Working Implementation Agreements for Open Systems Interconnection Protocols: Part 14 - Virtual Terminal

Output from the September 1993 Open Systems Environment Implementors' Workshop (OIW)

SIG Chair: **Michelle Conaway, HFSI**
SIG Editor: **Scott Wattum, Digital Equipment Corp.,** Workshop Editor: **Brenda**

Gray

Foreword

This part of the Working Implementation Agreements was prepared by the Virtual Terminal Special Interest Group (VTSIG) of the Open Systems Environment Implementors' Workshop (OIW). See Procedures Manual for Workshop charter.

Text in this part has been approved by the Plenary of the above-mentioned Workshop. This part replaces the previously existing chapter on this subject.

Only the pages that were changed in December 1992 are being printed. Please refer to the September 1992 Working Document for additional information.

Three normative annexes are given.

Future changes and additions to this version of these Implementor Agreements will be published as a new part. Deleted and replaced text will be shown as strikeouts. New and replacement text will be shown as shaded.

Table of Contents

Part 14 ISO Virtual Terminal Protocol

Editor's Note - References to Stable Agreements in this part refer to Version 5.

0 Introduction

See Stable Agreements.

1 Scope

1.1 Phase Ia agreements

See Stable Agreements.

1.2 Phase Ib agreements

See Stable Agreements regarding Forms profile.

The Scroll profile is intended to support line-at-a-time applications and has colour and text attribute capabilities.

1.3 Phase II agreements

See Stable Agreements regarding X.3 profile, Generalized Telnet profile and the S-mode Paged Application Profile.

2 Normative references

See Stable Agreements.

3 Status

These agreements are being done in phases. Below is the current status of each phase.

3.1 Status of phase Ia

The Phase Ia Agreements, which include the profiles for Telnet and Transparent operation, are complete and were stabilized in May, 1988. See Stable Agreements.

3.2 Status of phase Ib

The Forms profile of Phase 1b was stabilized in December, 1988. Alignment with EWOS Forms profile was achieved in September, 1989. See Stable Agreements.

3.3 Status of phase II

Phase II is still in progress and includes the remaining profile work for the Scroll profile.

The S-mode Paged Application Profile is being progressed as PDISP 11187-2 (AVT-23 S-mode Paged Application Profile).

The X.3 profile was stabilized in December 15, 1989.

The Generalized Telnet profile was stabilized in December 13, 1991.

It is intended that Phase II agreements be compatible with Phase I agreements.

4 Errata

See Stable Agreements.

5 Conformance

See Stable Agreements.

6 Protocol

See Stable Agreements.

7 OIW registered control objects

7.1 Sequenced Application (SA)

See Stable Agreements.

7.2 Unsequenced Application (UA)

See Stable Agreements.

7.3 Sequenced Terminal (ST)

See Stable Agreements.

7.4 Unsequenced Terminal (UT)

See Stable Agreements.

7.5 Termination Conditions CO (TC)

This CO is an instance of the standard type TCCO, as defined in ISO 9040. It is initially designed for use with the OIW Scroll VT profile, though as a registered CO it is available for use by other VT profiles.

In addition to the three standardized data elements, it provides a definition and update syntax for further types of Termination Condition. Each additional type is available for use in additional data elements of the CO. The number and type of such additional data elements is defined in the profile using this CO.

7.5.1 Entry number

To be supplied by the Registration Authority.

7.5.2 Name of sponsoring body

OSI Implementors' Workshop for Implementors of OSI, VTSIG.

7.5.3 Date

The date of submission of this proposal is September 15, 1989.

7.5.4 Identifier

```
oiw-vt-co-tcco-tc  OBJECT IDENTIFIER ::= { oiw-vt-co-tcco    tc(0) }
```

7.5.5 Descriptor value

"OIW VT CO for Termination Conditions"

7.5.6 CO VTE-parameters

```
CO-structure    =  ,      *(not defined in this registration, see note 1 in 14.7.5.8)*
CO-priority     = "normal"
        {
        CO-element-id  = 1,      *(termination length)*
        CO-category    = "integer",
        CO-size        = 65535 },
        {
        CO-element-id  = 2, *(time-out mantissa)*
        CO-category    = "integer",
        CO-size        = 65535 },
```

```
        {
        CO-element-id  = 3, *(time-out exponent)*
        CO-category    = "integer",
        CO-size        = 65535 },
```

(the following represents possibly multiple invocations of a generic data element type, according to the value of CO-structure for the instance of this CO.)

```
        FOR N=4 to CO-structure
        {
        CO-element-id  = N,    *(acts as integer identifier for the events in this element)*
        CO-category    = "transparent",
        CO-size        =       *(not defined in this registration, see note 2 in 14.7.5.8)* }
```

7.5.7 CO values, semantic and update syntax

The value fields for data elements 1,2 and 3 are defined in ISO 9040.

The value field for each additional data element is defined by the following ASN.1 construct which also defines the update syntax.

```
TermCondList  ::= SEQUENCE OF CHOICE {
              void                  [0] IMPLICIT NULL,
              x3ForwardingCond      [1] IMPLICIT INTEGER,
              stEventList           [2] IMPLICIT Range,
              anySTUpdate           [3] IMPLICIT NULL,
              stEventMasks          [4] IMPLICIT MaskValues,
              dOChars               [5] IMPLICIT DOCharacters }

Range         ::= SEQUENCE OF SEQUENCE {
                                    [1] IMPLICIT LogEvent,
                                    [2] IMPLICIT LogEvent OPTIONAL }
-- each pair represents an interval of values as defined for the value field of
--CO ST, see 14.7.3.7.  The second value in each pair shall not be smaller than
--the first value.  If the second value is omitted, the interval contains only
--the specified first value.

LogEvent      ::= INTEGER
-- values as defined for value field of CO ST, see 14.7.3.7.

MaskValues    ::= SEQUENCE OF SEQUENCE {
              mask                  [1] IMPLICIT LogEvent,
              value                 [2] IMPLICIT LogEvent }

DOCharacters  ::= SEQUENCE OF SEQUENCE {
                                    [1] IMPLICIT Repref,
                                    [2] IMPLICIT INTEGER,
                                    [3] IMPLICIT INTEGER OPTIONAL }

Repref        ::= INTEGER
-- index to the list of repertoires for the Display Object
```

7.5.8 Additional information

NOTE - The value of CO-structure is defined in the profile to be the number of types of termination conditions available for use within the profile.

NOTE - The value of CO-size for each additional data element of this CO must be defined within the profile definition which uses those additional termination conditions.

7.5.9 Usage

Defined in profile.

8 OIW defined VTE-profiles

8.1 Telnet profile

See Stable Agreements.

8.2 Transparent profile

See Stable Agreements.

8.3 Forms profile

See Stable Agreements.

8.4 X3 profile

See Stable Agreements.

8.5 Generalized Telnet profile

See Stable Agreements

8.6 Scroll profile

OIW VTE-Profile Scroll-1989 (r1,r2,...r9)

8.6.1 Introduction

This Scrolling A-mode VTE-profile is designed to support line-at-a-time interactions between a terminal and a host system, the type of operation typified by operating system command entry.

Scrolling is bi-directional, forward and backward.

The profile also provides a facility for switching local echo "on" or "off".

This VTE-Profile supports what is often referred to as "type-ahead", so input from the terminal user is available to the host application as soon as the application is ready for input, thus providing efficiency by minimizing communication delays.

This VTE-profile supports the definition of "input" termination events by the "Application VT-user" so the application can specify what events will cause "input" data to be forwarded to the "Application VT-user".

8.6.2 Association requirements

8.6.2.1 Functional units

The Urgent Data Functional Unit is optional, and will be used if available.

8.6.2.2 Mode

This profile operates in A-mode.

8.6.3 Profile body

```
Display-objects =
{
        {
        display-object-name = DOA,
        DO-access = profile-argument-rl,
        dimension = "two",
                x-dimension =
                {
                        x-bound = profile-argument-r2,
                        x-addressing = "no-constraint",
                        x-absolute = "no",
```

```
                    x-window = x-bound
          },
          y-dimension =
          {
                    y-bound = "unbounded",
                    y-addressing = "no-constraint",
                    y-absolute = "no",
                    y-window = profile-argument-r10
          },

erasure-capability = "yes",

*( repertoire-capability is implied by the number of occurrences of profile-argument-r4 )*

repertoire-assignment = profile-argument-r4,

DO-emphasis = profile-argument-r5,

foreground-colour-capability = profile-argument-r6,
foreground-colour-assignment = profile-argument-r7,
background-colour-capability = profile-argument-r6,
background-colour-assignment = profile-argument-r8
},
{
display-object-name = DOB,
DO-access = opposite of profile-argument-rl,
dimension = "two",
          x-dimension =
          {
                    x-bound = profile-argument-r2,
                    x-addressing = "no-constraint",
                    x-absolute = "no",
                    x-window = x-bound
          },
          y-dimension =
          {
                    y-bound = "unbounded",
                    y-addressing = "higher only",
                    y-absolute = "no",
                    y-window = 1
          },
erasure capability = "yes",
*( repertoire-capability is implied by the number of occurrences of profile-argument-r4 )*

repertoire-assignment = profile-argument-r4,

DO-emphasis = profile-argument-r5,

foreground-colour-capability = profile-argument-r6,
```

```
        foreground-colour-assignment = profile-argument-r7,
        background-colour-capability = profile-argument-r6,
        background-colour-assignment = profile-argument-r8
        }
},

Control-objects =
{
        {
        CO-name                 = E,    *(standard Echo CO)*
        CO-type-identifier      = vt-b-sco-echo,
        CO-access               = profile-argument-r1,
        CO-priority             = "normal",
        CO-trigger              = "selected",
        CO-category             = "boolean",
        CO-size                         = 1
        },
        IF r9 = "TE" THEN
        {
        CO-name                 = TE, *(Termination Event CO)*
        CO-type-identifier      = vt-b-sco-tco,
        CO-access               = opposite of profile-argument-r1,
        CO-priority             = "normal",
        CO-trigger              = "selected",
        CO-category             = "integer"
        },
                {
        CO-name                 = SA, *(NIST Registered CO)*
        CO-type-identifier      = nist-vt-co-misc-sa,
        CO-access               = profile-argument-r1,
        CO-priority             = "normal",
        CO-trigger              = "not selected",
        CO-category             = "integer",
        CO-size                 = 65535
        },
                        {
        CO-name                 = UA, *(NIST Registered CO)*
        CO-type-identifier      = nist-vt-co-misc-ua,
        CO-access               = profile-argument-r1,
        CO-priority             = "urgent",
        CO-category             = "integer",
        CO-size                 = 65535
        },
                        {
        CO-name                 = ST, *(NIST Registered CO)*
        CO-type-identifier      = nist-vt-co-misc-st,
        CO-access               = opposite of profile-argument-r1,
        CO-priority             = "normal",
        CO-category             = "integer",
```

```
        CO-size                 = 65535
        },

        {
        CO-name                 = UT, *(NIST Registered CO)*
        CO-type-identifier      = nist-vt-co-misc-ut,
        CO-access               = opposite of profile-argument-r1,
        CO-priority             = "urgent",
        CO-category             = "integer",
        CO-size                 = 65535
        },
        {
        CO-name                 = TC, *(Termination conditions CO)*
        CO-type-identifier      = nist-vt-co-tcco-tc,
        CO-structure            = N, *( defined with TCCO)*
        CO-access               = profile-argument-r1,
        CO-priority             = "normal",
                {
                CO-element-id   = 1, *(termination length)*
                CO-category     = "integer",
                CO-size         = 65535 },
                {
                CO-element-id   = 2, *(time-out mantissa)*
                CO-category     = "integer",
                CO-size         = 65535 },
                {
                CO-element-id   = 3, *(time-out exponent)*
                CO-category     = "integer",
                CO-size         = 65535 },
                {
                CO-element-id   = 4-N, *(from registered TCCO)*
                CO-category     = ???,
                CO-size         = ??? }
The NIST Workshop VT SIG is defining this registered TCCO.  This
TCCO is a reference to that registered control object.
        }
}

        Device-objects =
        {
                {
                device-name = DVA,     *("output" device object)*
                device-default-CO-access = profile-argument-rl,
                device-default-CO-initial-value = 1."true",
                device-display-object = DOA,
                device-minimum-X-array-length = profile-argument-r2,
                device-minimum-Y-array-length = profile-argument-r3,
                device-control-object = {SA,UA}
                },
```

```
        {
        device-name = DVB,     *("input" device object)*
        device-default-CO-access = opposite of profile-argument-r1,
        device-default-CO-initial-value = 1."true",
        device-display-object = DOB,
        device-minimum-X-array-length = profile-argument-r2,
        device-control-object = profile-argument-r9,
        device-control-object = {ST,UT},
        device-control-object = TE
        }
},

type-of-delivery-control = "simple-delivery-control".
```

8.6.4 Profile argument definitions

r1 - is mandatory and enables negotiation of which VT-user has update access to display object DOA. It takes values "WACI", "WACA". It implies the asymmetric roles of the VT-users as "Application VT-user" and "Terminal VT-user". If the value for DOA is "WACI", then the association initiator is the "Application VT-user"; if the value of DOA is "WACA", then the association initiator is the "Terminal VT-user". This profile argument is also used to determine which VT-user has access to other VT objects as described above. Reference in the profile definition to "opposite of profile-argument-r1" means that the alternative of the two possible values for profile- argument-r1 is to be used. This argument is identified by the identifier for DO-access for display object DOA.

r2 - is optional and enables negotiation of a value for the VTE-parameter x-bound for the display objects DOA and DOB. It takes an integer value greater than zero. This argument is identified by the identifier for x-bound for display object DOA. Default is 80.

r3 - is optional and enables the negotiation of a value for the VTE-parameter device-minimum-Y-array-length for device object DVA. It takes an integer value greater than zero; if absent, a device of any length will be satisfactory.

NOTE - Indicates screen length.

r4 - is optional and provides for the negotiation of value(s) for the VTE-parameter repertoire-assignment. The value of repertoire-capability is implied by the number of occurrences of this argument. Default is specified by 9040.

r5 - is optional and provides for the negotiation of a value for the VTE-parameter DO-emphasis. The default value is that given in ISO 9040, B.17.3. Refer to ISO 9040 B.17.4 for rules governing the selection of non-default values.

r6 - is optional and provides for the negotiation of value(s) for VTE-parameters foreground-colour-capability and background-colour-capability. Default is 8.

r7 - is optional and provides for the negotiation of a value for VTE-parameter foreground-colour-assignment. Default is {"white", "black", "red", "cyan", "blue", "yellow", "green", "magenta"}.

r8 - is optional and provides for the negotiation of a value for VTE-parameter background-colour-assignment. Default is {"black", "white", "cyan", "red", "yellow", "blue", "magenta","green"}.

r9 - is optional and enables negotiation of a termination control object. The value for this argument is the value of CO-name for the termination control object, i.e. "TE"; if absent, no termination control is defined.

r10 - is optional and provides for the negotiation of a value for the VTE-parameter y-window of the DOA Display Object. Default is 24.

8.6.5 Profile dependent CO information

This profile makes use of five OIW registered Control Objects, SA, UA, ST, UT and TCCO. The CO-access in each CO is defined within this profile.

8.6.6 Profile notes

8.6.6.1 Definitive notes

Only the first boolean of the default control object contained in each device object is defined. This boolean is defined as the "on/off" switch for the device where the value "true" ="on" and "false" = "off". These values were chosen so the initial value of the boolean, "true", means the device is initially "on" and data to/from the display objects is being mapped to the device.

Only one boolean is defined in the standard echo control object, E. The semantics of this boolean is defined such that "false" means "local echo off" and "true" means "local echo on"; these values were chosen so echoing is initially "off" (which would provide security when a password is entered at the start of a terminal session).

8.6.6.2 Informative notes

This profile models a scrolling device which is capable of scrolling both forwards and backwards. The display pointer may be moved backwards to modify earlier lines. A typical use for this profile is for applications where type-ahead may be advantageous and control over local echo "on"/"off" is required, e.g. the type of application where a conventional teletypewriter device or ‘teletype-compatible’ video device having ‘full duplex’ capability is often used. Display object DOA referred to above is typically mapped to the display or printing device and display object DOB is typically mapped to the keyboard.

Use of A-mode enables "typed-ahead"into display object DOB, and such updates can be delivered immediately to the peer VT-user, potentially reducing transmission delays. Such delivery will be forced, and marked, by a termination condition or a VT-DELIVER. Type-ahead is at the discretion of the terminal user.

Display object DOB has an unbounded y-dimension so as to provide a blank line for each new line entered.

Line-at-a-time forward scrolling is mapped onto an update-window (value zero) which allows NO backward updates to preceding lines (x-arrays). The device-minimum-Y-array-length negotiated by profile-argument-r3 can be used to indicate the number of lines (x-arrays) which should remain visible to the human terminal user although specifically NOT available for update.

The ability to switch local echo "on" or "off" is always present; the ECHO control object is used for this purpose.

8.6.7 Specific conformance requirements

None.

8.7 S-mode Paged Application profile

See Stable Agreements.

Annex A (normative)

Specific ASE requirements

See Stable Agreements.

Annex B (normative)

Clarifications

See Stable Agreements.

Annex C (normative)

Object identifiers

See Stable Agreements for Object Identifiers assigned to objects in the Stable Agreements. Object Identifiers below have been assigned to objects for which work is still in progress.

General Identifiers:

```
oiw-vt-rep OBJECT IDENTIFIER ::= { oiw-vt repertoire(2) }

oiw-vt-font OBJECT IDENTIFIER ::= { oiw-vt font(3) }

oiw-vt-colour OBJECT IDENTIFIER ::= { oiw-vt colour(4) }

oiw-vt-directory OBJECT IDENTIFIER ::= { oiw-vt useOfDirectory(5) }
```

Profiles defined by OIW VT SIG:

```
oiw-vt-pr-scroll-1989  OBJECT IDENTIFIER ::= { oiw-vt-pr scroll-1989(3) }
```

Control Objects defined by OIW VT SIG:

```
oiw-vt-co-tcco-tc  OBJECT IDENTIFIER ::= { oiw-vt-co-tcco  tc(0) }
```

Annex D (informative)

Recommended practice_Operating X Window System over OSI upper layers

This annex provides a "recommended practice" for the operation of the X Window System (X) over an OSI upper layer stack. The "recommended practice" provides an interim[1] solution for an area not addressed by base standards or existing profiles. This recommended practice reflects OIW agreement.

It is recommended that this interim solution be used when mapping X over an OSI upper layer stack. However, implementors should note the following_future specifications of the regional workshops may possibly result in different solutions than those proposed in this recommended practice.

D.1 Background

X is a graphical user interface standard which enables a user to view and gain access to multiple computer applications from a single window or multiple windows on a display screen. X is based on a client/server architecture which allows applications and resources to be distributed across a network.

The **X server** is a software program that is resident on a user's display unit that acts as an intermediary between the user and applications running on a local or remote system. The X server also maintains complex data structures such as specific windows, cursors and fonts which can be referenced and utilized by applications. Input from the keyboard and/or mouse is collected by the X-server and passed to local and/or remote applications for processing.

Applications are referred to as **X clients**. These applications access the display unit by sending messages to the X server which is then able to perform two dimensional drawing of lines, shapes and text.

X products are based on a de facto standard (MIT-X) maintained by the MIT X Consortium. However, this specification does not provide for the operation of X over OSI-based networks.

Two OSI mapping specifications were created to define the operation of X over an OSI upper layer stack: EWOS Technical Guide 13 (ETG13) and part 4 of ANSI dpANS X.196 (X3.196). Parts 1-3 were intended to define the X protocol. Part 4 was based on ETG13. .X3.196 never progressed beyond the draft proposal stage. ETG13 was approved by EWOS in 05/91.

ETG13 explicitly defines:

a) the required OSI upper layer facilities;

b) the mapping of the OSI upper layer services for sending and receiving X protocol.

Since the creation of these documents, the ISO ISP 11188-3 *Common Upper Layer Requirements_Part 3: Minimal OSI upper layer requirements* (CULR-3) came into existence. CULR-3 defines the minimal set

[1] It is intended that this Recommended Practice will be progressed as an RWS technical report.

of OSI upper layer facilities for basic communications applications such as X.

Unlike ETG13, this specification does not itself specify the required upper layer facilities. Rather, it references CULR-3 to indicate the required OSI upper layer facilities. On the other hand, like ETG13, it specifies the mapping of X to the OSI upper layers services (ACSE, Presentation and Session). The mapping specified is compatible with that in ETG13.

This specification is intended to be as brief as possible. ETG13 includes additional guidance and explanatory material for implementors.

D.2 Mapping specification

This clause defines the mapping of the OSI ACSE (ISO 8649) and Presentation Layer (ISO 8822) services for sending and receiving X messages. This mapping uses the following ACSE and presentation services:

a) ASSOCIATE;

b) RELEASE;

c) ABORT;

d) A-P-ABORT;

e) P-DATA.

The required ACSE, presentation and supporting session facilities are discussed in clause D.3

For the purposes of this specification, the operation of X over the OSI upper layers is referred to as **X-osi**.

D.2.1 Summary of mapping

All the X protocol Request, Reply, Error and Event messages (i.e., the "X messages") use the encodings specified in MIT-X. The X messages are treated by this mapping as unstructured stream of octets. Any arbitrary sequence of consecutive octets can be treated as a single octet-aligned presentation data value this is transmitted as the user data on a Presentation P-DATA primitive. The OPEN DISPLAY Request and Reply messages are treated in the same way, and are carried on P-DATA. This mapping does not use the user data of the ACSE services.

The OSI upper layer stack supporting X-osi shall be mOSI compliant as defined in clause D.3.

D.2.2 Association establishment

The initiative for connection and association establishment is always with the X client. The X client establishes a new association with the desired X server by issuing an A-ASSOCIATE request. As part of the A-ASSOCIATE procedure, an OSI transport-connection is established to the X server system. The class of Transport protocol is out of scope of this specification. There is no requirement for X clients or X servers to re-use OSI Transport connections.

Once the transport-connection is established, an AARQ PDU carried in a Presentation Connect request (CP PPDU) that is in turn carried in a Session Connect request (CONNECT SPDU). The parameters shall include:

a) Application Context Name : This shall be the value "x-application context", defined in ETG13 and shown below:

b) Presentation Context Definition List : Shall include the ACSE presentation context and the X-osi presentation context, using the abstract and transfer syntax names defined in ETG13 and shown below. Other contexts may be offered (these may include synonyms or alternative names for X abstract or transfer syntax);

c) Presentation context identifiers shall be integers not greater than 255. This is a more severe restriction than ISO ISP 11188-1, Common Upper Layer Requirements_Part 1: Basic connection-oriented requirements (CULR-1), that permits 2-octet integers.

d) The user information field of the A-ASSOCIATE request shall be absent.

All other parameters are subject only to the requirements of mOSI compliance (see clause D.3).

If the X server accepts the association, the Application Context Name parameter on the A-ASSOCIATE response shall have the same value as that received on the indication. The ACSE and X-osi presentation contexts shall be accepted. If synonym abstract syntax or transfer syntax names for X-osi were offered and recognized, only one shall be accepted (i.e., following this exchange, there shall be a unique presentation context established for X-osi). The user information field of the A-ASSOCIATE response shall be absent.

D.2.3 Data exchange

As stated in the summary above, once the association is established, all X-messages are carried as user data on P-DATA primitives, each carrying a single PDV-list element containing a single "octet-aligned" presentation data value, which is some sequence of consecutive octets from one or more X-messages. No correlation is required between the pdv's (i.e. between successive P-DATAs) and the division between the X-messages : the division into pdv's is entirely at the sender's option. (Obviously, in practice there will be some correlation, but there is no requirement to achieve this, nor should receivers rely on it.)

D.2.4 Connection termination

A CLOSE DISPLAY request from an X client is mapped to an A-RELEASE request. After receiving an A-RELEASE indication, the X server responds with an A-RELEASE response. Neither the request or response primitive shall contain any User Information.

A KILL CLIENT request from another client results in the issue of an A-ABORT request by the X server. A protocol or internal procedural error in either the X client or the X server also results in the issue of an A-ABORT request. The A-ABORT indication will conatin the Abort Source parameter with the value "ACSE service-user".

The receipt of an A-ABORT indication with the Abort Source parameter having the value "ACSE service-provider" indicates a failure in either the local or peer ACSE. The receipt of an A-P-ABORT indication indicates a failure in the supporting Presentation Layer or below.

D.3 Required OSI upper layer facilities.

X is a basic communications application as defined in the CULR-3. That is, it simply requires the ability to open and close communications with a peer and to send and receive messages with the peer. The required facilities of the OSI upper layers (Session, Presentation, and ACSE) are specified by stating the minimal mOSI compliance requirements as defined in the CULR-3.

mOSI compliance requirements depend on whether a system supports one or more X clients (requests an association) or X servers (accepts an association request).

D.3.1 X client mOSI compliance

An upper layer stack that supports an X client shall be mOSI compliant category I or category II.

An X client stack has the following minimal compliance requirement based on Table 2 in the CULR-3.

a) "Establishment role" shall have the value "Initiator" or "Both". An X client is always the association initiator; it is never an association-responder.

b) "Normal data role" shall have the value "Both". An X client shall be able to send or receive data.

c) "Release role" shall have the value "Requestor", or "Both". A CLOSE DISPLAY request is mapped to A-RELEASE.

d) "Authentication" shall have the value "Supported" or "Not supported". The X client - X server association does not use the ACSE Authentication functional unit.

e) "AC negotiation" shall have the value "Supported" or "Not supported". The X client - X server association does not use the ACSE Application context negotiation functional unit.

f) "All "m" parms sent and received and CULR-1 compliance?" shall have the value "Yes". If the value is "Yes", the stack is mOSI compliant, category I or category II.

g) "All "o" parms sent and received?" shall have the value "Yes" or "No." If the value is "Yes", the stack is category I. If the value is "No", the stack is of category II. In this case, the X client stack is only required to support the following features for sending(see Table 3).

_Called AE title

_ Form1 (Directory name)

D.3.2 X server mOSI compliance

An upper layer stack that supports an X server shall be mOSI compliant category I or category II. The X server stack has the following compliance requirement based on Table 2 in the CULR-3.

a) "Establishment role" shall have the value "Responder" or "Both". An X server is always the association responder; it is never an association-initiator.

b) "Normal data role" shall have the value "Both". An X server shall be able to send or receive data.

c) "Release role" shall have the value "Acceptor", or "Both". The receipt of an A-RELEASE indication indicates a CLOSE DISPLAY request from the X client.

d) "Authentication" shall have the value "Supported" or "Not supported". The X client - X server association does not use the ACSE Authentication functional unit.

e) "AC negotiation" shall have the value "Supported" or "Not supported". The X client - X server association does not use the ACSE Application context negotiation functional unit.

f) "All "m" parms sent and received?" shall have the value "Yes". If the value is "Yes", the stack is mOSI compliant, category I or category II.

g) "All "o" parms sent and received?" shall have the value "Yes" or "No". If the value is "Yes", the stack is category I. If the value is "No", the stack is of category II. No category II features are required for sending.

D.4 Object identifiers

Object identifiers used for this specification are assigned in ETG13.[2]

Application context for X-osi :
{iso(1) identified-organization(3) ewos(16) eg(2) vt(7)
x-osi(10) application-context(1) }

Abstract syntax name:
{iso(1) identified-organization(3) ewos(16) eg(2) vt(7)
x-osi(10) abstract-syntax-version-1(2) }

Transfer syntax name:
{iso(1) identified-organization(3) ewos(16) eg(2) vt(7)
x-osi(10)
binary-transfer-syntax-version-1(3) }

[2] These EWOS based object identifiers were also referenced in the last draft of X3.196_part 4.

D.5 Recommended encoding

It is recommended that the encoding of the Presentation PCI for the P-DATA follow a particular set of choices, among the optional features allowed by BER. This makes the P-DATA a (nearly) fixed header and allows implementations to be optimized to process this encoding. An implementation must be able to handle alternative encodings (i.e. any allowed by BER, subject to the restraints of CULR-1), within the mapping specification that each P-DATA carries a single octet-aligned presentation data value. The recommended encoding is :

a) the fully-encoded-data (SEQUENCE OF PDV-list) shall contain exactly one PDV-list;

b) both the SEQUENCE OF PDV-list and the PDV-list shall have indefinite length, but shall contain no levels of construction other than those required by the data types;

c) the length of the presentation-context-identifier value shall be expressed in short form;

d) the presentation-context-identifier value shall be encoded in one octet;

e) the OCTET STRING of presentation-data-values will contain a single presentation data value and shall have primitive encoding and

f) the (definite) length of this OCTET STRING shall be expressed in exactly four octets (i.e., the length itself will occupy three octets, prefixed by one octet which defines the length of this length).

These encoding choices mean that each TSDU user data consists of 16 octets of header, the X-message octets, and 4 octets of trailer (all zero). The length of the X-message segment is in the last three octets of the header.

This recommendation is identical to that in ETG13 except for the length field in (6). In ETG13 this is for a length of 1+4, not 1+3. This gives a 17-octet header. Since the X protocol, and many implementations go to some effort to get things on 4-byte boundaries, it is better to make this apply to X-osi as well. If a truly enormous P-DATA is needed i) the implementation is being very clever with its buffering; ii) it will have to use a longer length field; iii) the receiver is required to handle any legal encoding anyway.

D.6 Differences from ETG13

D.6.1 Abstract and transfer syntax names

In ETG13 the abstract and transfer syntax names are defined as names for the syntaxes defined in part II of X3.196, and ETG13 includes a copy of the April 1990 text for this. Since this is just a definition of the X data formats, there will be no problem in using them for X protocol as defined in MIT-X. ETG13 explicitly allows the extensibility features of X to be used without altering the syntax names.

Strictly speaking, X uses two transfer syntaxes, and the OPEN DISPLAY request defines which one will be used. The transfer syntax name defined in ETG13 covers both the "MSBfirst" and "LSBfirst" forms.

D.6.2 Application process title and application entity qualifier

ETG13 requires that the Called Application Process Title parameter on the A-ASSOCIATE request be a Directory Name (i.e. form1) in which the last RDN is the attribute value assertion CommonName=<displaynumber>, where <displaynumber> is a string representing the X Window System server number (and thus most commonly "0"), and that the Called Application Entity Qualifier be CommonName = "X-Window-System". The requirement was intended to facilitate X-osi : X-other relays, but this really requires integration with RFC 1275 to be general.

Although ETG13 requires these values it also recommends that implementations accept other values (or no value). Therefore there should be no interworking problems by omitting this requirement here.

~~**Annex E** (normative)~~

(ANNEX E WAS FORMALLY ANNEX D. THE WORKSHOP STYLES CHANGE THE NUMBERING AUTOMATICALLY.) ~~OIW X_{OSI} contributions~~

~~The following sections describe contributions by the OIW VT SIG to further X_{OSI}. Contributions in this section are not meant to replace any existing standards, but are meant to fill gaps where standards may not exist, or where they are still under development. Whenever possible, contributions in this section will be kept technically aligned with any developing standards.~~

~~E.1 XDMCP$_{OSI}$~~

~~E.1.1 Introduction~~

~~This section describes a means by which the protocol commonly known as XDMCP (X Display Manager Control Protocol) may be mapped over the OSI Upper Layers. This specification is meant to provide one solution to the problem of establishing an X connection in a pure OSI environment.~~

~~In general, the mapping of XDMCP to OSI relies on much of the information already present in the X_{OSI} Part IV document, and as such, that information will not be duplicated here.~~

~~E.1.2 Functional overview~~

~~XDMCP provides for three different types of query operations: Query, IndirectQuery and BroadcastQuery. The nature of the BroadcastQuery operation is such that is requires a connection less environment, and so will not be considered here.~~

~~In addition to the three query types, XDMCP provides for a number of other operations: ForwardQuery, Willing, Unwilling, Request, Accept, Decline, Manage, Refuse, and Failed. The actual semantics of these operations will not be discussed here except with regard to how they map to various OSI services. Additional information on XDMCP operations may be found in the XDMCP protocol specification provided by the MIT X Consortium.~~

~~E.1.2.1 ACSE requirements~~

~~The ACSE requirements for XDMCP$_{OSI}$ are the same as specified by X_{OSI} Part IV, with the exception of the abstract and transfer syntaxes.~~

~~E.1.2.2 Presentation requirements~~

~~XDMCP$_{OSI}$ has the same presentation requirements as specified by X_{OSI} Part IV.~~

~~E.1.2.3 Session requirements~~

~~XDMCP$_{OSI}$ has the same session requirements as specified by X$_{OSI}$ Part IV.~~

~~E.1.2.4 Abstract and transfer syntaxes~~

~~Note: We basically have two possible options here: one is to utilize the existing X$_{OSI}$ object identifiers as currently defined by EWOS, and request that EWOS register an abstract syntax which defines XDMCP, which would resemble: XDMCP-abstract-syntax :: = OBJECT IDENTIFIER { xosi, abstract-syntax(4) }. The other option would be to define the application context and abstract syntax under OIW, leaving the transfer syntax as specified by EWOS.~~

~~E.1.2.5 XDMCP to OSI mapping~~

~~The following sections outline how various XDMCP protocol elements are mapped to specific OSI services. The mapping is divided into four broad catagories, each describing mapping for specific operations based on the type of query.~~

~~Note: that the following discussion includes a possible way to map the IndirectQuery function to OSI, however, with X.500 IndirectQuery may become an unneccessary feature.~~

~~E.1.2.5.1 Query - display to manager~~

~~An implementation claiming to support XDMCP$_{OSI}$ must support this minumum functionality. The Query functionality allows a display to query a specific manager for willingness to provide management services.~~

~~A display may initiate an association with either the Query or IndirectQuery operation. Once the association is established, a display may request additional Query or IndirectQuery operations via P-Data regardless of the operation in the initial association request.~~

~~Table 1 - XDMCP mapping for query/display to manager~~

~~A-Associate Request & Indication~~	~~Query~~
~~A-Associate Response & Confirm~~	~~Willing, Unwilling~~
~~A-Release Request & Indication (manager initiates)~~	~~<null>, Unwilling, Decline, Failed~~
~~A-Release Response & Confirm~~	
~~P-Data (Display)~~	~~Query, IndirectQuery, Request, Manage~~
~~P-Data (Manager)~~	~~Willing, Unwilling, Accept, Decline, Refuse, Failed~~

~~E.1.2.5.2 IndirectQuery - display to manager~~

~~A display or manager may optionally support IndirectQuery in addition to Query. Managers which receive IndirectQuery requests in the A-Associate request which do not support IndirectQuery should refuse the association.~~

~~The manager which receives the IndirectQuery is known as the primary manager. The primary manager will forward the IndirectQuery to secondary managers via a ForwardQuery operation. Optionally, the primary manager may also indicate a willingness to perform the display management. As with secondary managers which indicate a willingness to manage, the display is under no obligation to accept the primary managers offer, and may simply release the association. The primary manager may also choose not to send a Willing, in which case it may simply release the association. However, it may be desirable for the primary manager and/or the display to maintain the association for a period of time, so as to potentially be available for additional requests from the display.~~

~~Table 2 - XDMCP mapping for indirectquery/display to manager~~

~~A-Associate Request & Indication~~	~~IndirectQuery~~
~~A-Associate Response & Confirm~~	~~<null>, Willing~~
~~A-Release Request & Indication (manager initiates)~~	~~<null>, Unwilling, Decline, Failed~~
~~A-Release Response & Confirm~~	
~~P-Data (Display)~~	~~Query, IndirectQuery, Request, Manager~~
~~P-Data (Manager)~~	~~Willing, Unwilling, Accept, Decline, Refuse, Failed~~

~~E.1.2.5.3 IndirectQuery - manager to manager~~

~~This section specifies the mapping specifically for a primary manager to communicate a ForwardQuery request to a secondary manager.~~

~~The IndirectQuery is sent by the display to a well known manager (or primary manager), which in turn forwards the request to a larger collection of secondary managers using ForwardQuery packets.~~

~~If associations haven't been previously established between the primary manager and the secondary manager, an A-Associate will be sent with the ForwardQuery operation. If an association already exists, the primary manager will simply send the ForwardQuery operation via a P-DATA. It is up to the implementation to determine how long an association between a primary and secondary manager should be maintained as various tradeoff's between association establishment overhead and maintaining an idle association exist and have not been examined in this guide.~~

~~Either the primary manager or a number of secondary managers may respond to the query by by sending a Willing response to the display which initiated the request. Secondary managers which are willing will notify the requesting display by establishing an association with the display with the WIlling operation in the A-Associate.~~

~~Table 3 - XDMCP mapping for indirectquery/manager to manager~~

~~A-Associate Request & Indication~~	~~ForwardQuery~~
~~A-Associate Response & Confirm~~	
~~P-Data~~	~~ForwardQuery~~

~~E.1.2.5.4 Indirect query - manager to display~~

~~This section specifies the mapping for a secondary manager which has received a ForwardQuery request, and wishes to respond with a WIlling to the requesting display. To perform this function, the secondary manager must first establish an association with the Willing operation in the A-Associate request. The display may accept the A-associate yet not immediately respond to the Willing indication. If the display subsequently decides not to accept the Willing from the manager in question, it will simply release the association.~~

~~Table 4 - XDMCP mapping for indirectquery/manager to display~~

~~A-Associate Request & Indication~~	~~Willing~~
~~A-Associate Response & Confirm~~	~~<null>, Request~~
~~A-Release Request & Indication (manager initiates)~~	~~<null>, Failed~~
~~A-Release Request & Indication (display initiates)~~	
~~P-Data (manager)~~	~~Accept, Decline, Refuse~~
~~P-Data (display)~~	~~Request, Manage~~