X Multiplexor Control Protocol

Version 3.00
May 5, 1998

John Bazik
Computer Science Department
Brown University
jsb@cs.brown.edu
1. Introduction

A network window system separates application programs from a window server by a data stream. The protocol that travels back and forth over that stream controls the graphics that the user sees and the input to which the application responds.

A network window system multiplexor manipulates that stream to allow application programs designed for one display to paint themselves on, and receive input from many of them.

It is useful for such a shared window session to be dynamic, for instance to allow users to join in or drop out, and to control who may provide input at what time. This document describes a general purpose, policy-independent, network protocol for controlling a network window system multiplexor.

2. Protocol

The protocol controls how multiple displays share a single window session made up of one or more client applications. The model is of a meeting at which at any given time there are three types of users: speakers, participants and spectators. Speakers have control over the proceedings; they have the floor. Participants are able to signal or interrupt a speaker; they have a seat. Spectators may watch the meeting only; they have a view.

The terms floor, seat and view are used to express these modes of interaction. A server that has the floor may provide arbitrary input events to client applications. All input events from servers that have a seat are ignored except for one special, agreed-upon event that indicates its wish to be recognized. All input events are ignored from servers that have only a view.

Any number of servers may be in any of these modes. Any server’s mode may be changed to any other mode. Assigning modes and changing them are the province of the application making multiplexor control protocol requests.

Types

The remainder of this document follows the syntactic conventions established in Part 2 of X Window System by Scheifler and Gettys.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISTofFOO</td>
<td>A counted list of elements of type FOO</td>
</tr>
<tr>
<td>BOOL</td>
<td>{True, False}</td>
</tr>
<tr>
<td>BYTE</td>
<td>8-bit value</td>
</tr>
<tr>
<td>CARD16</td>
<td>16-bit unsigned integer</td>
</tr>
<tr>
<td>MASK</td>
<td>32-bit unsigned integer</td>
</tr>
<tr>
<td>WINDOW</td>
<td>32-bit unsigned integer</td>
</tr>
<tr>
<td>TPRTRID</td>
<td>32-bit unsigned integer</td>
</tr>
<tr>
<td>DISPID</td>
<td>32-bit unsigned integer</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>[family: {Internet, DECnet, Chaos} address: LISTofBYTE display: 16-bit integer]</td>
</tr>
<tr>
<td>COOKIE</td>
<td>name: STRING8 data: STRING8</td>
</tr>
<tr>
<td>SCREEN</td>
<td>[DISPLAY real_screen: CARD16 virtual_screen: CARD16]</td>
</tr>
<tr>
<td>CONFIG</td>
<td>foo</td>
</tr>
<tr>
<td>SERVEREVENT</td>
<td>[window: WINDOW or ALL eventmask: MASK]</td>
</tr>
</tbody>
</table>
3. Connection Setup

Connection Initiation
Though an XMCP data stream is not likely to be compute-bound on either end, swapping bytes in accordance with the XMCP client’s preference is easily done since it is already done for X clients, so the convention is retained.

byte_order: BYTE
protocol_major_version: CARD16
protocol_minor_version: CARD16
authorization_protocol_name: STRING8
authorization_protocol_data: STRING8

This is identical to the client connection setup defined in the X Protocol.

Multiplexor Response
Upon successful connection, the multiplexor returns the following.

success: BOOL
cfg_mode: BYTE
base: CARD32
mask: CARD32
default_telepointer: TPTRID

Base and mask are used by XMC clients to construct valid resource ids. If the client connection block is not accepted, the multiplexor returns the following.

success: BOOL
length: CARD16
reason: STRING

4. Requests

Register

regid: REGID
family: BYTE
address: STRING8
port: CARD16
name: STRING8
url: STRING8
desc: STRING8

Register this session with the directory service at (family, address, port), as (name, desc) and directing users to the url for connection information.

Unregister

id: REGID

Unregister this session.
**SetAuth**

pmask: CARD32  
cookie: COOKIE  

Set the permissions of the cookie (name, data) to those specified in pmask.

**GetAuth**

cookie: COOKIE  
->  
pmask: CARD32  

Get the permissions associated with the cookie (name, data).

**AddDisplay**

dpid: DISPID  
display: DISPLAY  
cookie: COOKIE  
geometry: STRING8  
window: WINDOW or NONE  
mode: {Floor, Seat, View}  
tag: STRING8  
telepointer: TPTRID  

Errors: ErrDisplay, ErrConfig  

Add a display to the session.

**SetDisplayTag**

dpid: DISPID  
tag: STRING8  

Set the display’s tag string.

**QueryDisplay**

dpid: DISPID  
->  
display: DISPLAY  
mode: {Floor, Seat, View}  
tag: STRING8  
telepointer: TPTRID  
window: WINDOW  

Get information about the display.
ListDisplays

->
displays: LISTofDISPID

Get a list of all displays.

ListDisplaysWithInfo

->++
dpid: DISPID
display: DISPLAY
mode: {Floor, Seat, View}
tag: STRING8
telepointer: TPTRID
window: WINDOW

Get a list of all displays.

DropDisplay

dpid: DISPID

Errors: ErrDisplay

Drop a display from the session. The display is immediately removed from participation.

Sync

->

Sends a round-trip request to each X server and waits for all replies before replying itself.

SetConfig

width: CARD16
height: CARD16
depths: LISTofDEPTHTYPE
extensions: LISTofSTRING8

Set the current virtual configuration.

GetConfig

->
width: CARD16
height: CARD16
depths: LISTofDEPTHTYPE
extensions: LISTofSTRING8

Get the current virtual configuration.
**SetConfigMode**

mode: [Allow, Delay]

If the virtual configuration has not been fixed, the config mode specifies whether DisplayAdd requests should be processed immediately or postponed. If postponed, they will be processed all at once when the config mode is again set to Allow. If the virtual configuration is fixed, all DisplayAdd requests are processed immediately, irrespective of the config mode. Changing the ConfigMode causes a ConfigModeEvent to be sent to all XMC clients.

**ChangeInputMode**

dpid: DISPID
mode: [Floor, Seat, View]

Errors: ErrDisplay

Changes the input mode of the given display. In Floor mode, all input is fed to client applications. In Seat mode, selected input events generate XMCP events. In View mode, all input is ignored.

**SetEventMask**

mask: MASK

Selects the multiplexor events (described below) to receive.

**GetEventMask**

->

mask: MASK

Retrieves the multiplexor events (described below) to receive.

**SetXEventMask**

window: WINDOW
mask: MASK

Selects the X events (described below) to receive.

**GetXEventMask**

->

window: WINDOW
mask: MASK

Retrieves the X events (described below) to receive.

**GrabPointer**

dpid: DISPID

Errors: ErrDisplay
Causes the pointer to be grabbed whenever it is contained within the multiplexor’s virtual root window on the given display. Causes PointerGrab, PointerNoGrab and PointerUngrab events to be generated in response to pointer movements.

**UngrabPointer**

dpid: DISPID

Error: ErrDisplay

Release a display from a pointer grab.

**GrabKeyboard**

dpid: DISPID

Errors: ErrDisplay

Causes the keyboard to be grabbed whenever the multiplexor’s virtual root window has the keyboard focus on the given display. Causes KeyboardGrab, KeyboardNoGrab and KeyboardUngrab events to be generated in response to pointer movements.

**UngrabKeyboard**

dpid: DISPID

Error: ErrDisplay

Release a display from keyboard grab.

**ShareSelections**

dpid: DISPID

Causes selections to exist seamlessly between the virtual shared session of the multiplexor and the local X session of the given display.

**UnshareSelections**

dpid: DISPID

Causes selection sharing with the given display to stop.

**CreateTptr**

telepointer: TPTRID
mask: BITMASK
sourceID: PIXMAP
maskID: PIXMAP or None
hotX, hotY: INT16
foreRed, foreGreen, foreBlue: CARD16
backRed, backGreen, backBlue: CARD16
Create a new telepointer. The telepointer mimics the behavior of pointers on displays to which it is assigned. The telepointer is visible only when it is assigned to a display, and only on displays other than the one providing input to it at any moment. By default, the telepointer looks exactly like the pointer on the X display.

**CreateGlyphTptr**

- telepointer: TPTRID
- mask: BITMASK
- sourceID: FONT
- maskID: FONT or None
- sourceChar, maskChar: INT16
- foreRed, foreGreen, foreBlue: CARD16
- backRed, backGreen, backBlue: CARD16

Create a new telepointer. The telepointer mimics the behavior of pointers on displays to which it is assigned. The telepointer is visible only when it is assigned to a display, and only on displays other than the one providing input to it at any moment. By default, the telepointer looks exactly like the pointer on the X display.

**DestroyTptr**

- telepointer: TPTRID

Destroy a telepointer. The default telepointer may not be destroyed. Any displays to which the telepointer is assigned will revert to using the default telepointer.

**AssignTptr**

- telepointer: TPTRID
- dpid: DISPID

Assign a telepointer to a display.

**HideTptr**

- telepointer: TPTRID

Hide a telepointer.

**ShowTptr**

- telepointer: TPTRID

Show a telepointer.

5. Events

**DisplayIn**

- dpid: DISPID

The given display was successfully added.
**DisplayRefused**

dpid: DISPID

The given display was not added.

**DisplayOut**

dpid: DISPID

The given display has left the session.

**ModeFloor**

dpid: DISPID

The given display’s input mode was changed to Floor.

**ModeSeat**

dpid: DISPID

The given display’s input mode was changed to Seat.

**ModeView**

dpid: DISPID

The given display’s input mode was changed to View.

**PointerGrab**

dpid: DISPID

The pointer on the given display has been actively grabbed by the multiplexor.

**PointerNoGrab**

dpid: DISPID

The pointer on the given display could not be grabbed by the multiplexor.

**PointerUngrab**

dpid: DISPID

The pointer grab on the given display has been released by the multiplexor.

**KeyboardGrab**

dpid: DISPID
The keyboard on the given display has been actively grabbed by the multiplexor.

**KeyboardNoGrab**

dpid: DISPID

The keyboard on the given display could not be grabbed by the multiplexor.

**KeyboardUngrab**

dpid: DISPID

The keyboard grab on the given display has been released by the multiplexor.

**ShareSelections**

dpid: DISPID

Selections are being shared with the given display.

**UnshareSelections**

dpid: DISPID

Selections are no longer being shared with the given display.

**TptrAssign**

telepointer: TPTRID
dpid: DISPID

The given telepointer has been assigned to the given display.

**TptrHide**

telepointer: TPTRID

The given telepointer has been hidden.

**TptrShow**

dpid: DISPID

The given telepointer has been unhidden.

**ConfigMode**

mode: {Allow, Delay}

The multiplexor’s config mode has changed. Clients may not express disinterest in this event.
**ButtonPressed**  
**ButtonReleased**

- dpid: DISPID  
- state:  
- time:  
- event: WINDOW  
- child: WINDOW  
- root_x: INT16  
- root_y: INT16  
- event_x: INT16  
- event_y: INT16

**KeyPressed**  
**KeyReleased**

- dpid: DISPID  
- state:  
- time:  
- event: WINDOW  
- child: WINDOW  
- root_x: INT16  
- root_y: INT16  
- event_x: INT16  
- event_y: INT16  
- detail: BYTE

### 6. Errors

**ErrDisplay**

- display: DISPLAY

For AddDisplay, display connection failed. For all others, display is not active.

**ErrConfig**

- display: DISPLAY

An incompatible server configuration caused a merge or a map to fail.

**ErrMerge**

- display: DISPLAY

The virtual configuration was frozen (due to client activity) so the merge of the given display failed.

**ErrEvent**

- ?
The event is not supported by the virtual server configuration.

**ErrAlloc**

?  
The multiplexor was unable to allocate memory while servicing an XMC request.

**ErrConnect**

?  
Could not connect?

**ErrTelepointer**

?  
Telepointer error [?]

**ErrId**

id: ID[?]

Bad id.

7. Protocol Encoding

**Connection Setup**

1  
\#x42 MSB first
\#x6C LSB first

1  
unused

2  
CARD16  
protocol-major-version

2  
CARD16  
protocol-minor-version

2  
n  
length of authorization-protocol-name

2  
d  
length of authorization-protocol-data

n  
STRING8  
authorization-protocol-name

d  
STRING8  
authorization-protocol-data

q  
unused, q=pad(d)

**Response If Connect Failed**

1  
0  
success

1  
unused

2  
CARD16  
length of reason

12  
unused

d  
STRING8  
reason

q  
unused, q=pad(d)

**Response If Connect Succeeded**

1  
1  
success
config_mode
unused
base
mask
telepointer-id

Requests

Register
1 2 opcode
1 CARD8 family
2 5+(a+n+u+d+p)/4 length
4 DISPID registration-id
2 CARD16 port
2 a length-of-address
2 n length-of-name
2 u length-of-url
2 d length-of-description
 unused
a ADDRESS session-server-address
n STRING8 name
u STRING8 url
d STRING8 description
 unused, p=pad(a+n+u+d)

Unregister
1 3 opcode
1 unused
2 2 length
4 DISPID registration-id

SetAuth
1 4 opcode
1 unused
2 3+(n+d+p)/4 length
4 PMASK permissions-mask
2 n length-of-name
2 d length-of-data
 n STRING8 auth-protocol-name
d STRING8 auth-protocol-data
 unused, p=pad(n+d)

GetAuth
1 5 opcode
1 unused
2 2+(n+d+p)/4 length
2 n length-of-name
2 d length-of-data
 n STRING8 auth-protocol-name
d STRING8 auth-protocol-data
 unused, p=pad(n+d)
AddDisplay
1   6                  opcode
1   CARD8             input-mode
2  8+(a+g+t+n+d+p)/4  length
4  DISPID             display-id
2  CARD16             screen-number
1  CARD8              address-protocol-family
1  CARD8              config-mode
2  a                  length-of-address
2  CARD16             display-number
4  WINDOW             window
2  g                  length-of-geometry
2  t                  length-of-tag
2  n                  length-of-name
2  d                  length-of-data
4  TPTRID             telepointer-id
   0  None
   a  ADDRESS          x-server-address
   g  STRING8          geometry
   t  STRING8          tag-name
   n  STRING8          auth-protocol-name
   d  STRING8          auth-protocol-data
   p        unused, p=pad(a+g+t+n+d)

DropDisplay
1   2                  opcode
1   2                  unused
2  CARD16             length
4  DISPID             display-id

******this document is not up-to-date beyond this point******

QueryDisplay
1   CARD8              code
1   CARD8              pad0
2  CARD16             length
4  DISPID             dispID
   ->
1  CARD8              reply
1  CARD8              address-protocol-family
2  CARD16             sequence-number
4  CARD32             length
2  n                  length-of-address
2  CARD16             display-number
1  CARD8              mode
1  CARD8              how
2  t                  length-of-tag
4  TPTRID             telepointer-id
n  ADDRESS           x-server-address
q unused, q=pad(n)
t  STRING8         tag-name
r unused, r=pad(t)
**ListDisplays**

1 CARD8 opcode
1 unused
2 CARD16 length
->
1 CARD8 reply
1 unused
2 CARD16 sequence-number
4 CARD32 length
2 n number-of-displays
2 unused
4n LISTofDISPID all-displays

**ListDisplaysWithInfo**

1 CARD8 opcode
1 unused
2 CARD16 length
->+
1 CARD8 reply
1 CARD8 address-protocol-family
2 CARD16 sequence-number
4 CARD32 length
4 CARD32 display-id
0 Last Reply
2 n length-of-address
2 CARD16 display-number
1 CARD8 mode
1 CARD8 how
2 t length-of-tag
4 TPTRID telepointer-id
2 CARD16 count
n ADDRESS x-server-address
q unused, q=pad(n)
t STRING8 tag-name
r unused, r=pad(t)

**AddScreen**

1 CARD8 opcode
1 unused
2 CARD16 length
4 DISPID display-id
4 CARD8 address-protocol-family
1 unused
2 n length-of-address
2 CARD16 display-number
2 unused
2 CARD16 real-screen
2 CARD16 virtual-screen
n ADDRESS x-server-address
q unused, q=pad(n)

**DropScreen**

1 CARD8 opcode
1 CARD16 unused
2 CARD16 length
4 DISPID display-id
2 CARD16 virtual-screen

Reset
1 CARD8 opcode
1 CARD16 recalculate-flag
2 CARD16 length

SetVConfig
1 CARD8 opcode
1 CARD16 unused
2 CARD16 length

GetVConfig
1 CARD8 opcode
1 CARD16 unused
2 CARD16 length
->
1 CARD8 reply
1 CARD16 unused
2 CARD16 sequence-number
4 CARD32 length

ChangeInputMode
1 CARD8 opcode
1 CARD8 mode
2 CARD16 length
4 DISPID display-id

Grab
1 CARD8 opcode
1 CARD16 unused
2 CARD16 length
4 DISPID display-id

Ungrab
1 CARD8 opcode
1 CARD16 unused
2 CARD16 length
4 DISPID display-id

SetEventMask
1 CARD8 opcode
1 CARD16 unused
2 CARD16 length
4 CARD32 mask
GetEventMask
1  CARD8  opcode
1  unused
2  CARD16  length
->
1  CARD8  reply
1  unused
2  CARD16  sequence-number
4  CARD32  length
4  CARD32  mask

SetHandUp
1  CARD8  opcode
1  unused
2  CARD16  length
4  CARD32  window
2  CARD16  xmask
2

GetHandUp
1  CARD8  opcode
1  unused
2  CARD16  length
4  CARD32  window
->
1  CARD8  reply
1  unused
2  CARD16  sequence-number
4  CARD32  length
2  CARD16  xmask
2

CreateTptr
1  CARD8  opcode
1  CARD8  mask
2  CARD16  length
4  TPTRID  telepointer-id
4  CARD32  source-id
4  CARD32  mask-id
2  INT16  x-hot-spot
2  INT16  y-hot-spot
2  CARD16  red-foreground
2  CARD16  green-foreground
2  CARD16  blue-foreground
2  CARD16  red-background
2  CARD16  green-background
2  CARD16  blue-background

CreateGlyphTptr
1  CARD8  opcode
1    CARD8                     mask
2    CARD16                    length
4    TPTRID                    telepointer-id
4    CARD32                   source-id
4    CARD32                   mask-id
2    CARD16                   source-character
2    CARD16                   mask-character
2    CARD16                   red-foreground
2    CARD16                   green-foreground
2    CARD16                   blue-foreground
2    CARD16                   red-background
2    CARD16                   green-background
2    CARD16                   blue-background

**DestroyTptr**
1    CAR*8                     opcode
1                              unused
2    CARD16                    length
4    TPTRID                    telepointer-id

**ShowTptr**
1    CARD8                     opcode
1                              unused
2    CARD16                    length
4    TPTRID                    telepointer-id

**HideTptr**
1    CARD8                     opcode
1                              unused
2    CARD16                    length
4    TPTRID                    telepointer-id

**RecolorTptr**
1    CARD8                     opcode
1                              unused
2    CARD16                    length
4    TPTRID                    telepointer-id
2    CARD16                   red-foreground
2    CARD16                   green-foreground
2    CARD16                   blue-foreground
2    CARD16                   red-background
2    CARD16                   green-background
2    CARD16                   blue-background

**AssignTptr**
1    CARD8                     opcode
1                              unused
2    CARD16                    length
4    DISPID                    display-id
4    TPTRID                    telepointer-id
ChangeProperty
1   CARD8  
opcode
1   
unused
2   CARD16  
length

GetProperty
1   CARD8  
opcode
1   
unused
2   CARD16  
length
->
1   CARD8  
reply
1   
unused
2   CARD16  
sequence-number
4   CARD32  
length

ListProperties
1   CARD8  
opcode
1   
unused
2   CARD16  
length
->
1   CARD8  
reply
1   
unused
2   CARD16  
sequence-number
4   CARD32  
length

Sync
1   CARD8  
opcode
1   
unused
2   CARD16  
length
->
1   CARD8  
reply
1   
unused
2   CARD16  
sequence-number
4   CARD32  
length

Events

HandUp
1   CARD8  
   code
1   CARD8  
   same-screen
2   CARD16  
   sequence-number
4   DISPID  
   display-id
1   CARD8  
   type
1   CARD8  
   detail
2   CARD16  
   state
4   CARD32  
   time
4   CARD32  
   root
4   CARD32  
   event
4   CARD32  
   child
2   INT16  
   root-x
2   INT16  
   root-y
2   INT16  
   event-x
INT16 event-y

Display
1 CARD8 code
1 CARD8 inout
2 CARD16 sequence-number
4 DISPID display-id

Reset
1 CARD8 code
1 CARD8 recal
2 CARD16 sequence-number

InputMode
1 CARD8 code
1 CARD8 mode
2 CARD16 sequence-number
4 DISPID display-id

AssignTptr
1 CARD8 code
1 unused
2 CARD16 sequence-number
4 TPTRID telepointer-id
4 DISPID display-id

HideTptr
1 CARD8 code
1 CARD8 mode
2 CARD16 sequence-number
4 TPTRID telepointer-id

Property
1 CARD8 code
1 CARD8 mode
2 CARD16 sequence-number
4 DISPID display-id

Errors

ErrDisplay
1 CARD8 error
1 CARD8 error-code
2 CARD16 sequence-number
1 CARD8 request-code
3 CARD8 unused
4 CARD32 data

ErrConfig
1 CARD8 error
1 CARD8 error-code
2 CARD16
1 CARD8
3
4 CARD32

ErrMerge
1 CARD8
1 CARD8
2 CARD16
1 CARD8
3
4 CARD32

ErrEvent
1 CARD8
1 CARD8
2 CARD16
1 CARD8
3
4 CARD32

ErrAlloc
1 CARD8
1 CARD8
2 CARD16
1 CARD8
3
4 CARD32

ErrConnect
1 CARD8
1 CARD8
2 CARD16
1 CARD8
3
4 CARD32

ErrTelepointer
1 CARD8
1 CARD8
2 CARD16
1 CARD8
3
4 CARD32

ErrId
1 CARD8
1 CARD8
2 CARD16
1 CARD8
3
4 CARD32
**Connection Close**

When the multiplexor or the multiplexor client severs an XMC protocol connection, all display connections, state and resources associated with the client are retained.