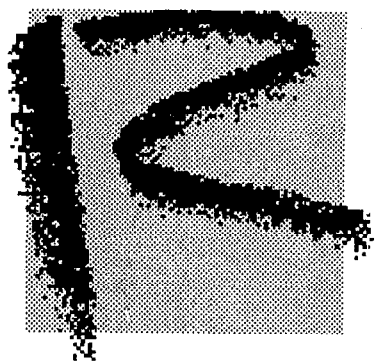
X-CUA

MICHAEL JOHNSON
Relay Technology

X-CUA

The X Window System under VM



RELAY

TECHNOLOGY, INC.

(Formerly VM Systems Group)

History of "X"

- **Display system developed at MIT**
  - — " Project Athena" formed in 1983; funded by MIT, DEC, and IBM
  - — addressed the need to communicate between different kinds of computers over a variety of networks
  - — adapted Stanford University's windowing system ("W") to Unix, renamed "X"
- **Goal: permit graphical presentation under UNIX**
  - — provided framework for a *Graphical User Interface* (GUI)
- **Led to formation of the X Consortium in 1987**
  - — partnership to guide future X standards
  - — founding members included IBM, Apple, Tektronics, DEC, Sun, Hewlett-Packard and AT&T

"X" Features and Functions

- **Object-oriented programming methods**
  - — application deals with whole objects vs. screen areas
- **Separates *client* and *server***
  - — application program is the client
  - — display management program is the server
  - — X Window server ("X server") handles the display; shares terminal with other X clients
- **Software comprised of:**
  - — intrinsics library–simple objects used to create "widgets"–more complex objects
  - — Xlib functions–procedures to perform low-level primitives (e.g., "DrawLine", "CreateWindow")

"X" Features and Functions *(continued)*

- **Display is performed on an X Terminal**
  - — real X Terminal–firmware X server/window manager
  - — PC running X server software
  - — UNIX/AIX workstation running X server software
- **Portable to virtually all platforms**
  - — any hardware capable of supporting an X terminal
  - — any operating system that can support C & communications
  - — communicates over Ethernet, TCP/IP, or DECnet
  - — "X" software available on UNIX, PCs, MACs, etc.
- **Does not impose GUI rules**
  - — adaptable to any GUI foundation: Open Look, OSF Motif, IBM's CUA

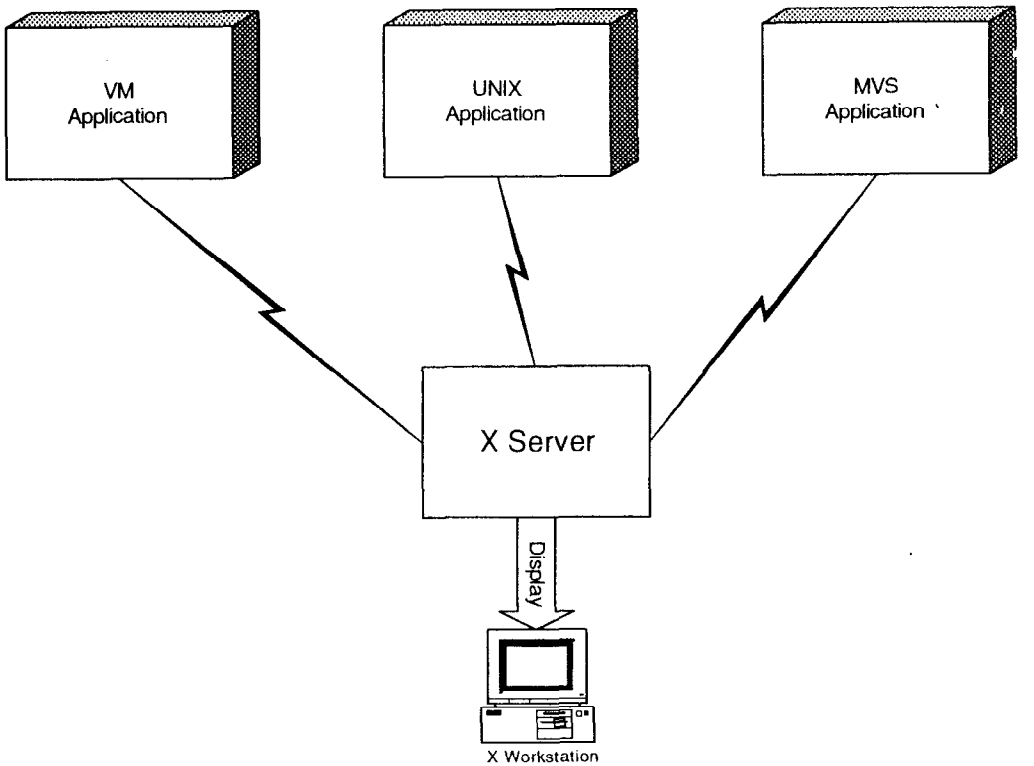
Why use "X"?

- **It is an enabling technology**
  — supports all GUIs, X terminals
- **All platforms can interoperate**
  — users may access distributed applications in parallel
- **Allows seamless integration across networks**
  — location of application is transparent (and usually irrelevant) to the user
- **Provides hardware-independent GUI foundation**
  — placement of client and server is unconstrained
- **Network transparent–TCP/IP, DECnet, etc.**
  — operates with existing communications

Why use "X"? *(continued)*

- **Supports corporate IS goals**
  — provides user-friendly GUI, increases productivity
- **An "Open" system–publicly available**
  — non-proprietary; no vendor dependencies
- **Increases product life cycle**
  — applications can be developed today, ported to other platforms in the future if necessary

The X Server



- **Controls the display screen, resides on workstation**

The X Server *(continued)*

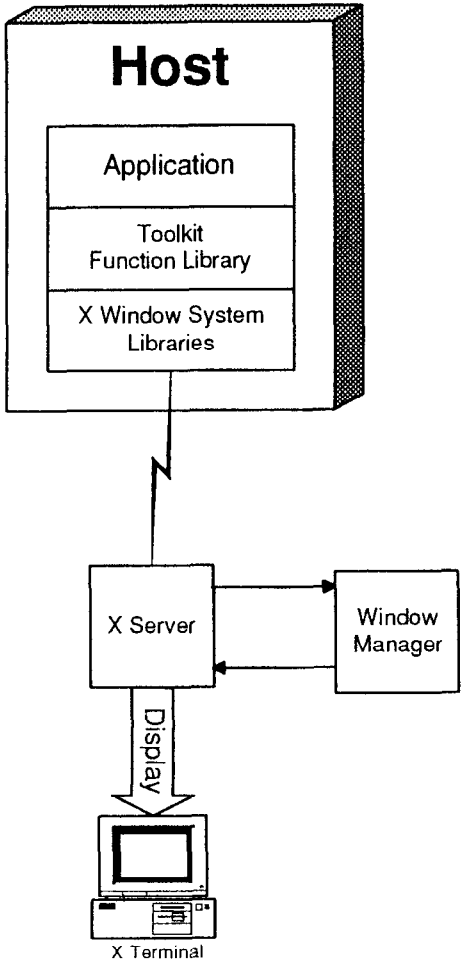
- **Receives instructions from the application in the form of X protocol messages**
- **Performs graphical tasks on behalf of the application (the X client)**
  - — understands terminal's characteristics
  - — makes application device-independent
  - — communicates geometry requests to window manager (a separate program usually included with X server)
- **Sends event messages back to the X Client**
  - — client receives notification, e.g., "ButtonPressed", etc.

The X Server *(continued)*

- **Can handle multiple, concurrent X clients (applications)**

  — single server handles all clients of user's terminal

The X Client



- The X server produces the window's contents and physically writes the display
- The window manager places the "dressing" (borders, etc.) around the window and manages the screen's "real estate" (window position, etc.)

The X Client *(continued)*

- **Oblivious to terminal's characteristics**
  — development effort focus is on tasks vs. environment
- **May exist on the same or different platform**
- **Event-driven methodology**
  — client establishes "callbacks" for desired events
  — client issues instructions, enters enabled wait state
- **Receives event notification from X server**
  — key pressed, window uncovered, etc.

What is X-CUA?

- **An X-based toolkit**
  - — adds CUA-compliant GUI functions to VM, AIX
  - — provides high-level X programming facilities for CUA-compliant applications ("CreateOpenDialog", etc.)
  - — programs can be written in C or REXX
- **Written in C**
  - — operates under UNIX, AIX, and VM
  - — X-CUA applications in C are portable
- **Under VM, uses IBM's TCP/IP product**
  - — enhanced version of the X Window support; recompiled using SAS/C to take advantage of advanced functions, permit royalty-free run time libraries.

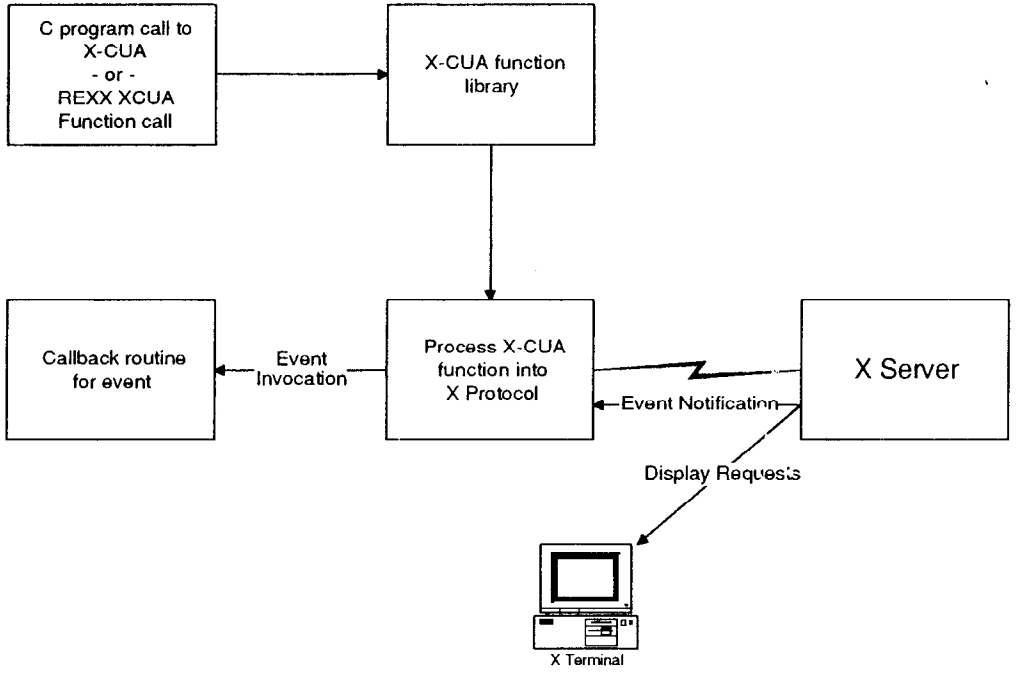
What is X-CUA? *(continued)*

- **REXX function package**
  - — it's REXX!
  - — runs in a saved segment for increased throughput
  - — relatively small applications due to non-redundancy
- **Provides enhanced 3270 emulation program**
  - — supports extended attributes, PSS, graphics
  - — want a 3279 model 4?
  - — users logon to VM, invoke X-CUA application
  - — X-CUA application begins GUI dialog
  - — So...

What is X-CUA? *(continued)*

- **Legacy systems can convert as resources permit**
  - — current system runs alongside in 3270 emulation
  - — pieces of applications can be converted gradually

X-CUA's role

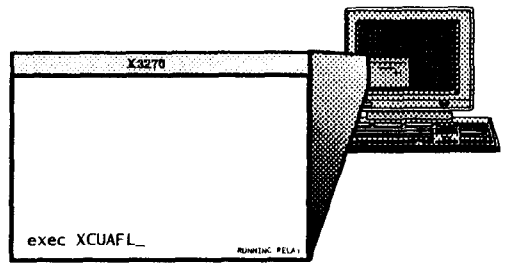


- **Applications in C or REXX call X-CUA**
  - — use supplied "widgets" – objects which implement CUA constructs

X-CUA's role *(continued)*

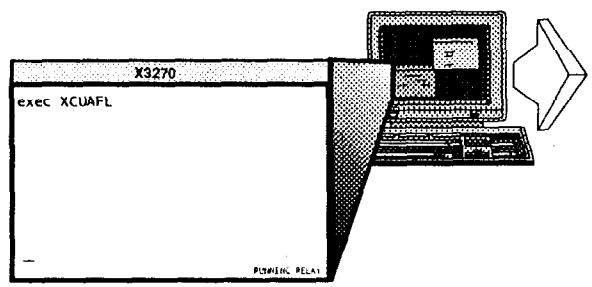
- **The process:**
  - — application calls X-CUA function to build GUI display
  - — registers callbacks for desired events:
    - external REXX function—e.g., "MYSORT EXEC" to handle "Sort" button
    - a C routine
  - — calls X-CUA to make window visible
  - — X-CUA translates requests into X protocol
  - — X-CUA transmits requests to X server
  - — X server sends back event notification
  - — X-CUA invokes callback routine

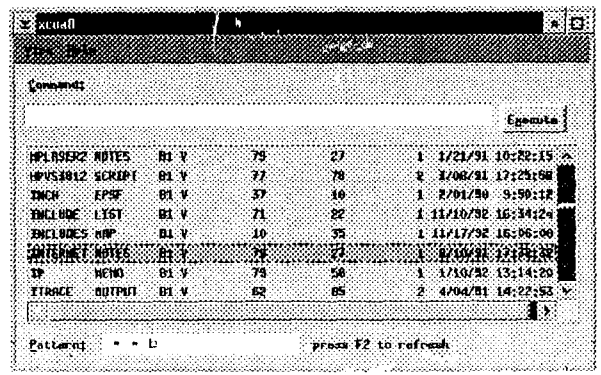
An X-CUA Application



- User logs onto VM through 3270 emulation
- Invokes X-CUA application
- X-CUA application begins initialization

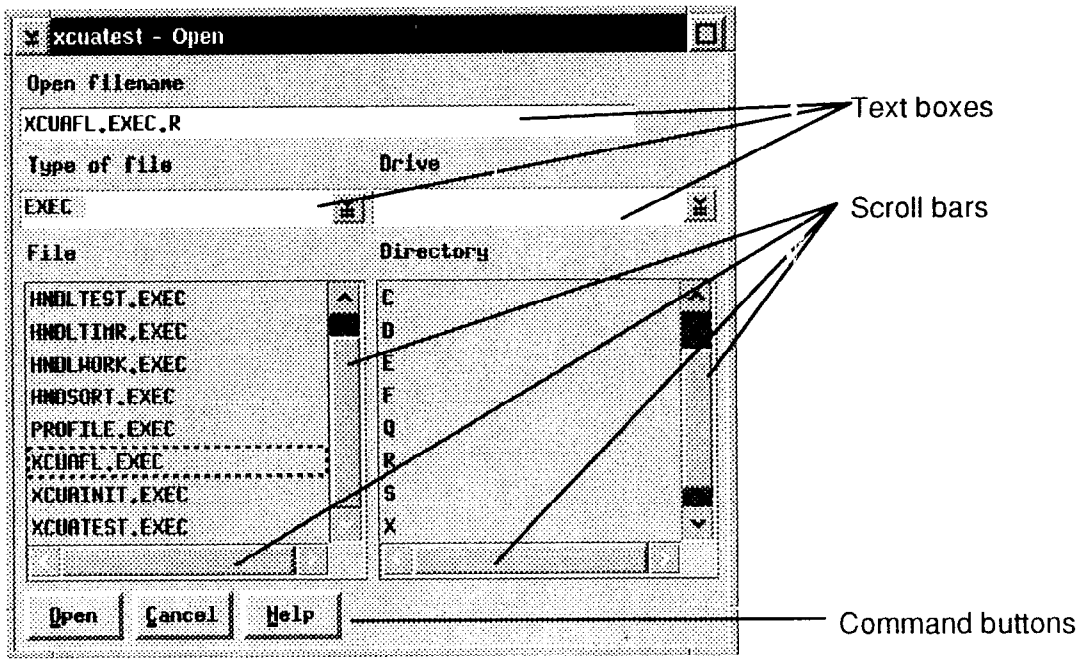
- X-CUA application opens a new window on user's terminal
- Interaction is through new window



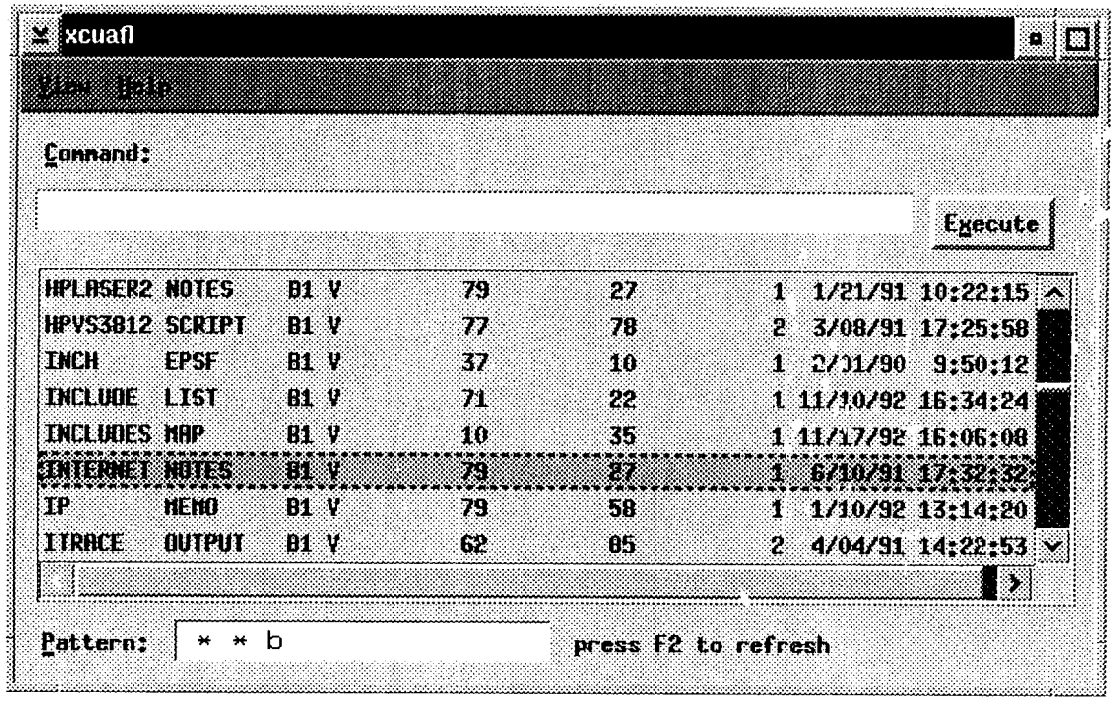


An X-CUA Widget

- **Open Dialog Box**
  - — X-CUA handles scroll bars, text edit, etc.
  - — application is presented with completed data



sis of an X-CUA Application



Analysis of an X-CUA application
(continued)

- **X-CUA REXX functions:**
  - — XCUA("SubfunctionName",args...)
    - Toolkit, windowing functions
    - Performs calls to C-level subroutines

- **GETARGS(), SETOPTION(), SETBUTTONREC(), etc.**
  - — X-CUA REXX utility functions
    - Provide bridge from REXX-think to C-think, vice versa

Analysis of an X-CUA application
(continued)

- **Let's get started:**

```
/* ------------------------------------------------------------------*/
/* X-CUA version of FILELIST utility.                                */
/* ------------------------------------------------------------------*/

address command                                    /* Good hygiene */
parse source . calltype .
signal on syntax

FALSE = 0                                        /* For clearer code */
TRUE = 1

if (calltype ¬== 'COMMAND') then      /* Figure out why we were called */
  signal CALLBACK                            /* More about this later */
```

Analysis of an X-CUA application
(continued)

- **First define the environment:**
  - — Fetch Arguments
  - — Configuration defaults
  - — Options, if there are any
  - — Create primary window
  - — Examine arguments
  - — Configure presentation space

Analysis of an X-CUA application
(continued)

```
/* ------------------------------------------------------------------ */
/* XCUAFL mainline.                                                   */
/* ------------------------------------------------------------------ */

  'EXEC XCUAINIT'              /* Initialize the X-CUA REXX interface */

  call GETARGS 'ARGV.'      /* Get REXX args into a C-style args array */

/* ------------------------------------------------------------------ */
/* Create a Primary window, register an XcNcloseCallback routine.     */
/* ------------------------------------------------------------------ */
  fb.1 = "*TextFont: 9x15"           /* Define fallback configuration */
  fb.0 = 1                           /* In case no config file around */

  call SETOPTION 'OPTIONS.',1,"-ib","*inverseBackground","SepArg",""

  primary = XCUA("XcuaInitialize",'APP_CONTEXT',"XcuaFL",'OPTIONS.','ARGV.','FB.',)
  call XCUA "XtAddCallback",primary,"closeCallback","XCUAFL","EXIT"
```

Analysis of an X-CUA application
(continued)

```
filename = ''                          /* Examine remaining arguments */
if (argv.0 > 1) then                     /* Get file pattern if there */
  do i=2 to argv.0 for 3
    filename = filename argv.i
  end
else
  filename = '* * a'                  /* Otherwise set a good default */

/* ------------------------------------------------------------------- */
/* Set properties of the Presentation area.                            */
/* ------------------------------------------------------------------- */

presentation = XCUA("XcuaPropertyOf",primary,"clientArea")
call SETARG 'ARG.',1,"xScrollBar",FALSE
call SETARG 'ARG.',2,"yScrollBar",FALSE
call XCUA "XtSetValues",presentation,'ARG.'
```

Analysis of an X-CUA application
(continued)

- **Define the interface**
  - — Create a client area
  - — Populate it with objects
  - — Register callbacks
  - — Define secondary windows

Analysis of an X-CUA application
(continued)

```
/* ------------------------------------------------------------------ */
/* Create a Form widget as the client area and configure it.          */
/* ------------------------------------------------------------------ */

  call SETARG 'ARG.',1,"hSpace",10
  call SETARG 'ARG.',2,"vSpace",10
  form = XCUA("XcuaCreateForm","form",presentation,'ARG.')

  call SETARG 'ARG.',1,"chars",50
  call SETARG 'ARG.',2,"inset",TRUE
  command = XCUA("XcuaCreateSingleEntry","command",form,'ARG.')

  prompt = XCUA("XcuaAddFieldPrompt",command,"Command:","C","above")
  call SETARG 'ARG.',1,"below",form
  call SETARG 'ARG.',2,"rightof",form
  call XCUA "XtSetValues",prompt,'ARG.'

  call SETARG 'ARG.',1,"label","Execute"
  call SETARG 'ARG.',2,"mnemonic","X"
  call SETARG 'ARG.',3,"rightof",command
  call SETARG 'ARG.',4,"acceptFocus",FALSE
  execute = XCUA("XtCreateManagedWidget","execute","PushButton",form,'ARG.')
  call XCUA "XtAddCallback",execute,"triggerCallback","XCUAFL","EXECUTE"
  call XCUA "XcuaSetDefaultButton",form,execute
```

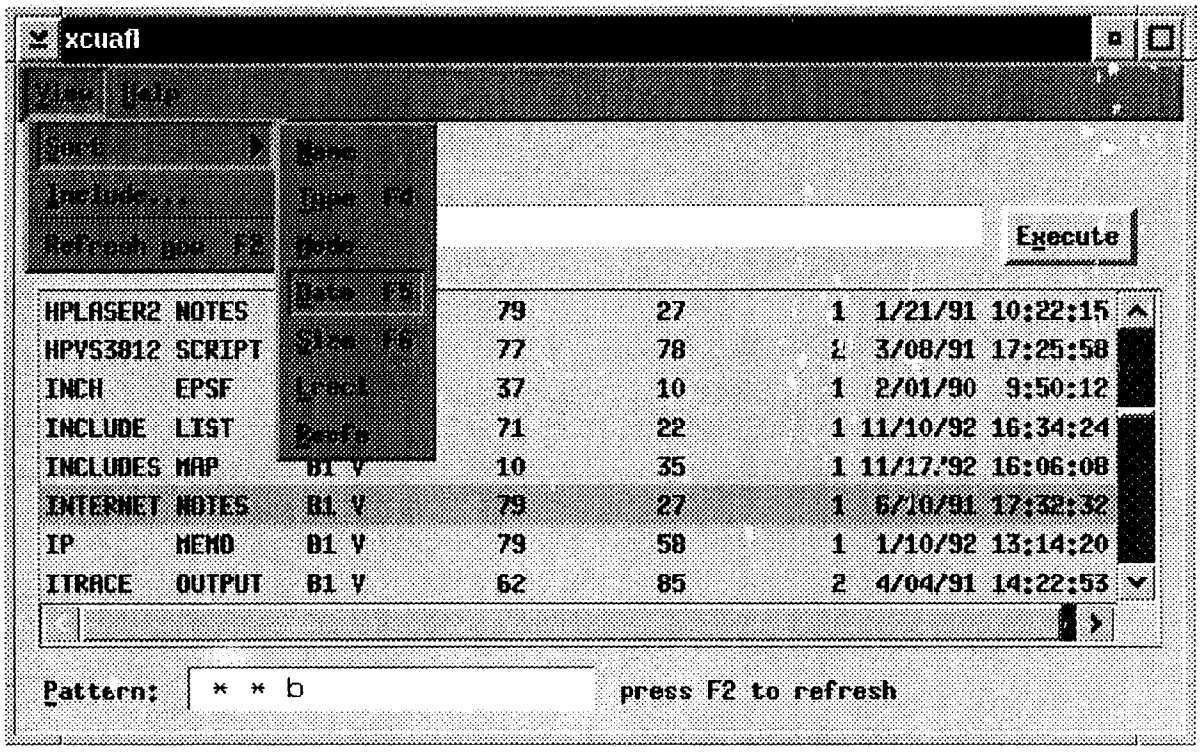
Analysis of an X-CUA application
(continued)

```
background = XCUA("XcuaPropertyOf",form,"background")
call SETARG 'ARG.',1,"below",execute
call SETARG 'ARG.',2,"rightof",form
call SETARG 'ARG.',3,"background",background
list = XCUA("XcuaCreateList","list",form,'ARG.')

call SETARG 'ARG.',1,"chars",21
call SETARG 'ARG.',2,"inset",TRUE
pattern = XCUA("XcuaCreateSingleEntry","pattern",form,'ARG.')
prompt = XCUA("XcuaAddFieldPrompt",pattern,"Pattern:","P","left")
call SETARG 'ARG.',1,"above",form
call SETARG 'ARG.',2,"rightof",form
call XCUA "XtSetValues",prompt,'ARG.'
call XCUA "XcuaAddDescriptiveText",pattern,"press F2 to refresh","right"
call SETARG 'ARG.',1,"above",prompt
call XCUA "XtSetValues",list,'ARG.'

call XCUA "XcuaSetText",pattern,filename
call XCUA "XcuaSetProperty",form,"initialFocus",command
```

Analysis of an X-CUA application
(continued)

```
/* ---------------------------------------------------------------- */
/* Create an Input dialog for the Include... action.                */
/* ---------------------------------------------------------------- */

  call SETARG 'ARG.',1,"message","Enter include file pattern:"
  include = XCUA("XcuaCreateInput","include",primary,'ARG.')
  call XCUA "XtAddCallback",include,"okCallback","XCUAFL","INCLUDE"
  call XCUA "XtAddCallback",include,"cancelCallback",,
    "XcuaCallbackCloseDialog",include
```

Analysis of an X-CUA application
(continued)



Analysis of an X-CUA application
(continued)

```
/* ---------------------------------------------------------------- */
/* Define custom menus for this application.                        */
/* ---------------------------------------------------------------- */

  ACTION = 1
  CASCADE = 2
  SEP = 3

  call SETBUTTONREC 'VIEWBUTTON.',1,CASCADE,"view","View",,"V",,"VIEW."

  call SETBUTTONREC 'VIEW.',1,CASCADE,"sort","Sort",,"S",,"SORT."
  call SETBUTTONREC 'VIEW.',2,ACTION,"include","Include...",,"I",,,
    "XcuaCallbackOpenDialog",include
  call SETBUTTONREC 'VIEW.',3,SEP
  call SETBUTTONREC 'VIEW.',4,ACTION,"refresh_now","Refresh now",,"N","F2",,
    "XCUAFL","REFRESH"

  call SETBUTTONREC 'SORT.',1,ACTION,"name","Name",,"N",,"XCUAFL","SORT NAME"
  call SETBUTTONREC 'SORT.',2,ACTION,"type","Type",,"T",'F4","XCUAFL","SORT TYPE"
  call SETBUTTONREC 'SORT.',3,ACTION,"mode","Mode",,"M",,"XCUAFL","SORT MODE"
  call SETBUTTONREC 'SORT.',4,ACTION,"date","Date",,"D","F5","XCUAFL","SORT DATE"
  call SETBUTTONREC 'SORT.',5,ACTION,"size","Size",,"S","F6","XCUAFL","SORT SIZE"
  call SETBUTTONREC 'SORT.',6,ACTION,"lrecl","Lrecl",,"L",,"XCUAFL","SORT LRECL"
  call SETBUTTONREC 'SORT.',7,ACTION,"recfm","Recfm",,"R",,"XCUAFL","SORT RECFM"
```

Analysis of an X-CUA application
(continued)

```
/* ------------------------------------------------------------------ */
/* Set up the menu bar, create the custom menus.                      */
/* ------------------------------------------------------------------ */

  menubar = XCUA("XcuaMenuBarOf",presentation)
  call XCUA "XcuaGenerateMenu",menubar,'VIEWBUTTON.'
  call XCUA "XcuaMenuBarVisible",menubar,"file",0
  call XCUA "XcuaMenuBarVisible",menubar,"edit",0
  call XCUA "XcuaMenuBarCallback",menubar,"help","about","XcuaProductInfo",,
    "This is XCUAFL, a CUA-compliant"||'15'x||,
    "X-based FILELIST utility."||'15'x||'15'x||,
    "To contact the author, write to:"||'15'x||'15'x||,
    "     Relay Technology, Inc."||'15'x||,
    "     1604 Spring Hill Road"||'15'x||,
    "     Vienna, VA 22182"
```

Analysis of an X-CUA application
(continued)

- **Load the contents of the primary window:**

```
call XCUAFL primary,"REFRESH"
```

- **Make it visible:**

```
call XCUA "XcuaOpenPrimary",primary
```

- **Enter the main event handling loop:**

```
rc = XCUA("XcuaMainLoop",app_context)
exit rc
```

Analysis of an X-CUA application
(continued)

- **And the handler for possible syntax error:**

SYNTAX:

```
say "Syntax error" rc "occured on line" sigl":" errortext(rc)
if (symbol('APP_CONTEXT') <> "LIT") then
  call XCUA "XtDestroyApplicationContext",app_context

exit rc
```

Analysis of an X-CUA application
(continued)

- **Callbacks revisited**

  — X-CUA applications are event-driven, via callbacks
  — The application is called as a command
  — The callback handler is called as a function
  — We use PARSE SOURCE to determine calltype, handle both in the same source file

Analysis of an X-CUA application
(continued)

```
/* ------------------------------------------------------------------ */
/* Callback()                                                         */
/*                                                                    */
/* Function: Handle callbacks                                         */
/*                                                                    */
/* ------------------------------------------------------------------ */

CALLBACK:
parse arg widget,client_data,call_data,callback_id

  valid = "EXIT EXECUTE REFRESH SORT INCLUDE"
  label = word(client_data,1)

  if (find(valid,label) == 0) then
    return
  signal value (label)

  /* ------------------------------------------------------------------ */
  /* Handle an XcNcloseCallback.                                        */
  /* ------------------------------------------------------------------ */

  EXIT:

    call XCUA "XcuaExit",0

  return
```

Analysis of an X-CUA application
(continued)

```
/* ------------------------------------------------------------- */
/* Handle an XcNtriggerCallback for the EXECUTE button.          */
/* ------------------------------------------------------------- */

EXECUTE:

  form = XCUA("XtParent",widget)
  command = XCUA("XcuaGetChild",form,"command")
  list = XCUA("XcuaGetChild",form,"list")
  primary = XCUA("XcuaPrimaryOf",widget)
  display = XCUA("XtDisplay",widget)

  directive = XCUA("XcuaSubstr",command,0,-2)
  call XCUA "XcuaClearText",command

  if (directive = '') then
    return

  item = XCUA("XcuaListItemOf",list,0)
```

Analysis of an X-CUA application
(continued)

```
  do until (item == '00000000'x)
    next = XCUA("XcuaNextItem",item);
    if (XCUA("XcuaItemSelected",item)) then do
      label = XCUA("XcuaItemLabel",item)
      parse var label fname ftype fmode .
      call XCUA "XcuaSetWaiting",primary,TRUE
      call RunCommand directive,fname,ftype,fmode
      call XCUA "XcuaSetWaiting",primary,FALSE
      'ESTATE' fname ftype fmode
      if (rc == 28) then
        call XCUA "XcuaDeleteListItem",item
    end
    item = next
  end

return
```

Analysis of an X-CUA application
(continued)

```
/* ------------------------------------------------------------------ */
/* Handle an XcNtriggerCallback for the REFRESH menu choice.          */
/* ------------------------------------------------------------------ */

REFRESH:

  primary = XCUA("XcuaPrimaryOf",widget)
  presentation = XCUA("XcuaPropertyOf",primary,"clientArea")
  form = XCUA("XcuaScrolledOf",presentation)
  pattern = XCUA("XcuaGetChild",form,"pattern")
  list = XCUA("XcuaGetChild",form,"list")

  call XCUA "XcuaSetProperty",pattern,"error",FALSE
  directive = XCUA("XcuaSubstr",pattern,0,-2)

  parse upper var directive fname ftype fmode
  if (fmode = '') then
    fmode = 'A'
  if (ftype = '') then
    ftype = '*'
  if (fname = '') then
    fname = '*'
```

Analysis of an X-CUA application
(continued)

```
stacked = queued()
   'MAKEBUF'
   'LISTFILE' fname ftype fmode '( NOHEADER DATE FIFO )'
   if (rc ¬== 0) then do
     'DROPBUF'
     call XCUA "XcuaSetProperty",pattern,"error",TRUE
     return
   end

   stacked = queued() - stacked
   call XCUA "XcuaSetWaiting",primary,TRUE
   call XCUA "XcuaClearList",list
   do i=1 to stacked
     parse pull label
     call XCUA "XcuaAddListItem",list,label
   end
   'DROPBUF'
   call XCUA "XtCallActionProc",list,"bod"
   call XCUA "XcuaSetWaiting",primary,FALSE

return
```

Analysis of an X-CUA application
(continued)

```
/* ---------------------------------------------------------------- */
/* Handle a SORT call                                               */
/* ---------------------------------------------------------------- */

SORT:
parse var client_data . field .

  primary = XCUA("XcuaPrimaryOf",widget)
  presentation = XCUA("XcuaPropertyOf",primary,"clientArea")
  form = XCUA("XcuaScrolledOf",presentation)
  list = XCUA("XcuaGetChild",form,"list");

  ASCENDING = TRUE
  DESCENDING = FALSE
  REFRESH = TRUE
  NOREFRESH = FALSE

  select
    when (field == 'NAME') then do
      call XCUA "XcuaSortList",list,ASCENDING,19,20,NOREFRESH
      call XCUA "XcuaSortList",list,ASCENDING,10,17,NOREFRESH
      call XCUA "XcuaSortList",list,ASCENDING,1,8,REFRESH
    end
    when (field == 'TYPE') then do
      call XCUA "XcuaSortList",list,ASCENDING,19,20,NOREFRESH
      call XCUA "XcuaSortList",list,ASCENDING,1,8,NOREFRESH
      call XCUA "XcuaSortList",list,ASCENDING,10,17,REFRESH
    end
```

Analysis of an X-CUA application
(continued)

```
when (field == 'MODE') then do
    call XCUA "XcuaSortList",list,ASCENDING,10,17,NOREFRESH
    call XCUA "XcuaSortList",list,ASCENDING,1,8,NOREFRESH
    call XCUA "XcuaSortList",list,ASCENDING,19,20,REFRESH
  end
  when (field == 'DATE') then do
    call XCUA "XcuaSortList",list,DESCENDING,72,73,NOREFRESH
    call XCUA "XcuaSortList",list,DESCENDING,69,70,NOREFRESH
    call XCUA "XcuaSortList",list,DESCENDING,66,67,NOREFRESH
    call XCUA "XcuaSortList",list,DESCENDING,60,61,NOREFRESH
    call XCUA "XcuaSortList",list,DESCENDING,57,58,NOREFRESH
    call XCUA "XcuaSortList",list,DESCENDING,63,64,REFRESH
  end
  when (field == 'SIZE') then do
    call XCUA "XcuaSortList",list,DESCENDING,35,44,NOREFRESH
    call XCUA "XcuaSortList",list,DESCENDING,46,55,REFRESH
  end
```

Analysis of an X-CUA application
(continued)

```
      when (field == 'RECFM') then do
        call XCUA "XcuaSortList",list,ASCENDING,35,44,NOREFRESH
        call XCUA "XcuaSortList",list,ASCENDING,46,55,NOREFRESH
        call XCUA "XcuaSortList",list,ASCENDING,22,22,REFRESH
      end
      when (field == 'LRECL') then do
        call XCUA "XcuaSortList",list,ASCENDING,19,20,NOREFRESH
        call XCUA "XcuaSortList",list,ASCENDING,10,17,NOREFRESH
        call XCUA "XcuaSortList",list,ASCENDING,1,8,NOREFRESH
        call XCUA "XcuaSortList",list,ASCENDING,24,33,REFRESH
      end
    otherwise
      call XCUA "XcuaInfoMessage",widget,"sort message","Unrecognized",
        "sort type '"field"'"
      return
    end

    call XCUA "XcuaSetPosition",list,0,0

  return
```

Analysis of an X-CUA application
(continued)

```
/* ---------------------------------------------------------------- */
/* Handle an INCLUDE call                                           */
/* ---------------------------------------------------------------- */

INCLUDE:

  call GETCALLBACKEVENT 'EVENT.',call_data
  drop result
  string = GETSTRING(event.result)
  call XCUA "XcuaCloseDialog",widget

  primary = XCUA("XcuaPrimaryOf",widget)
  presentation = XCUA("XcuaPropertyOf",primary,"clientArea")
  form = XCUA("XcuaScrolledOf",presentation)
  pattern = XCUA("XcuaGetChild",form,"pattern")
  list = XCUA("XcuaGetChild",form,"list")

  parse upper var string fname ftype fmode
  if (fmode = '') then
    fmode = 'A'
  if (ftype = '') then
    ftype = '*'
  if (fname = '') then
    fname = '*'
```

Analysis of an X-CUA application
(continued)

```
    stacked = queued()
    'MAKEBUF'
    'LISTFILE' fname ftype fmode '( NOHEADER DATE FIFO )'
    if (rc ¬== 0) then do
      'DROPBUF'
      return
    end

    stacked = queued() - stacked
    call XCUA "XcuaSetWaiting",primary,TRUE
    do i=1 to stacked
      parse pull label
      call XCUA "XcuaAddListItem",list,label
    end
    'DROPBUF'
    call XCUA "XcuaSetWaiting",primary,FALSE

  return
```

Analysis of an X-CUA application
(continued)

```
/* ---------------------------------------------------------------- */
/* RunCommand()                                                     */
/*                                                                  */
/* Function: run a command on a file, with substitution.            */
/*                                                                  */
/* ---------------------------------------------------------------- */
RUNCOMMAND:
parse arg command, fn, ft, fm
  subbed = 0
  index = 1
  do until (index == 0)
    index = pos('/',command,index)
    drop sub
    if (index > 0) then do
      flag = substr(command,index+1,1)
      word = substr(command,index,2)
      upper flag
      if (flag == 'N') then sub = fn
      else if (flag == 'T') then sub = ft
      else if (flag == 'M') then sub = fm
      else if (flag == 'O') then sub = ' '
      else if (flag == ' ') then do
        sub = fn ft fm
        word = '/'
      end
    end
  end
```

Analysis of an X-CUA application
(continued)

```
 if (symbol('SUB') <> "LIT") then do
     subbed = 1
     command = delstr(command,index,length(word))
     command = insert(sub,command,index-1)
     index = index + length(sub)
   end
   else if (index > 0) then
     index = index + 1
  end

  if (¬subbed) then
    command = command fn ft fm

  address CMS command

return
```

X-CUA Requirements

- X Terminal
- VM and/or AIX (currently; other Unix to follow)
- IBM TCP/IP (for VM)
- In Alpha test, GA scheduled for Q3

Summary

- **X-CUA provides an enabling tool to move mainframe-based application user interfaces to a GUI**
  - — CUA-compliant
  - — not vendor-specific
- **Provides true, transparent, seamless interoperability**
- **Maximizes productivity through familiarity, GUI power**

Summary *(continued)*

- **CUA compliance enables standardization, reduced training costs**
- **Legacy systems may be updated with a modern look while preserving corporate investment**
- **Many legacy systems belong on the mainframe; X-CUA allows these to stay there**