                         (QUOTE BODY)
                                         filename
                                         (QUOTE EXTENSION)
                                         (QUOTE LOG]
          (COND
             (logfile (SETQ localLogfile (COPYFILE logfile (QUOTE {core}localLogfile)))
                    (DELFILE logfile)
                    (PRIN1 "fetching Batch Log File..." promptW)
                    (SETQ errorWindow (MS.AttachErrorWindow WINDOW "Batch Log File"))
                    (_ host PutErrorInWindow localLogfile errorWindow WINDOW))
             (T (PRIN1 "Batch Log File not found" promptW])

)


          (* * Icon BITMAPS)

(RPAQ MS.Icon (READBITMAP))


(78 74 "OOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOL" "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L"
    "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L"
    "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L"
    "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@O@@@@@@@@L" "L@@@@@@@@I@GH@@@@@@L"
    "L@@@@@@B@O@DHGH@@@@L" "L@@@@@@B@O@@HDH@@@@L" "L@@@@@@B@I@AHGH@@@@L" "L@@@@@@B@O@C@GH@@@@L"
    "L@@@@@@B@@@GLDH@@@@L" "L@@@@@@B@@@@@GH@@@@L" "L@@@@@@@@@@D@@@D@@@L" "L@@@@@@@@@@F@@@L@@@L"
    "L@@@@CN@@@@C@DAH@@@L" "L@@@@@F@@@@A@DA@@@@L" "L@@@@@L@@@@AHDC@@@@L" "L@@@@AH@@@@@HDB@@@@L"
    "L@@@CA@@@@@LDF@@@@L" "L@@@CC@@@@@D@D@@@@L" "L@@@@@@@@@@D@D@CO@L" "L@@@@@@@@@@FNL@CO@L"
    "L@@@GH@@@@@BJH@AH@L" "L@@@DH@@@@@BJH@AH@L" "L@@@@H@@@COOOOOOIH@L" "L@@@AH@@@B@@@@@@IH@L"
    "L@@@C@@@@B@@@@@@IH@L" "L@@@GL@@@B@@@@@@IH@L" "L@@@@@@@@B@D@@@@IH@L" "L@@@@@@@@B@D@@L@IH@L"
```

```
    "L@@@@@@@@B@D@@@@IH@L" "L@COOOH@@BCOHCO@IH@L" "L@COOOH@@B@D@@@@IH@L" "L@CH@CH@@B@D@@L@IH@L"
    "L@C@@AH@@B@D@@@@OH@L" "L@C@@AH@@B@@@@@@OH@L" "L@C@@AH@@B@@@@@@H@@L" "L@C@@AH@@B@@@@@@H@@L"
    "L@C@@AH@@BBDH@@@H@@L" "L@C@@AH@@BAE@@@@H@@L" "L@C@@AH@@B@N@@@@H@@L" "L@C@@AH@@BCOHCO@H@@L"
    "L@C@@AH@@B@N@@@@H@@L" "L@CH@CKFMJAE@@@@H@@L" "L@COOOH@@BBDH@@@H@@L" "L@COOOH@@B@@@@@@H@@L"
    "L@@AO@@@@COOOOOOH@@L" "LAOOOON@@@@@@@@@@@@L" "LA@@@@B@@@@@@@@@@@@L" "LA@@@@B@@@@@@@@@@@@L"
    "LA@@@@B@@@@@@@@@@@@L" "LAOOOON@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L"
    "LC@HNGLICLOCNDBOCN@L" "LCOIOA@IB@HBBDBHBB@L" "LBNIAA@ICLNCNFFNCN@L" "LBDIOA@O@DHCHCLHCH@L"
    "LB@IAA@IBDHBLAHHBL@L" "LB@IAA@ICLOBFAHOBF@L" "L@@@@@@@@@@@@@@@@@@L" "L@@@@@@@@@@@@@@@@@@L"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL")
```

(RPAQ **MS.IconMask** (READBITMAP))

```
(78 74 "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOH@COOOOOOOOOOOOOL"
    "OOO@@AOOOOOOOOOOOOOL" "OOO@@AOOOOOOOOOOOOOL" "OOO@@AOOOOOOOOOOOOOL" "OOO@@AOOOOOOOOOOOOOL"
    "OOO@@AOOOOOOOOOOOOOL" "OOO@@AOOOOOOOOOOOOOL" "OOO@@AOOOOOOOOOOOOOL" "OOO@@AOOOOOOOOOOOOOL"
    "OOO@@AOOOOOOOOOOOOOL" "OOOH@COOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL"
    "OOOOOOOOOOOOOOOOOOOL" "OOOOOOOOOOOOOOOOOOOL")
```

(* * VARS)

(ADDTOVAR **BackgroundMenuCommands** (Server% Menu (QUOTE (MS.TopLevel))
"Start the Server Menu"))

(RPAQQ **BackgroundMenu** NIL)

(DECLARE: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS MS.HostPopMenu)
)

(* * FORTRAN EDIT -
Sets up a Fortran Edit Process)

(* * MAIN FUNCTIONS)

(DEFINEQ

(**FE.TopLevel**
[LAMBDA (program window) (* DSB "22-Aug-86 11:49")

(* * Edit a Fortran program using a specialized TEdit.)

(* * Program is either a File name, an open Stream, or a string.
If NIL, an empty edit window is open.)

(* * Window is an optional window to be used for editing. If NIL, the user will be asked to sweep out a window on the screen.)

(PROG (editProps editWindow messageWindow processHandle textStream)

(* * Get prop list for edit props)

(SETQ editProps (**FE.GetEditProps**))

(* * Get the window to be used)

(SETQ editWindow (**FE.GetEditWindow** window))

(* * Get the message window)

(SETQ messageWindow (**FE.GetMessageWindow** editWindow))

(* * Set up the edit process)

(SETQ processHandle (TEDIT program editWindow NIL editProps))

(* * Get the textstream parameter of the edit process)

(SETQ textStream (TEXTSTREAM editWindow))

(* * Do any final modifications)

```
        (FE.AdjustProps processHandle textStream)
        (WINDOWPROP editWindow (QUOTE FORTRANEDIT)
               T)
        (TEXTPROP textStream (QUOTE EDITWINDOW)
               editWindow)
        (TEXTPROP textStream (QUOTE MESSAGEWINDOW)
               messageWindow)
        (TEXTPROP textStream (QUOTE PROCESS)
               processHandle)
        (TEXTPROP textStream (QUOTE FE.POSITIONWINDOW)
               (WINDOWPROP editWindow (QUOTE POSITIONWINDOW)))
```

(* * Finally, return the process handle)

```
        (RETURN textStream])
```

(FE.AdjustProps

```
  [LAMBDA (processHandle textStream)                    (* DSB " 7-Jul-86 15:52")
```

(* * Do final adjustments to Fortran editor)

```
    (LET NIL (COND
                ((PROCESSP processHandle)
```

(* * The following gives the process a name)

```
                 (PROCESSPROP processHandle (QUOTE NAME)
                        (QUOTE FORTRAN% EDITOR))
```

(* * The following disables image object insertion into the document)

```
                 (until (PROCESSPROP processHandle (QUOTE TEDITTTYWINDOW)) do (BLOCK)
                    finally (WINDOWPROP (PROCESSPROP processHandle (QUOTE TEDITTTYWINDOW))
                                  (QUOTE COPYINSERTFN)
                                  NIL])
```

(FE.CaretPosition

```
  [LAMBDA (textStream)                                  (* DSB " 7-Jul-86 15:48")
```

(* * Write the line# and the column# of the position of the caret in Textstream)

```
    (PROG* (charWidth column midpoint position positionWindow textWindow (margin 8)
                  (textStream (TEXTSTREAM textStream)))
           (COND
              ([AND (TEXTSTREAMP textStream)
                    (WINDOWP (SETQ positionWindow (TEXTPROP textStream (QUOTE FE.POSITIONWINDOW]
               [SETQ textWindow (WINDOWP (CAR (LISTP (fetch (TEXTOBJ \WINDOW) of (TEXTOBJ textStream]
               [SETQ charWidth (CHARWIDTH (CHCON1 "X")
                                          (TEXTPROP textStream (QUOTE FONT]
               (COND
                  ([NOT (EQUAL (SETQ position (TEDIT.PARA&CHAR textStream))
                               (TEXTPROP textStream (QUOTE FE.POSITION]
                   (SETQ midpoint (IPLUS 3 (IQUOTIENT (IQUOTIENT (WINDOWPROP positionWindow (QUOTE WIDTH))
                                                                 (CHARWIDTH (CHARCODE X)
                                                                        (DSPFONT NIL positionWindow)))
                                                      2)))
                   (SETQ column (IPLUS (QUOTIENT (IDIFFERENCE (DSPXPOSITION NIL textWindow)
                                                              margin)
                                                 charWidth)
                                       1))
                   (CLEARW positionWindow)
                   [printout positionWindow .TAB0 0 (COND
                                                       ((MINUSP (IDIFFERENCE column (CDR position)))
                                                        " ? ")
                                                       (T ""))
                          .CENTER midpoint (CONCAT "L: " (CAR position))
                          .TAB0 midpoint .CENTER 0 (CONCAT "C: " (COND
                                                                    ((NOT (ZEROP charWidth))
                                                                     column)
                                                                    (T -1]
                   (TEXTPROP textStream (QUOTE FE.POSITION)
                          position)))
               (RETURN position])
```

(TEDIT.PARA&CHAR

```
  [LAMBDA (TEXTSTREAM SEL)                              (* RAR " 9-Oct-85 15:49")
```

(* * Given a text stream, (and optionally a selection) find the pagagraph # and ch within paragraph that the caret is at)

```
    (PROG (CH# PC PCTB (TEXTOBJ (TEXTOBJ TEXTSTREAM))
          (PARA# 1)
          (CHAR# 1)
          (LASTPARACH# 1))
```

```
          (SETQ PCTB (fetch PCTB of TEXTOBJ))
          (SETQ PC (ELT PCTB 3))
          (SETQ CH# (TEDIT.GETPOINT TEXTOBJ SEL))
          [COND
             ((ZEROP (fetch TEXTLEN of TEXTOBJ))
              (RETURN (CONS 1 0]
          (RETURN (while PC do [COND
                                  [(IGREATERP (add CHAR# (fetch PLEN of PC))
                                          CH#)                    (* Passed the relevant char; return a result)
                                   (RETURN (CONS PARA# (IDIFFERENCE CH# LASTPARACH#]
                                  (T                              (* Not past the caret; keep going)
                                     (COND
                                        ((fetch PPARALAST of PC)  (* Crossing a paragraph boundary.
                                                                  Count (QUOTE em) up.)
                                         (add PARA# 1)
                                         (SETQ LASTPARACH# CHAR#]
                                (SETQ PC (fetch NEXTPIECE of PC))
                   finally (RETURN (CONS PARA# (IDIFFERENCE CH# LASTPARACH#])
```

(FE.CharFn

```
  [LAMBDA (textObj charCode)                                      (* DSB " 7-Jul-86 15:51")
```

(* * This function filters out the effects of someone trying to alter the "LOOKS" of something in the FORTRANEDITOR)

```
    (COND
       ((ILEQ charCode 127)
        charCode])
```

(FE.GetEditProps

```
  [LAMBDA NIL                                                     (* DSB "20-Aug-86 14:19")
```

(* * Return a prop list for TEdit call)

```
    (PROG (charWidth (font FE.defaultFont))
          (SETQ charWidth (CHARWIDTH (CHCON1 "X")
                                     font))
          (RETURN (APPEND (QUOTE (CLEARGET T))
                          (QUOTE (CLEARPUT T))
                          (LIST (QUOTE FONT)
                                font)
                          (LIST (QUOTE MENU)
                                (FE.CreateLocalMenu))
                          (LIST (QUOTE PARALOOKS)
                                (LIST (QUOTE TABS)
                                      (CONS (ITIMES 8 charWidth)
                                            NIL)))
                          (LIST (QUOTE LOOPFN)
                                (FUNCTION FE.LoopFn))
                          (LIST (QUOTE CHARFN)
                                (FUNCTION FE.CharFn))
                          (QUOTE (COPYBYBKSYSBUF T))
                          (LIST (QUOTE READTABLE)
                                (PROG ((table (COPYREADTABLE TEDIT.READTABLE)))
```

(* * Return the read table to be used with this process)

```
                                      (TEDIT.SETSYNTAX 15 NIL table)
                                                                  (* Turns of inserting with CTRL-O)
                                      (RETURN table])
```

(FE.GetSourceFileName

```
  [LAMBDA (textObj)                                               (* DSB "22-Aug-86 14:08")
```

(* * Return filename associated with textObj)

(* * Due to a TEDIT bug, we can't use (FULLNAME (fetch TXTFILE of textObj)), because this can be changed when a file is opened to the same non-leaf server. Thus, we use the same FILENAME textprop that FE.MyPut uses and FE.MyGet updates.)

```
    (PROG (fileStream textStream filename messageWindow promptWindow dirty)
```

(* * Make sure we have a text object)

```
          (COND
             ((NOT (TYPENAMEP textObj (QUOTE TEXTOBJ)))
              (RETURN NIL)))
```

(* * See if the file is ready to use)

```
          (SETQ fileStream (fetch TXTFILE of textObj))
          (SETQ textStream (TEXTSTREAM textObj))
          (SETQ filename (TEXTPROP textStream (QUOTE FILENAME)))
          (SETQ messageWindow (TEXTPROP textStream (QUOTE MESSAGEWINDOW)))
          (CLEARW messageWindow)
```

```
          (SETQ promptWindow (fetch PROMPTWINDOW of textObj))
          (CLEARW promptWindow)
          (SETQ dirty (TEDIT.STREAMCHANGEDP textStream))
          (COND
             ((NOT dirty)
              (RETURN filename)))
          (COND
             ((AND dirty (NOT fileStream))
              (PRIN1 "Can't.  No file has been saved yet." promptWindow)
              (RETURN NIL)))
          (COND
             ((AND dirty fileStream)
              (COND
                 ((MOUSECONFIRM "Not saved yet; LEFT to use previous version." T promptWindow)
                  (PRIN1 "Using old version on the server" messageWindow)
                  (RETURN filename))
                 (T (RETURN NIL])
```

(FE.LoopFn

```
  [LAMBDA (textStream)                                    (* DSB " 7-Jul-86 17:08")

          (* * Things to be done each time around TEdit command loop)

    (LET NIL

          (* * Shade the edit window)

         (FE.ShadeWindow textStream)

          (* * Update the position display)

         (FE.CaretPosition textStream])

)


          (* * WINDOW FUNCTIONS)

(DEFINEQ
```

(FE.GetPositionWindow

```
  [LAMBDA (mainWindow)                                    (* DSB " 7-Jul-86 17:13")

          (* * Return the window to be used as the caret-position indicator window)

    (PROG (height positionWindow (font FE.defaultFont))
          [SETQ height (HEIGHTIFWINDOW (FONTPROP font (QUOTE HEIGHT]
          (SETQ positionWindow (CREATEW (CREATEREGION 0 0 (WIDTHIFWINDOW (WINDOWPROP mainWindow (QUOTE WIDTH)))
                                                      height)
                                        NIL NIL T))
          (DSPFONT font positionWindow)
          (DSPTEXTURE BLACKSHADE positionWindow)
          (DSPOPERATION (QUOTE INVERT)
                 positionWindow)
          (WINDOWPROP positionWindow (QUOTE CLOSEFN)
                 (QUOTE (DETACHWINDOW)))
          (WINDOWPROP positionWindow (QUOTE MINSIZE)
                 (CONS 0 height))
          (WINDOWPROP positionWindow (QUOTE MAXSIZE)
                 (CONS 64000 height))
          (WINDOWPROP positionWindow (QUOTE PAGEFULLFN)
                 (FUNCTION NILL))
          (RETURN positionWindow])
```

(FE.GetEditWindow

```
  [LAMBDA (window)                                        (* DSB "15-Aug-86 09:19")

          (* * Return a window to be used by the Fortran editor)

    (PROG (fontHeight minWidth minHeight positionWindow (font FE.defaultFont))

          (* * Set the minimum window dimensions to be 20 characters wide by four lines high)

          [SETQ minWidth (WIDTHIFWINDOW (ITIMES 20 (CHARWIDTH (CHCON1 "X")
                                                              font]
          (SETQ minHeight (HEIGHTIFWINDOW [ITIMES 4 (SETQ fontHeight (FONTPROP font (QUOTE HEIGHT]
                                  T))

          (* * If not passed a window, then create one)

          [COND
             ((NOT (WINDOWP window))
              (SETQ window (CREATEW (GETREGION minWidth minHeight)
                                    (QUOTE Fortran% Editor)
                                    NIL T]
```

```
        (* * Add our window properties)

        (WINDOWADDPROP window (QUOTE RESHAPEFN)
               (FUNCTION FE.ReshapeFn))
        (WINDOWPROP window (QUOTE ICONFN)
               (FUNCTION FE.ShrinkIconCreate))
        (WINDOWPROP window (QUOTE MINSIZE)
               (CONS minWidth minHeight))

        (* * Now add a window for displaying the caret position)

        (SETQ positionWindow (FE.GetPositionWindow window))
        (ATTACHWINDOW positionWindow window (QUOTE TOP)
               (QUOTE JUSTIFY))
        (WINDOWPROP positionWindow (QUOTE PASSTOMAINCOMS)
               T)

        (* needed due to bug in ATTACHWINDOW which does not set the prop correctly when WINDOWCOMACTION=MAIN)

        (WINDOWPROP window (QUOTE POSITIONWINDOW)
               positionWindow)

        (* * Return the main window)

        (RETURN window])
```

(FE.GetMessageWindow

```
  [LAMBDA (editWindow)                                    (* DSB "19-Aug-86 17:55")
```

```
        (* * Create, attach, and return the messageWindow on the bottom of the editWindow)

    (PROG (messageWindow)
          (SETQ messageWindow (CREATEW (CREATEREGION 0 0 200 60)
                                       "Message Window" NIL T))
          (ATTACHWINDOW messageWindow editWindow (QUOTE BOTTOM)
                 (QUOTE JUSTIFY))
          (RETURN messageWindow])
```

(FE.ReshapeFn

```
  [LAMBDA (window oldImage oldRegion)                     (* DSB " 7-Jul-86 17:23")
```

```
        (* * Need to set the TEXTPROP FE.POSITION to Nil to force position update following reshape of main window)

    (TEXTPROP (TEXTSTREAM window)
           (QUOTE FE.POSITION)
           NIL])
```

(FE.ShadeWindow

```
  [LAMBDA (stream)                                        (* DSB " 7-Jul-86 17:28")
```

```
        (* * Highlight the sixth and seventy-third columns of the editor window)

    (PROG (charWidth height window (margin 8)
                 (textObj (TEXTOBJ stream)))
          (SETQ window (CAR (fetch (TEXTOBJ \WINDOW) of textObj)))
          [SETQ charWidth (CHARWIDTH (CHCON1 "X")
                                     (TEXTPROP textObj (QUOTE FONT]
          (SETQ height (WINDOWPROP window (QUOTE HEIGHT)))
          (BITBLT NIL NIL NIL window (IPLUS margin (ITIMES 5 charWidth))
                 0 1 height (QUOTE TEXTURE)
                 (QUOTE REPLACE)
                 GRAYSHADE)
          (BITBLT NIL NIL NIL window (SUB1 (IPLUS margin (ITIMES 6 charWidth)))
                 0 1 height (QUOTE TEXTURE)
                 (QUOTE REPLACE)
                 GRAYSHADE)
          (BITBLT NIL NIL NIL window (IPLUS margin (ITIMES 72 charWidth))
                 0 1 height (QUOTE TEXTURE)
                 (QUOTE REPLACE)
                 GRAYSHADE])

)
```

```
        (* * LOCALMENU FUNCTIONS)

(DEFINEQ
```

(FE.CreateLocalMenu

```
  [LAMBDA NIL                                             (* DSB " 7-Nov-86 09:42")
```

```
        (* * Return the local menu that pops up when the left or middle buttons are pressed when the mouse pointer is in the title bar
        area of the Fortran editor window.)
```

```
(create MENU
       ITEMS _ (QUOTE (Quit Hardcopy (Put (FUNCTION FE.MyPut)
                                          "Write edit buffer to specified file")
                                   (Get (FUNCTION FE.MyGet)
                                        "Replace contents of edit buffer with contents of specified file")
                                   (Include (QUOTE Include)
                                          "Add contents of specified file to edit buffer at present location")
                                   (Find (QUOTE Find)
                                         "Find first occurence of specified string beyond present location")
                                   (Substitute (QUOTE Substitute)
                                          "Replace all occurances of specified string with new string in selected
                                          text")
                                   (Host (FUNCTION FE.SetHost)
                                         "Declare host server")
                                   (Directory (FUNCTION FE.SetDirectory)
                                          "Declare host directory")
                                   (Compile (FUNCTION FE.Compile)
                                          "Compile file on host")
                                   (Link (FUNCTION FE.Link)
                                         "Link file on host")
                                   (C/L/R (FUNCTION FE.CompileLinkRun)
                                          "Compile,link and run file on host")
                                   (Run (FUNCTION FE.RunInteractive)
                                        "Run file on host")))
       CENTERFLG _ T
       MENUFONT _ (FONTCREATE (QUOTE HELVETICA)
                              10
                              (QUOTE BOLD))
       WHENSELECTEDFN _ (FUNCTION \TEDIT.MENU.WHENSELECTEDFN)
       WHENHELDFN _ (FUNCTION \TEDIT.MENU.WHENHELDFN])
```

(FE.SetHost

```
[LAMBDA (textStream)                                                    (* DSB "22-Aug-86 13:07")
```

(* * Ask user to declare a host server, using the present host server as a default answer.)

(* * the host, hostname and default directory are all calculated and stored as TEXTPROPs)

(* * Note that FE.GetServer returns the pointer to the server instance.
When the message is sent to host, host is evaled.)

(* * Note also that hostname must be an upper-case atom when it is passed to FE.ValidHostname and FE.GetServer)

```
(PROG [messageWindow host hostname (oldHostname (TEXTPROP textStream (QUOTE MS.HOSTNAME]
       (SETQ messageWindow (TEXTPROP textStream (QUOTE MESSAGEWINDOW)))
       (CLEARW messageWindow)
       (COND
          ([FE.ValidHostname (SETQ hostname (U-CASE (MKATOM (TEDIT.GETINPUT textStream "Hostname :"
                                                                            oldHostname]
           (TEXTPROP textStream (QUOTE MS.HOSTNAME)
                  hostname)
           (SETQ host (FE.GetServer hostname))
           (TEXTPROP textStream (QUOTE MS.HOST)
                  host)
           (TEXTPROP textStream (QUOTE MS.DIRECTORY)
                  (_ host UserDirectory))
           (TEDIT.PROMPTPRINT textStream (CONCAT "Hostname is " hostname)
                  T))
          (T (TEDIT.PROMPTPRINT textStream (CONCAT hostname " is not valid...Hostname unchanged")
                    T)))
       (RETURN NIL])
```

(FE.SetDirectory

```
[LAMBDA (textStream)                                                    (* DSB "22-Aug-86 13:09")
```

(* * ask user to declare a default directory, using the previously defined directory as a default.
The default is initially set to the user's root directory when the host is declared.)

```
(PROG [messageWindow newDirectory (oldDirectory (TEXTPROP textStream (QUOTE MS.DIRECTORY)))
              (host (TEXTPROP textStream (QUOTE MS.HOST]
```

(* * first clear the message window)

```
(SETQ messageWindow (TEXTPROP textStream (QUOTE MESSAGEWINDOW)))
(CLEARW messageWindow)
```

(* * then make sure a host has been declared)

```
[COND
   ((NOT host)
    (RETURN (TEDIT.PROMPTPRINT textStream "No host has yet been declared.  Name your host first." T]
```

(* * then reset directory if changed)

```
          (SETQ newDirectory (MKATOM (TEDIT.GETINPUT textStream "Directory: " oldDirectory)))
          (COND
             ((EQUAL newDirectory oldDirectory)
              (TEDIT.PROMPTPRINT textStream (CONCAT newDirectory " is the same as the previous value...Directory
                                                   unchanged")
                     T))
             (T (TEXTPROP textStream (QUOTE MS.DIRECTORY)
                       newDirectory)
                (TEDIT.PROMPTPRINT textStream (CONCAT "Directory is " newDirectory)
                       T])
```

(FE.MyGet

```
  [LAMBDA (textStream)                                    (* DSB "22-Aug-86 12:59")
                                                          (* My TEDIT Get Function)
```

(* * after getting the file, it sets the FILENAME textprop to the new fullFilename.
This textprop is only changed by a Get, whereas due to an error in TEDIT, the TXTFILE slot of textObj can change whenever an OPENFILE is made to the server)

(* * Note that we only store the versionless filename, because it then gets updated properly on a Put.)

```
    (PROG (messageWindow textObj fileStream fullFilename versionlessFilename)
          (SETQ messageWindow (TEXTPROP textStream (QUOTE MESSAGEWINDOW)))
          (CLEARW messageWindow)
          (SETQ textObj (TEXTOBJ textStream))
          (TEDIT.GET textObj)
          (SETQ fileStream (fetch TXTFILE of textObj))
          (SETQ fullFilename (FULLNAME fileStream))
          (SETQ versionlessFilename (FE.StripVersion fullFilename))
          (TEXTPROP textStream (QUOTE FILENAME)
                 versionlessFilename])
```

(FE.MyPut

```
  [LAMBDA (textStream)                                    (* DSB " 7-Nov-86 11:03")
                                                          (* my TEDIT put function)
```

(* * When the edit buffer is to be saved for the first time, the FILENAME TEXTPROP is NIL.
Subsequently, it has a (versionless) value, which remains the same if put the the same filename or is altered if put to a different filename)

(* * The new filename is stored without version number in the FILENAME field of TEXTPROP)

(* * The reason for this stuff is that the name in (FULLNAME (fetch TXTFILE of textObj)) can be altered by an OPENFILE, when both files are on a non-leaf server. Thus, we have to keep track of the TEDIT filename ourselves.)

```
    (PROG (messageWindow newFilename textObj fileStream fullFilename versionlessFilename)
```

(* * first clear the message window)

```
          (SETQ messageWindow (TEXTPROP textStream (QUOTE MESSAGEWINDOW)))
          (CLEARW messageWindow)
```

(* * get the filename to be put. If the user inputs a c.r., it returns the old filename;
otherwise, it retains the user input.)

```
          (SETQ oldFilename (TEXTPROP textStream (QUOTE FILENAME)))
          [SETQ newFilename (MKATOM (U-CASE (TEDIT.GETINPUT textStream "(Put) Filename:" oldFilename]
```

(* * put the returned filename)

```
          (TEDIT.PUT textStream newFilename NIL T)
```

(* * a new FILENAME textprop is saved only when the new filename differs from the old filename)

```
          (COND
             ((EQUAL oldFilename newFilename)
              (RETURN))
             (T (SETQ textObj (TEXTOBJ textStream))
                (SETQ fileStream (fetch TXTFILE of textObj))
                (SETQ fullFilename (FULLNAME fileStream))
                (SETQ versionlessFilename (FE.StripVersion fullFilename))
                (TEXTPROP textStream (QUOTE FILENAME)
                       versionlessFilename])
```

(FE.StripVersion

```
  [LAMBDA (fullFilename)                                  (* DSB "20-Aug-86 13:50")
                                                          (* returns a filename with the version stripped number out)
    (PROG (host directory name extension)
          (SETQ host (UNPACKFILENAME fullFilename (QUOTE HOST)))
          (SETQ directory (UNPACKFILENAME fullFilename (QUOTE DIRECTORY)))
          (SETQ name (UNPACKFILENAME fullFilename (QUOTE NAME)))
          (SETQ extension (UNPACKFILENAME fullFilename (QUOTE EXTENSION)))
          (RETURN (PACKFILENAME (QUOTE HOST)
                        host
```

```
                        (QUOTE DIRECTORY)
                        directory
                        (QUOTE NAME)
                        name
                        (QUOTE EXTENSION)
                        extension])
```

(FE.Compile

```
  [LAMBDA (textStream)                                    (* DSB "22-Aug-86 11:54")
```

(* * Compile the file associated with this edit process.)

```
    (PROG (fullFilename filename directory host result errorFile errorWindow editWindow messageWindow
                 (textObj (TEXTOBJ textStream)))
          (SETQ messageWindow (TEXTPROP textStream (QUOTE MESSAGEWINDOW)))
          (CLEARW messageWindow)
```

(* * Make sure we have a text object)

```
          (COND
             ((NOT (TYPENAMEP textObj (QUOTE TEXTOBJ)))
              (PRIN1 "No text to compile" messageWindow)
              (RETURN NIL)))
```

(* * See if the file is ready to use)

```
          (COND
             ((NOT (SETQ fullFilename (FE.GetSourceFileName textObj)))
              (PRIN1 "Text non-existant or unsaved:  compile aborted" messageWindow)
              (RETURN NIL)))
```

(* * Check that a host has been specified)

```
          (COND
             ([NOT (SETQ host (TEXTPROP textObj (QUOTE MS.HOST]
              (PRIN1 "Can't compile.  No Host has been declared" messageWindow)
              (RETURN NIL)))
```

(* * Do the compile)

```
          (TEDIT.PROMPTPRINT textStream (CONCAT "Compiling " fullFilename " on " (_ host Name)
                                                "...")
                 T)
          (SETQ editWindow (TEXTPROP textStream (QUOTE EDITWINDOW)))
          (SETQ directory (TEXTPROP textObj (QUOTE MS.DIRECTORY)))
          (SETQ filename (UNPACKFILENAME fullFilename (QUOTE NAME)))
          (MS.CloseErrorWindow editWindow)
          (SETQ result (_ host Compile (PACKFILENAME (QUOTE DIRECTORY)
                                                     directory
                                                     (QUOTE NAME)
                                                     filename)))
          (CLEARW messageWindow)
          (PRIN1 result messageWindow)
          (COND
             ((EQUAL (CAR result)
                     (QUOTE ERROR))
              (TEDIT.PROMPTPRINT textStream "Compilation errors" T)
              (SETQ errorWindow (MS.AttachErrorWindow editWindow "Compilation Errors"))
              (SETQ errorFile (_ host ExtractFilename result))
              (_ host PutErrorInWindow errorFile errorWindow editWindow))
             (T (TEDIT.PROMPTPRINT textStream "Successful compilation." T])
```

(FE.Link

```
  [LAMBDA (textStream)                                    (* DSB "22-Aug-86 12:01")
```

(* * Link the file associated with this edit process.)

```
    (PROG (fullFilename filename directory host result errorFile errorWindow editWindow messageWindow
                 (textObj (TEXTOBJ textStream)))
          (SETQ messageWindow (TEXTPROP textStream (QUOTE MESSAGEWINDOW)))
          (CLEARW messageWindow)
```

(* * Make sure we have a text object)

```
          (COND
             ((NOT (TYPENAMEP textObj (QUOTE TEXTOBJ)))
              (PRIN1 "No file to link" messageWindow)
              (RETURN NIL)))
```

(* * See if the file is ready to use)

```
          (COND
             ((NOT (SETQ fullFilename (FE.GetSourceFileName textObj)))
              (PRIN1 "File non-existant or unsaved:  link aborted" messageWindow)
              (RETURN NIL)))
```

```
(* * Check that a host has been specified)

(COND
   ([NOT (SETQ host (TEXTPROP textObj (QUOTE MS.HOST]
    (PRIN1 "Can't link.  No Host has been declared" messageWindow)
    (RETURN NIL)))

(* * Do the link)

(TEDIT.PROMPTPRINT textStream (CONCAT "Linking " fullFilename " on " (_ host Name)
                                      "...")
       T)
(SETQ editWindow (TEXTPROP textStream (QUOTE EDITWINDOW)))
(SETQ directory (TEXTPROP textObj (QUOTE MS.DIRECTORY)))
(SETQ filename (UNPACKFILENAME fullFilename (QUOTE NAME)))
(MS.CloseErrorWindow editWindow)
(SETQ result (_ host Link (PACKFILENAME (QUOTE DIRECTORY)
                                  directory
                                  (QUOTE NAME)
                                  filename)))
(CLEARW messageWindow)
(PRIN1 result messageWindow)
(COND
   ((EQUAL (CAR result)
           (QUOTE ERROR))
    (TEDIT.PROMPTPRINT textStream "Linking error" T)
    (SETQ errorWindow (MS.AttachErrorWindow editWindow "Link Errors"))
    (SETQ errorFile (_ host ExtractFilename result))
    (_ host PutErrorInWindow errorFile errorWindow editWindow))
   (T (TEDIT.PROMPTPRINT textStream "Successful link" T])
```

(FE.CompileLinkRun

```
[LAMBDA (textStream)                                    (* DSB "22-Aug-86 11:54")

(* * sequentially compiles, links, and runs the file associated with this edit process.)

  (PROG (fullFilename filename directory host result errorFile errorWindow editWindow messageWindow
         (textObj (TEXTOBJ textStream)))
(SETQ messageWindow (TEXTPROP textStream (QUOTE MESSAGEWINDOW)))
(CLEARW messageWindow)

(* * Make sure we have a text object)

(COND
   ((NOT (TYPENAMEP textObj (QUOTE TEXTOBJ)))
    (PRIN1 "No text to compile" messageWindow)
    (RETURN NIL)))

(* * See if the file is ready to use)

(COND
   ((NOT (SETQ fullFilename (FE.GetSourceFileName textObj)))
    (PRIN1 "Text non-existant or unsaved:  compile aborted" messageWindow)
    (RETURN NIL)))

(* * Check that a host has been specified)

(COND
   ([NOT (SETQ host (TEXTPROP textObj (QUOTE MS.HOST]
    (PRIN1 "Can't compile.  No Host has been declared" messageWindow)
    (RETURN NIL)))

(* * Do the compile)

(TEDIT.PROMPTPRINT textStream (CONCAT "Compiling " fullFilename " on " (_ host Name)
                                      "...")
       T)
(SETQ editWindow (TEXTPROP textStream (QUOTE EDITWINDOW)))
(SETQ directory (TEXTPROP textObj (QUOTE MS.DIRECTORY)))
(SETQ filename (UNPACKFILENAME fullFilename (QUOTE NAME)))
(MS.CloseErrorWindow editWindow)
(SETQ result (_ host Compile (PACKFILENAME (QUOTE DIRECTORY)
                                  directory
                                  (QUOTE NAME)
                                  filename)))
(CLEARW messageWindow)
(PRIN1 result messageWindow)
(COND
   ((EQUAL (CAR result)
           (QUOTE ERROR))
    (TEDIT.PROMPTPRINT textStream "Compilation errors" T)
    (SETQ errorWindow (MS.AttachErrorWindow editWindow "Compilation Errors"))
    (SETQ errorFile (_ host ExtractFilename result))
    (_ host PutErrorInWindow errorFile errorWindow editWindow)
    (RETURN))
```

```
            (T (TEDIT.PROMPTPRINT textStream "Successful compilation." T)))

         (* * Do the link)

         (TEDIT.PROMPTPRINT textStream (CONCAT "Linking " fullFilename " on " (_ host Name)
                                               "...")
                T)
         (MS.CloseErrorWindow editWindow)
         (SETQ result (_ host Link (PACKFILENAME (QUOTE DIRECTORY)
                                                 directory
                                                 (QUOTE NAME)
                                                 filename)))
         (CLEARW messageWindow)
         (PRIN1 result messageWindow)
         (COND
            ((EQUAL (CAR result)
                    (QUOTE ERROR))
             (TEDIT.PROMPTPRINT textStream "Linking error" T)
             (SETQ errorWindow (MS.AttachErrorWindow editWindow "Link Errors"))
             (SETQ errorFile (_ host ExtractFilename result))
             (_ host PutErrorInWindow errorFile errorWindow editWindow)
             (RETURN))
            (T (TEDIT.PROMPTPRINT textStream "Successful link" T)))

         (* * Run the job interactively)

         (TEDIT.PROMPTPRINT textStream (CONCAT "Running interactive job " fullFilename " on " (_ host Name)
                                               "...")
                T)
         (MS.CloseErrorWindow editWindow)
         (SETQ result (_ host RunJob (PACKFILENAME (QUOTE DIRECTORY)
                                                   directory
                                                   (QUOTE NAME)
                                                   filename)))
         (CLEARW messageWindow)
         (PRIN1 result messageWindow)
         (COND
            ((EQUAL (CAR result)
                    (QUOTE ERROR))
             (TEDIT.PROMPTPRINT textStream "Run-time warning or error" T)
             (SETQ errorWindow (MS.AttachErrorWindow editWindow "Run-time Errors"))
             (SETQ errorFile (_ host ExtractFilename result))
             (_ host PutErrorInWindow errorFile errorWindow editWindow))
            (T (TEDIT.PROMPTPRINT textStream "Done" T])
```

(FE.RunInteractive

```
  [LAMBDA (textStream)                                    (* DSB "22-Aug-86 12:49")

         (* * Run (interactively) the file associated with this edit process.)

    (PROG (fullFilename filename directory host result errorFile errorWindow editWindow messageWindow
                 (textObj (TEXTOBJ textStream)))
         (SETQ messageWindow (TEXTPROP textStream (QUOTE MESSAGEWINDOW)))
         (CLEARW messageWindow)

         (* * Make sure we have a text object)

         (COND
            ((NOT (TYPENAMEP textObj (QUOTE TEXTOBJ)))
             (PRIN1 "No file to run" messageWindow)
             (RETURN NIL)))

         (* * See if the file is ready to use)

         (COND
            ((NOT (SETQ fullFilename (FE.GetSourceFileName textObj)))
             (PRIN1 "File non-existant or unsaved:  run aborted" messageWindow)
             (RETURN NIL)))

         (* * Check that a host has been specified)

         (COND
            ([NOT (SETQ host (TEXTPROP textObj (QUOTE MS.HOST]
             (PRIN1 "Can't run.  No Host has been declared" messageWindow)
             (RETURN NIL)))

         (* * Run it)

         (TEDIT.PROMPTPRINT textStream (CONCAT "Running interactive job " fullFilename " on " (_ host Name)
                                               "...")
                T)
         (SETQ editWindow (TEXTPROP textStream (QUOTE EDITWINDOW)))
         (SETQ directory (TEXTPROP textObj (QUOTE MS.DIRECTORY)))
         (SETQ filename (UNPACKFILENAME fullFilename (QUOTE NAME)))
         (MS.CloseErrorWindow editWindow)
         (SETQ result (_ host RunJob (PACKFILENAME (QUOTE DIRECTORY)
```

```
                                          directory
                                          (QUOTE NAME)
                                          filename)))
          (CLEARW messageWindow)
          (PRIN1 result messageWindow)
          (COND
             ((EQUAL (CAR result)
                     (QUOTE ERROR))
              (TEDIT.PROMPTPRINT textStream "Run-time warning or error" T)
              (SETQ errorWindow (MS.AttachErrorWindow editWindow "Run-time Errors"))
              (SETQ errorFile (_ host ExtractFilename result))
              (_ host PutErrorInWindow errorFile errorWindow editWindow))
             (T (TEDIT.PROMPTPRINT textStream "Done" T])

)


          (* * SERVER METACLASS FUNCTIONS)

(DEFINEQ

(FE.ValidHostname
  [LAMBDA (hostname)                                   (* DSB "20-Aug-86 08:40")
                                                       (* returns the hostname if it is on the list of valid hosts)

          (* * Note that hostname must be passed from FE.SetHost as an upper-case atom)

    (PROG (server validHostname)
          (SETQ server (FE.GetServer hostname))
          (COND
             (server (SETQ validHostname (_ server Name))
                     (RETURN validHostname))
             (T NIL])

(FE.GetServer
  [LAMBDA (hostname)                                   (* DSB "20-Aug-86 08:42")
                                                       (* given a hostname, returns the pointer to the server)

          (* * note that the hostname must be an upper-case atom)

    (find server in (_ ($ Server)
                       AllInstances!)
       suchthat (EQUAL hostname (_ server Name])

)


          (* * ICON STUFF)

(DEFINEQ

(FE.ShrinkIconCreate
  [LAMBDA (W ICON ICONW)                               (* DSB " 6-Oct-86 13:52")
                                                       (* Create the icon that represents this window.)
    [PROG [(icon (WINDOWPROP W (QUOTE ICON)))
           (iconTitle (WINDOWPROP W (QUOTE TEDIT.ICON.TITLE)))
           (shrinkfn (WINDOWPROP W (QUOTE SHRINKFN]
          (COND
             ((NOT (WINDOWPROP W (QUOTE TEXTOBJ)))     (* This isn't really a TEdit window any more.
                                                       Don't do anything)
              NIL)
             ((WINDOWPROP W (QUOTE TEDITMENU))         (* This is a text menu, and shrinks without trace.)
              NIL)
             ((OR (IGREATERP (FLENGTH shrinkfn)
                             3)
                  (AND (NOT (FMEMB (QUOTE SHRINKATTACHEDWINDOWS)
                                   shrinkfn))
                       (IGREATERP (FLENGTH shrinkfn)
                                  2)))                 (* There are other functions that expect to handle this.
                                                       Don't bother.)
              NIL)
             ((OR [AND iconTitle (EQUAL iconTitle (TEXTSTREAM.TITLE (TEXTSTREAM W]
                  (AND (NOT iconTitle)
                       icon))

          (* we built this and the title is the same, or he has already put an icon on this.
          Do nothing)

              NIL)
             (icon                                     (* There's an existing icon window;
                                                       change the title in it)
                   [WINDOWPROP W (QUOTE TEDIT.ICON.TITLE)
                          (SETQ iconTitle (TEXTSTREAM.TITLE (TEXTSTREAM W]
                   (ICONTITLE iconTitle NIL NIL icon))
             (T                                        (* install a new icon)
                [WINDOWPROP W (QUOTE TEDIT.ICON.TITLE)
```

```
                        (SETQ iconTitle (TEXTSTREAM.TITLE (TEXTSTREAM W]
                 (WINDOWPROP W (QUOTE ICON)
                        (TITLEDICONW FE.titledIconTemplate iconTitle FE.iconFont NIL T (QUOTE TOP]
    (WINDOWPROP W (QUOTE ICON])

)

(RPAQ FE.Icon (READBITMAP))


(120 61 "@@@COOOOOOOOOOOOOOOOOOOOOOOOOO@@" "@@@GOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "@@@N@CHCHAL@L@O@GCION@C@CHAHAO@@" "@@AN@C@AH@L@L@F@CCION@C@AHAHAO@@" "@@CNGOCIILOCLNFGCAIONGOCINGNGO@@"
     "@@GN@GCIILOCLNFGC@ION@GCINGNGO@@" "@@ON@GCIH@OCL@F@C@AON@GCINGNGO@@" "@AONGOCIHAOCL@N@CBAONGOCINGNGO@@"
     "@CONGOCIIAOCLHNGCCAONGOCINGNGO@@" "@GONGO@AIHOCLLFGCCION@C@AHANGO@@" "@OONGOHCILGCLNBGCCION@C@CHANGO@@"
     "AOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "CN@@@@@@@@@@@@@@@@@@@B@@@@@@@A@@" "GN@@@@@@@@@@@@@@@@@B@@@@@@@@@A@@"
     "OJ@@@@@@@@@@@@@@@@@B@@@@@@@@@A@@" "OJ@@@@@@@@@@@@@@@@@@H@@@@@@@@A@@" "NJ@@@@@@@@@@@@@@@@@@H@@@@@@@@A@@"
     "NJ@H@@@@@@@B@@@@@@@H@@H@@@@@@A@@" "JJ@H@@@@@@@B@@@@@@@H@@H@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@"
     "KJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@"
     "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@"
     "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@H@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@H@@@@@@@@@@@@@@@@@@@@@@@A@@"
     "JJ@@@@@@H@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@@@H@@@@@B@H@@@@@@@@@@@@A@@" "JN@@@@@@@@@@@@B@H@@@@@@@@@@@@A@@"
     "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@"
     "JJB@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "NJB@@@@B@B@H@B@@@B@@@@@@@@@@@A@@" "JJ@@@@@B@B@H@B@@@B@@@@@@@@@@@A@@"
     "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@BH@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@BH@@@@@H@@@@B@@@@B@@@@@@@@A@@"
     "JJ@@@@@@@@H@@@@B@@@@B@@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@"
     "JJ@@@@@@@@@@@@@@@@@@@@B@@@@@@A@@" "JJH@@@@@@@@@@@@@@@@@@@B@@@@@@A@@" "JJH@@@@@@@@@@@@@@@@@@@@@@@@@@A@@"
     "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JJ@@@@B@@@@@@@@@@@@@@@@@@@@@@A@@"
     "JJ@@@@B@@@@@@@@@@@@@@H@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@H@@@@@@@A@@" "JJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@"
     "KJ@@@@@@@@@@@@@@@@@@@@@@@@@@@A@@" "JKOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "JH@@@@@D@@@@@@@@A@@@@@@@@@H@@D@@"
     "JOOOOOOOOOOOOOOOOOOOOOOOOOOOOL@@" "J@@@@@@@@@@D@@@@@@D@@@@D@@@@A@@@" "KOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@@"
     "H@@@A@@@@@@@@@@@H@@@@@@@@@@BD@@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOL@@@")

(RPAQ FE.IconMask (READBITMAP))


(120 61 "@@@COOOOOOOOOOOOOOOOOOOOOOOOOO@@" "@@@GOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "@@@OOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "@@AOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "@@COOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "@@GOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "@@OOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "@AOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "@COOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "@GOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "@OOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "AOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "COOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "GOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOL@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOOL@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOO@@@"
     "OOOOOOOOOOOOOOOOOOOOOOOOOOOOL@@@" "OOOOOOOOOOOOOOOOOOOOOOOOOOOOL@@@")

(RPAQ? FE.defaultFont [FONTCLASS (QUOTE FORTRANEDITFONT)
                             (QUOTE (1 (GACHA 12)
                                       (GACHA 10)
                                       (GACHA 10])

(RPAQ? FE.iconFont (FONTCREATE (QUOTE HELVETICA)
                          8
                          (QUOTE BOLD)))

(RPAQ? FE.iconTitleRegion (create REGION LEFT _ 8 BOTTOM _ 8 WIDTH _ 110 HEIGHT _ 40))

(RPAQ? FE.titledIconTemplate (create TITLEDICON ICON _ FE.Icon MASK _ FE.IconMask TITLEREG _ FE.iconTitleRegion))


          (* * VARS)

(ADDTOVAR BackgroundMenuCommands (Fortran% Edit (QUOTE (FE.TopLevel))
                                        "Start a Fortran Edit"))

(RPAQQ BackgroundMenu NIL)

(DECLARE: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS FE.defaultFont FE.iconFont FE.titledIconTemplate TEDIT.READTABLE)
)


          (* * SERVERS -- Defines the Loops MathServer objects)

(DEFCLASSES Cray FortranServer MathServer Server VMSServer)
```

```
(DEFCLASS Cray (MetaClass Class doc                                    (* If you want something to be done quickly, use this class)
                      Edited:                                          (* DSB "30-May-86 14:55")
                      )
       (Supers MathServer))

(DEFCLASS FortranServer (MetaClass Class Edited:                       (* DSB "13-May-86 16:09"))
       (Supers Server))

[DEFCLASS MathServer (MetaClass Class Edited:                          (* DSB "20-May-86 17:21"))
       (Supers Server)
       (ClassVariables (jobManagerProcess NIL doc

            (* the current process on which the job manager is working. All processes alert the job manager by calling the AlertManager
            method and giving this variable as an argument)
]

[DEFCLASS Server (MetaClass AbstractClass Edited:                      (* DSB "10-Nov-86 08:20"))
       (Supers IndexedObject Object)
       (InstanceVariables (host NIL doc                                (* network name of host ; eg., GSLVAX))
              (name NIL doc                                            (* vernacular name of host; eg., GSLVAX))
              (description NIL doc                                     (* short description of server))
              (serverDirectory NIL doc                                 (* directory for server command files))
              (queues NIL doc                                          (* list of names of batch queues))
              (sourceExtension NIL doc                                 (* default extension, such as FOR, for source files))
              (commandFileExtension NIL doc                            (* default extension, such as COM, for command files)]

[DEFCLASS VMSServer (MetaClass Class doc                               (* this is a DEC VMS machine)
                           Edited:                                     (* DSB "10-Nov-86 08:22")
                           )
       (Supers MathServer FortranServer)
       (InstanceVariables (sourceExtension FOR doc                     (* VMS fortran extension))
              (commandFileExtension COM doc                            (* VMS command file default extension))
              (comFileName (SUBMITJOB submitJob.com ABORTJOB abortJob.com RUNJOB runJob.com STATUS status.com
                                LINK link.com COMPLINK complink.com COMPILE compile.com GETTIME getTime.com)
                     doc                                               (* VMS com files)
                     )
              (resultFileName (SUBMITJOB submitJob.res ABORTJOB abortJob.res RUNJOB runJob.res STATUS status.res
                                  COMPILE compile.res LINK link.res COMPLINK complink.res GETTIME getTime.res
                                  )
                     doc                                               (* VMS result files)
                     ]

(METH FortranServer Compile (filename)                                 (* compiles file, which must be on the host)
      )

(METH FortranServer Compiled? (host defaultDirectory filename)         (* Checks if an object file exists on the host.
                                                                       If so, returns T)
      )

(METH FortranServer Link (filename linkedFilesList)                    (* links object files on the host into an executable file)
      )

(METH FortranServer Linked? (host defaultDirectory filename)           (* Checks if an executable file exists on the host.
                                                                       If so, returns T)
      )

(METH MathServer AlertManager NIL

            (* This method is called by all job processes. It starts up the JobManager process if not awakened, and passes the
            jobManagerProcess variable.)
)

(METH Server AbortJob (jobNumber queue)                                (* aborts specific batch job on stated queue)
      (category MainOps))

(METH Server CommandFileExtension NIL                                  (* returns the extension recognized by the system as a
                                                                       command file))

(METH Server Description NIL                                           (* returns description of the server))

(METH Server Error? (result)                                           (* checks if CAR of result list is "ERROR")
      (category Results))

(METH Server ErrorFile (result)

            (* returns the full name of the error file, specified by the third element in the result list)

      (category Results))

(METH Server ErrorString (result)                                      (* returns the error string: the second element in result list)
      (category Results))

(METH Server ExecuteCommandFile (commandFile parameterList)

            (* method to run a command file. The command string is assembled by the local method CommandString)
```

```
      )

(METH Server ExtractFilename (result)                                    (* Extract error file name from result)
      )

(METH Server GetQueues NIL                                               (* returns the list of queues for the server)
      (category Name))

(METH Server GetTime NIL                                                 (* gets the time from the server)
      (category MainOps))

(METH Server Host NIL                                                    (* returns local server instance variable host)
      (category Name))

(METH Server MakeError (string fileName)                                 (* makes an ERROR ... list)
      (category Results))

(METH Server MakeFullName (fileName directory)                           (* Constructs full name of file and host)
      (category Name))

(METH Server MakePartialName (fileName directory)                        (* Constructs name of file with directory, but without host)
      )

(METH Server Name NIL                                                    (* returns vernacular server name)
      (category Name))

(METH Server PutErrorInWindow (errorFile errorWindow mainWindow)         (* puts text of errorFile in a window)
      )

(METH Server PutTextInWindow (filename position)                         (* Opens a scrollable TEDIT window for the file)
      )

(METH Server Result (result)

          (* returns the second element in the result list when there is no error.
          This is typically the jobID.)

      (category Results))

(METH Server RunFile (file parameterList resultFile noErrorFlg)          (* general method to run a command file and get result and
                                                                         errors)
      (category MainOps))

(METH Server RunJob (filename parameterList)

          (* runs com file, "filename" , with additional parameters "parameterList" , all of which must be on the host, as an
          interactive-type job)

      (category MainOps))

(METH Server ServerDirectory NIL                                         (* returns the name of the server directory for command files)
      (category Name))

(METH Server SourceExtension NIL                                         (* returns default extension for source files))

(METH Server Status (jobNumber)                                          (* get machine status of batch jobs)
      (category MainOps))

(METH Server SubmitJob (filename queue parameterList)                    (* submits file, which must be on the host, as a batch job)
      (category MainOps))

(METH Server UserDirectory NIL                                           (* Gets user name on appropriate host)
      (category Name))

(METH VMSServer MakeCommandString (commandFile parameterList)            (* assembles command string from given name of commandFile
      and parameterList)                                                 )
      )

(DEFINEQ
```

(FortranServer.Compile

```
  (Method ((FortranServer Compile)
           self filename)                                                (* DSB " 5-Aug-86 10:44")
                                                                         (* compiles file, which must be on the host)
          (_ self RunFile (_ self MakePartialName (LISTGET (@ comFileName)
                                                           (QUOTE COMPILE))
                              (_ self ServerDirectory))
             filename
             (_ self MakeFullName (LISTGET (@ resultFileName)
                                           (QUOTE COMPILE)))
             T)))
```

(FortranServer.Compiled?

```
  [Method ((FortranServer Compiled?)
           self host defaultDirectory filename)                          (* DSB "12-Aug-86 09:46")
                                                                         (* Checks if an object file exists on the host.
                                                                         If so, returns T)
```

```
        (INFILEP (PACKFILENAME (QUOTE HOST)
                       (_ host Name)
                       (QUOTE DIRECTORY)
                       defaultDirectory
                       (QUOTE NAME)
                       filename
                       (QUOTE EXTENSION)
                       (QUOTE OBJ])
```

(FortranServer.Link

```
  (Method ((FortranServer Link)
           self filename linkedFilesList)                    (* DSB " 8-Aug-86 09:33")
                                                             (* links object files on the host into an executable file)

           (* * linkedFilesList is either NIL or a list composed of a single string.
           The parameterList sent to RunFile is thus a list composed of the filename and optional string of linked files.)

          (_ self RunFile (_ self MakePartialName (LISTGET (@ comFileName)
                                                           (QUOTE LINK))
                              (_ self ServerDirectory))
             (CONS filename linkedFilesList)
             (_ self MakeFullName (LISTGET (@ resultFileName)
                                           (QUOTE LINK)))
             T)))
```

(FortranServer.Linked?

```
  [Method ((FortranServer Linked?)
           self host defaultDirectory filename)              (* DSB "12-Aug-86 10:10")
                                                             (* Checks if an executable file exists on the host.
                                                             If so, returns T)
          (INFILEP (PACKFILENAME (QUOTE HOST)
                         (_ host Name)
                         (QUOTE DIRECTORY)
                         defaultDirectory
                         (QUOTE NAME)
                         filename
                         (QUOTE EXTENSION)
                         (QUOTE EXE])
```

(MathServer.AlertManager

```
  (Method ((MathServer AlertManager)
           self)                                             (* DSB "22-May-86 16:54")

           (* This method is called by all job processes. It starts up the JobManager process if not awakened, and passes the
           jobManagerProcess variable.)

           (* * Start MS.JobManager if it isn't going)

          (COND
             [(NOT (PROCESSP (@ ::jobManagerProcess)))
              (_@
               ::jobManagerProcess
               (ADD.PROCESS (QUOTE (MS.JobManager))
                      (NAME (QUOTE JobManager)
                            RESTARTABLE
                            (QUOTE HARDRESET]
             (T NIL))))
```

(Server.AbortJob

```
  [Method ((Server AbortJob)
           self jobNumber queue)                             (* DSB "12-Aug-86 18:07")
                                                             (* aborts specific batch job on stated queue)
          (_ self RunFile (_ self MakePartialName (LISTGET (@ comFileName)
                                                           (QUOTE ABORTJOB))
                              (_ self ServerDirectory))
             (LIST jobNumber queue)
             (_ self MakeFullName (LISTGET (@ resultFileName)
                                           (QUOTE ABORTJOB])
```

(Server.CommandFileExtension

```
  (Method ((Server CommandFileExtension)
           self)                                             (* DSB "10-Nov-86 08:19")
                                                             (* returns the extension recognized by the system as a
                                                             command file)

          (@ commandFileExtension)))
```

(Server.Description

```
  (Method ((Server Description)
           self)                                             (* DSB "19-Aug-86 14:29")
                                                             (* returns description of the server)

          (@ description)))
```

(Server.Error?

```
  (Method ((Server Error?)
           self result)                                    (* DSB "21-May-86 11:44")
                                                           (* checks if CAR of result list is "ERROR")
          (EQ (QUOTE ERROR)
              (CAR result))))
```

(Server.ErrorFile

```
  [Method ((Server ErrorFile)
           self result)                                    (* DSB "21-May-86 11:50")

           (* returns the full name of the error file, specified by the third element in the result list)

          (INFILEP (_ self MakeFullName (CADDR result])
```

(Server.ErrorString

```
  (Method ((Server ErrorString)
           self result)                                    (* DSB "21-May-86 11:46")
                                                           (* returns the error string: the second element in result list)
          (CADR result)))
```

(Server.ExecuteCommandFile

```
  (Method ((Server ExecuteCommandFile)
           self commandFile parameterList)                 (* DSB "10-Nov-86 10:59")

           (* method to run a command file. The command string is assembled by the local method CommandString)

          (PROGRAMCHAT (_ self Host)
                 (_ self MakeCommandString commandFile parameterList)
                 NIL)))
```

(Server.ExtractFilename

```
  (Method ((Server ExtractFilename)
           self result)                                    (* DSB " 6-Aug-86 11:28")
                                                           (* Extract error file name from result)
          (CAR (REVERSE result))))
```

(Server.GetQueues

```
  (Method ((Server GetQueues)
           self)                                           (* DSB " 9-Jun-86 08:41")
                                                           (* returns the list of queues for the server)
          (@ queues)))
```

(Server.GetTime

```
  (Method ((Server GetTime)
           self)                                           (* DSB "13-Jun-86 13:17")
                                                           (* gets the time from the server)
                                                           (* RunFile returns a list whose CAR is OK)
          (_ self Result (_ self RunFile (_ self MakePartialName (LISTGET (@ comFileName)
                                                                          (QUOTE GETTIME))
                                         (_ self ServerDirectory))
                         NIL
                         (_ self MakeFullName (LISTGET (@ resultFileName)
                                                       (QUOTE GETTIME)))
                         NIL))))
```

(Server.Host

```
  (Method ((Server Host)
           self)                                           (* DSB "23-May-86 13:52")
                                                           (* returns local server instance variable host)
          (CANONICAL.HOSTNAME (@ host))))
```

(Server.MakeError

```
  (Method ((Server MakeError)
           self string fileName)                           (* DSB "21-May-86 15:49")
                                                           (* makes an ERROR ... list)
          (LIST (QUOTE ERROR)
                string fileName)))
```

(Server.MakeFullName

```
  [Method ((Server MakeFullName)
           self fileName directory)                        (* DSB "22-May-86 14:54")
                                                           (* Constructs full name of file and host)

           (* * if directory is not specified, it uses the user's login name on the host;
           i.e., the user's directory)
```

```
         (COND
            ((NOT directory)
             (PACKFILENAME (QUOTE HOST)
                    (_ self Host)
                    (QUOTE DIRECTORY)
                    (_ self UserDirectory)
                    (QUOTE BODY)
                    fileName))
            (T

          (* * otherwise, it uses the specified directory name)

               (PACKFILENAME (QUOTE HOST)
                      (_ self Host)
                      (QUOTE DIRECTORY)
                      directory
                      (QUOTE BODY)
                      fileName])
```

(Server.MakePartialName

```
  [Method ((Server MakePartialName)
           self fileName directory)                    (* DSB "13-Jun-86 13:07")
                                                       (* Constructs name of file with directory, but without host)

          (* * if directory is not specified, it uses the user's login name on the host;
          i.e., the user's root directory)

         (COND
            ((NOT directory)
             (PACKFILENAME (QUOTE DIRECTORY)
                    (_ self UserDirectory)
                    (QUOTE BODY)
                    fileName))
            (T

          (* * otherwise, it uses the specified directory name)

               (PACKFILENAME (QUOTE DIRECTORY)
                      directory
                      (QUOTE BODY)
                      fileName])
```

(Server.Name

```
  (Method ((Server Name)
           self)                                       (* DSB " 9-Jun-86 08:34")
                                                       (* returns vernacular server name)
         (@ name)))
```

(Server.PutErrorInWindow

```
  (Method ((Server PutErrorInWindow)
           self errorFile errorWindow mainWindow)      (* DSB "15-Aug-86 17:42")
                                                       (* puts text of errorFile in a window)

          (* * put the errorFile in the errorWindow and set errorWindow props)

         [OPENTEXTSTREAM errorFile errorWindow NIL NIL (QUOTE (PROMPTWINDOW (WINDOWPROP mainWindow (QUOTE
                                                                                                    PROMPTWINDOW
                                                                                                          ]

         (WINDOWPROP errorWindow (QUOTE ERRORFILE)
                errorFile)
         (WINDOWADDPROP errorWindow (QUOTE CLOSEFN)
                (QUOTE MS.CleanupErrorFile))))
```

(Server.PutTextInWindow

```
  [Method ((Server PutTextInWindow)
           self filename position)                     (* DSB "21-Jul-86 14:23")
                                                       (* Opens a scrollable TEDIT window for the file)
         (TEDIT filename NIL NIL (QUOTE (READONLY T])
```

(Server.Result

```
  (Method ((Server Result)
           self result)                                (* DSB "21-May-86 17:52")

          (* returns the second element in the result list when there is no error.
          This is typically the jobID.)

         (CADR result)))
```

(Server.RunFile

```
  (Method ((Server RunFile)
```

```
          self file parameterList resultFile noErrorFlg)          (* DSB " 9-Nov-86 13:49")
                                                                  (* general method to run a command file and get result and
                                                                  errors)
```

(* * "file" is the name of the command file given to PROGRAMCHAT to be run on the host, and it must be in the proper host format (eg, <gslws.server>submitjob) whereas "resultFile" is the name of the result file returned on the host, but it must be in the proper LISP naming format (eg, {GSLVAX10}<bloomberg>submitjob.res))

```
        (LET (f result fullResultFile newFile)
```

(* * runs a command file)

```
            (_ self ExecuteCommandFile file parameterList)
```

(* * look for result file)

```
            [COND
               ((SETQ fullResultFile (INFILEP (_ self MakeFullName resultFile)))
                (SETQ f (OPENFILE fullResultFile (QUOTE INPUT)))
                (SETQ result (READ f))
                (CLOSEF f)                                        (* (DELFILE f))
                )
               (T (SETQ result (_ self MakeError "no result" NIL]
```

(* * handle the errors)

(* * default case (noErrorFlg=NIL) is not to return on errors. In this case, generate a break with an error message.)

(* * Otherwise, do not break (if noErrorFlg=T)%. Instead, copy the error file to a file on core, and return its filename (e.g., {core}compile.err))

(* * after this runs, start up Job Manager, using (_ self AlertManager))

```
            [COND
               [(NOT noErrorFlg)
                (COND
                   [(_ self Error? result)
                    (DELFILE (_ self ErrorFile result))
                    (ERROR (CONCAT (_ self Name)
                                   ": "
                                   (_ self ErrorString result]
                   (T (SETQ result (_ self Result result]
               (T                                                 (* return complete result to user)
                  (COND
                     ((_ self Error? result)
                      (COND
                         ((SETQ f (_ self ErrorFile result))
                          [SETQ newFile (COPYFILE f (PACKFILENAME (QUOTE HOST)
                                                                  (QUOTE CORE)
                                                                  (QUOTE BODY)
                                                                  (_ self ExtractFilename result]
                                                                     (* (DELFILE f))
                          (SETQ result (_ self MakeError (_ self ErrorString result)
                                          newFile)))
                         (T result)))
                     (T result]
```

(* * starts up JobManager, etc. Not yet implemented)

```
                                                                  (* (_ self AlertManager))
            result)))
```

(Server.RunJob

```
  (Method ((Server RunJob)
           self filename parameterList)                           (* DSB "11-Aug-86 11:06")
```

(* runs com file, "filename" , with additional parameters "parameterList" , all of which must be on the host, as an interactive-type job)

(* * parameterList is in RunJob either NIL or a list of parameters composed of a single string.
The parameterList sent to RunFile is thus a list composed of the filename and optional string of associated parameters.)

```
        (_ self RunFile (_ self MakePartialName (LISTGET (@ comFileName)
                                                         (QUOTE RUNJOB))
                        (_ self ServerDirectory))
           (CONS filename parameterList)
           (_ self MakeFullName (LISTGET (@ resultFileName)
                                         (QUOTE RUNJOB)))
           T)))
```

(Server.ServerDirectory

```
  (Method ((Server ServerDirectory)
           self)                                                  (* DSB "22-May-86 15:37")
                                                                  (* returns the name of the server directory for command files)
          (@ serverDirectory)))
```

(Server.SourceExtension

```
  (Method ((Server SourceExtension)
           self)                                                    (* DSB "21-Aug-86 16:46")
                                                                    (* returns default extension for source files)
          (@ sourceExtension)))
```

(Server.Status

```
  [Method ((Server Status)
           self jobNumber)                                          (* DSB "12-Aug-86 13:51")
                                                                    (* get machine status of batch jobs)

           (* if a jobNumber is specified it returns either the CPU time elapsed
           (if running) or the error message if it bombed, or NIL if neither.)

           (* if no jobNumber is specified, returns a list, each element of which is a prop list of the form
           ((JOB jobNumber) (CPU timeElapsed)))

          (LET (errorFile f result)

           (* * if a jobNumber is specified, return its status)

               (COND
                  [jobNumber (OR (CAR (_ self RunFile (_ self MakePartialName (LISTGET (@ comFileName)
                                                                                       (QUOTE STATUS))
                                                      (_ self ServerDirectory))
                                      jobNumber
                                      (_ self MakeFullName (LISTGET (@ resultFileName)
                                                                    (QUOTE STATUS)))
                                      NIL))
                                 (COND
                                    ([SETQ errorFile (INFILEP (_ self MakeFullName (CONCAT jobNumber ".res"]
                                     (SETQ f (OPENFILE errorFile (QUOTE INPUT)))
                                     (SETQ result (READ f))
                                     (CLOSEF f)                     (* (DELFILE f))
                                     result)
                                    (T NIL]
               (T

           (* * else, return the status of all active jobs)

                  (_ self RunFile (_ self MakePartialName (LISTGET (@ comFileName)
                                                                   (QUOTE STATUS))
                                                     (_ self ServerDirectory))
                     jobNumber
                     (LISTGET (@ resultFileName)
                              (QUOTE STATUS))
                     NIL])
```

(Server.SubmitJob

```
  (Method ((Server SubmitJob)
           self filename queue parameterList)                       (* DSB " 8-Aug-86 11:47")
                                                                    (* submits file, which must be on the host, as a batch job)

           (* * parameterList in SubmitJob is either NIL or a list of parameters composed of a single string.
           The parameterList sent to RunFile is thus a list ocmposed of the filename, queue, and optional string of associated
           parameters.)

          (_ self RunFile (_ self MakePartialName (LISTGET (@ comFileName)
                                                           (QUOTE SUBMITJOB))
                                                  (_ self ServerDirectory))
             (CONS filename (CONS queue parameterList))
             (_ self MakeFullName (LISTGET (@ resultFileName)
                                           (QUOTE SUBMITJOB)))
             NIL)))
```

(Server.UserDirectory

```
  [Method ((Server UserDirectory)
           self)                                                    (* DSB "13-Jun-86 11:34")
                                                                    (* Gets user name on appropriate host)
                                                                    (* Forces login if not logged in)
          (OR (MKATOM (CAAR (GETHASH (CANONICAL.HOSTNAME (_ self Host))
                                     LOGINPASSWORDS)))
              (LOGIN (_ self Host])
```

(VMSServer.MakeCommandString

```
  [Method ((VMSServer MakeCommandString)
           self commandFile parameterList)                          (* DSB "22-May-86 16:05")
                                                                    (* assembles command string from given name of commandFile
          and parameterList)

           (* Note that the commandFile and the parameterList must be quoted when this function is called)
```

```
         (CONCAT "@" commandFile (for p in (MKLIST parameterList) bind (s _ "")
                                    do (SETQ s (CONCAT s " " p)) finally (RETURN s])

)

(DEFINEQ
```

(MS.MakeInstances

```
  [LAMBDA NIL                                                    (* DSB " 9-Oct-86 11:18")
                                                                 (* Initialization routine: makes browser and instances of servers)

          (* * make class browser for Server)

    (LET (newBrowser)
         (SETQ newBrowser (_ ($ ClassBrowser)
                             New))
         (_ newBrowser AddRoot ($ Server)))

          (* * make $GSLVAX instance of VMSServer)

    (_ ($ VMSServer)
       New
       (QUOTE GSLVAX))
    (_@
     ($ GSLVAX)
     host
     (QUOTE GSLVAX))
    (_@
     ($ GSLVAX)
     name
     (QUOTE GSLVAX))
    (_@
     ($ GSLVAX)
     description "The GSL 11/780 VMS Server")
    (_@
     ($ GSLVAX)
     serverDirectory
     (QUOTE <GSLWS.SERVER>))
    (_@
     ($ GSLVAX)
     queues
     (QUOTE (Fast Medium Slow)))

          (* * make $SITKA instance of VMSServer)
                                                                 (* the host value, SITKA, refers to the pup address 204#156#)
    (_ ($ VMSServer)
       New
       (QUOTE SITKA))
    (_ ($ SITKA)
       PutValue
       (QUOTE host)
       (QUOTE SITKA))
    (_@
     ($ SITKA)
     name
     (QUOTE SITKA))
    (_@
     ($ SITKA)
     description "The GSL microVAX VMS Server")
    (_@
     ($ SITKA)
     serverDirectory
     (QUOTE <bloomberg.gslws>))
    (_@
     ($ SITKA)
     queues
     (QUOTE (Fast Slow)))
    (_@
     ($ SITKA)
     comFileName
     (QUOTE (SUBMITJOB submitJob.com ABORTJOB abortJob.com RUNJOB runJob.com STATUS status.com COMPILE
                   compile.com LINK link.com GETTIME getTime.com)))
    (_@
     ($ SITKA)
     resultFileName
     (QUOTE (SUBMITJOB submitJob.res ABORTJOB abortJob.res RUNJOB runJob.res STATUS status.res COMPILE
                   compile.res LINK link.res GETTIME getTime.res)))

          (* * make $MADVAX instance of VMSServer)

    (_ ($ VMSServer)
       New
       (QUOTE MADVAX))
    (_@
     ($ MADVAX)
     host
     (QUOTE MADVAX))
```

```
    (_@
     ($ MADVAX)
     name
     (QUOTE MADVAX))
    (_@
     ($ MADVAX)
     description "The AIS 11/750 VMS Server")
    (_@
     ($ MADVAX)
     serverDirectory
     (QUOTE <bloomberg.gslws>))
    (_@
     ($ MADVAX)
     queues
     (QUOTE (Fast Medium Slow)))
```

(* * make $CRAYZY instance of Cray VaporServer)

```
    (_ ($ Cray)
       New
       (QUOTE CRAYZY))
    (_ ($ CRAYZY)
       PutValue
       (QUOTE host)
       (QUOTE CRAYZY))
    (_@
     ($ CRAYZY)
     name
     (QUOTE CRAYZY))
    (_@
     ($ CRAYZY)
     description "Not yet plugged in..."])
```

(**StripPA**

```
  [LAMBDA (username)                                          (* DSB "22-May-86 11:50")
    (SUBATOM username 1 (LET ((POS (STRPOS "." username)))
                             (COND
                                ((FIXP POS)
                                 (SUB1 POS))
                                (T NIL])
```

)

(**MS.DestroyInstances**)

(**MS.MakeInstances**)

(* * PROGRAMCHAT -
Windowless CHAT for communication)

```
(DEFINEQ
```

(**OPENCHATSTREAM**

```
  [LAMBDA (HOST)                                              (* ejs: "23-Feb-85 19:22")
    (PROG (OPENFUNCTION)
          (COND
             [(BOUNDP (QUOTE CHAT.PROTOCOLTYPES))
              (COND
                 ((for PROTOCOL in CHAT.PROTOCOLTYPES thereis (SETQ OPENFUNCTION (APPLY* (CDR PROTOCOL)
                                                                                         HOST)))
                  (RETURN (APPLY* (CADR OPENFUNCTION)
                                  (CAR OPENFUNCTION]
             ((BOUNDP (QUOTE CHAT.PROTOCOLS))
              (COND
                 ((for PROTOCOL in CHAT.PROTOCOLS thereis (SETQ OPENFUNCTION (APPLY* PROTOCOL HOST)))
                  (RETURN (APPLY* (CADR OPENFUNCTION)
                                  (CAR OPENFUNCTION])
```

(**PROGRAMCHAT**

```
  [LAMBDA (HOST CMDSTREAM LOGSTREAM)                          (* DSB " 9-Nov-86 13:02")
    (PROG ((STREAMPAIR (OPENCHATSTREAM HOST))
           INCHAT OUTCHAT)
          (COND
             (STREAMPAIR (SETQ INCHAT (CAR STREAMPAIR))
                    (SETQ OUTCHAT (CDR STREAMPAIR))
                    (SETFILEINFO OUTCHAT (QUOTE ENDOFSTREAMOP)
                           (FUNCTION CHAT.ENDOFSTREAMOP))
                    (SETFILEINFO INCHAT (QUOTE ENDOFSTREAMOP)
                           (FUNCTION CHAT.ENDOFSTREAMOP))
                    [ADD.PROCESS (BQUOTE (PROGRAMCHAT.OUTPUT (QUOTE , INCHAT)
                                                             (QUOTE , LOGSTREAM]
                    (BLOCK)
                    (PROGRAMCHAT.LOGIN HOST INCHAT OUTCHAT)
                    [COND
```

```
                    ((STRINGP CMDSTREAM)
                     (SETQ CMDSTREAM (OPENSTRINGSTREAM CMDSTREAM (QUOTE INPUT]
                  [COND
                    ((NULL LOGSTREAM)
                     (SETQ LOGSTREAM (OPENSTREAM (QUOTE {NULL})
                                                 (QUOTE OUTPUT]
                  (while (AND (OPENP OUTCHAT (QUOTE OUTPUT))
                              (NOT (EOFP CMDSTREAM)))
                     do (BOUT OUTCHAT (BIN CMDSTREAM))
                        (BLOCK)
                     finally (COND
                               ((EOFP CMDSTREAM)
                                (CLOSEF CMDSTREAM)
                                (BOUT OUTCHAT (CHARCODE CR))
                                (PROGRAMCHAT.LOGIN HOST INCHAT OUTCHAT (QUOTE LOGOUT))
                                (FORCEOUTPUT OUTCHAT T)
                                (until (NOT (OPENP INCHAT (QUOTE INPUT))) do (BLOCK) finally (CLOSEF OUTCHAT])
```

(PROGRAMCHAT.LOGIN

```
  [LAMBDA (HOST INSTREAM OUTSTREAM OPTION)                    (* ejs: "24-Jan-85 18:52")

          (* * Login to HOST. If a job already exists on HOST, Attach to it unless OPTION overrides.)

    (PROG ((LOGINFO (CDR (ASSOC (OR (GETOSTYPE HOST)
                                    (QUOTE IFS))
                                NETWORKLOGINFO)))
           NAME/PASS COM)
          (OR LOGINFO (RETURN))
          (SETQ NAME/PASS (\INTERNAL/GETPASSWORD HOST))
          [SETQ COM (COND
                      (OPTION)
                      ((ASSOC (QUOTE ATTACH)
                              LOGINFO)
                       (OR (CHAT.LOGINFO INSTREAM HOST (CAR NAME/PASS))
                           (QUOTE LOGIN)))
                      (T                                      (* Don't know how to do anything but login, so silly to try
                                                                 anything else)
                         (QUOTE LOGIN]
          (COND
             ((NULL (SETQ LOGINFO (ASSOC COM LOGINFO)))
              (printout PROMPTWINDOW T "Login option " COM " not implemented for this type of host"))
             (T (for X in (CDR LOGINFO) do (SELECTQ X
                                             (CR (BOUT OUTSTREAM (CHARCODE CR))
                                                 (FORCEOUTPUT OUTSTREAM))
                                             (USERNAME (PRIN3 (CAR NAME/PASS)
                                                              OUTSTREAM))
                                             (PASSWORD (PRIN3 (\DECRYPT.PWD (CDR NAME/PASS))
                                                              OUTSTREAM))
                                             (WAIT              (* Some systems do not permit typeahead)
                                                   (COND
                                                      ((NOT (CHAT.FLUSH&WAIT INSTREAM))
                                                                 (* Couldn't sync, so wait longer.)
                                                       (DISMISS CHAT.WAIT.TIME)))
                                                   (DISMISS CHAT.WAIT.TIME))
                                             (PRIN3 X OUTSTREAM)))
                (FORCEOUTPUT OUTSTREAM])
```

(PROGRAMCHAT.OUTPUT

```
  [LAMBDA (INCHATSTREAM LOGSTREAM)                            (* ejs: "23-Feb-85 19:18")
    (bind CH while (AND (NEQ CH -1)
                        (OPENP INCHATSTREAM (QUOTE INPUT)))
       do (SETQ CH (BIN INCHATSTREAM))
          [COND
             ((NEQ CH -1)
              (COND
                 (LOGSTREAM (BOUT LOGSTREAM CH]
       finally (COND
                  ((OPENP INCHATSTREAM)
                   (CLOSEF INCHATSTREAM])

)


          (* * VARS for our site)

(RPAQQ NETWORKLOGINFO
       ((TENEX (LOGIN "LOGIN " USERNAME " " PASSWORD "
                      ")
               (ATTACH "ATTACH " USERNAME " " PASSWORD "
                       ")
               (WHERE "WHERE " USERNAME CR "ATTACH " USERNAME " " PASSWORD CR)
               (LOGOUT "LOGOUT" CR))
        (TOPS20 (LOGIN "LOGIN " USERNAME CR PASSWORD CR)
                (ATTACH "ATTACH " USERNAME "Ÿ" CR PASSWORD CR)
                (WHERE "LOGIN " USERNAME CR PASSWORD CR)
```

```
            (LOGOUT "LOGOUT" CR))
        (UNIX (LOGIN WAIT CR WAIT USERNAME CR WAIT PASSWORD CR WAIT WAIT WAIT WAIT CR)
              (LOGOUT WAIT CR "logout" CR))
        (IFS (LOGIN "Login " USERNAME " " PASSWORD CR)
             (ATTACH)
             (LOGOUT "Quit" CR))
        (VMS (LOGIN USERNAME CR PASSWORD CR)
             (LOGOUT "LOGOUT" CR))
        (NS (LOGIN "Logon" CR USERNAME CR PASSWORD CR)
            (LOGOUT "LOGOFF" CR))))

(pushnew NETWORKOSTYPES (QUOTE (GSLVAX . VMS))
       (QUOTE (SITKA . VMS))
       (QUOTE (MADVAX . VMS)))
```

(* * PROGRAMMER'S INTERFACE -
use remote servers with LISP calls)

```
(DEFINEQ
```

(PRIN.RunRemote

```
  [LAMBDA (hostname filename parameterList)                    (* DSB "25-Nov-86 09:41")
                                                               (* Main Programmer's Interface (PRIN) function)
    (PROG (host file result)
```

(* * check preliminaries)

```
          (SETQ host (PRIN.ValidateHost hostname))
          [COND
             ((NOT host)
              (RETURN (PRIN.Error (CONCAT "Host " (U-CASE hostname)
                                          " is not valid"]
          (SETQ file (PRIN.ValidateFilename filename host hostname))
          [COND
             ((NOT file)
              (RETURN (PRIN.Error (CONCAT "Command file " filename " does not exist"]
```

(* * run the job)

```
          (SETQ result (_ host RunJob file parameterList))
```

(* * handle the results)

```
          (COND
             ((EQUAL (CAR result)
                     (QUOTE ERROR))
              (PRIN.Error "Run-time warning or error" host file result))
             (T (PROMPTPRINT (CONCAT "Call to remote host " (U-CASE hostname)
                                     " succeeded without error"))
                (RETURN T])
```

(PRIN.ValidateHost

```
  [LAMBDA (hostname)                                           (* DSB "10-Nov-86 08:06")
                                                               (* returns the host, or NIL if nonexistant)
    (PROG (host)
          (SETQ host (FE.GetServer (U-CASE hostname)))
          (RETURN host])
```

(PRIN.ValidateFilename

```
  [LAMBDA (filename host hostname)                             (* DSB "10-Nov-86 08:40")
                                                               (* returns file if command file exists, or NIL otherwise)
    (PROG (directory name extension wholename fileExists?)
          (SETQ directory (UNPACKFILENAME filename (QUOTE DIRECTORY)))
          (SETQ name (UNPACKFILENAME filename (QUOTE NAME)))
          (SETQ extension (_ host CommandFileExtension))
          (SETQ wholename (PACKFILENAME (QUOTE HOST)
                                        (U-CASE hostname)
                                        (QUOTE DIRECTORY)
                                        directory
                                        (QUOTE NAME)
                                        name
                                        (QUOTE EXTENSION)
                                        extension))
          (SETQ fileExists? (INFILEP wholename))
          (COND
             ((NOT fileExists?)
              (RETURN NIL))
             (T (RETURN (PACKFILENAME (QUOTE DIRECTORY)
                                      directory
                                      (QUOTE NAME)
                                      name])
```

(PRIN.Error

```
  [LAMBDA (errorString host file result)                    (* DSB "10-Nov-86 11:17")
                                                            (* opens an error window and prints the error string and any
                                                            run-time error messages)
    (PROG (hostname errorWindow errorFile)
          (COND
             ((AND host file)
              (SETQ hostname (_ host Name))
              (SETQ errorWindow (CREATEW (QUOTE (300 300 420 200))
                                         (CONCAT "PRIN: " errorString " on host " hostname)))
              (SETQ errorFile (_ host ExtractFilename result))
              (_ host PutErrorInWindow errorFile errorWindow))
             (T (SETQ errorWindow (CREATEW (QUOTE (300 300 300 80))
                                           "Programmer's Interface Error Window"))
                (PRIN1 errorString errorWindow])

)

(PUTPROPS MATHSERVER COPYRIGHT ("Xerox Corporation" 1986 1987))
```

FUNCTION INDEX

VARIABLE INDEX