DRAFT

pr **ETS 300 777-1**

July 1996

EUROPEAN TELECOMMUNICATION STANDARD

Source: ETSI TC-TE

Reference: DE/TE-01057-1

ICS: 33.020

Key words: API, MHEG, multimedia, terminal

Terminal Equipment (TE); End-to-end protocols for multimedia information retrieval services; Part 1: Coding of multimedia and hypermedia information for basic multimedia applications (MHEG-5)

ETSI

European Telecommunications Standards Institute

ETSI Secretariat

Postal address: F-06921 Sophia Antipolis CEDEX - FRANCE
Office address: 650 Route des Lucioles - Sophia Antipolis - Valbonne - FRANCE
X.400: c=fr, a=atlas, p=etsi, s=secretariat - **Internet:** secretariat@etsi.fr

Tel.: +33 92 94 42 00 - Fax: +33 93 65 47 16

Contents

Blank page

Foreword

This draft European Telecommunication Standard (ETS) has been produced by the Terminal Equipment (TE) Technical Committee of the European Telecommunications Standards Institute (ETSI), and is now submitted for the Public Enquiry phase of the ETSI standards approval procedure.

This ETS consists of 4 parts as follows:

Part 1: "Coding of multimedia and hypermedia information for basic multimedia applications **(i.e. this part)**";

Part 2: "Use of Digital Storage Media Command and Control (DSM-CC) for basic multimedia applications "(DE/TE-01057-2);

Part 3: "Application Programmable Inteface (API) for MHEG-5 ";

Part 4: "Videotex Man Machine Interface (VEMMI) enhancements to support broadband multimedia information retrieval services" (DE/TE-01057-4).

Proposed transposition dates	
Date of latest announcement of this ETS (doa):	3 months after ETSI publication
Date of latest publication of new National Standard or endorsement of this ETS (dop/e):	6 months after doa
Date of withdrawal of any conflicting National Standard (dow):	6 months after doa

Blank page

1 Scope

This European Telecommunications Standard (ETS) specifies the Multimedia and Hypermedia information coding Experts Group (MHEG) part 5 Abstract Syntax Notation One (ASN.1) notation consisting of a syntax description (equivalent to the Extended Backus Naur Form (EBNF) syntax) and encoding rules.

MHEG part 5 (ISO/IEC DIS 13522-5 [1]) specifies the coded representation of interchanged multimedia/hypermedia information objects (MHEG-5 objects) for use in the domain of base-level interactive applications such as movies-on-demand, teleshopping, near video-on-demand.

MHEG-5 specifies objects and their semantics using an informal text description. It also provides a formal description of the interchanged objects syntax using EBNF.

This specification is included in ISO/IEC DIS 13522-5 [1] as normative annex A.

2 Normative references

This ETS incorporates by dated or undated reference, provisions from other publications. These normative references are cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to this ETS only when incorporated in it by amendment or revision. For undated references the latest edition of the publication referred to applies.

[1] ISO/IEC DIS 13522-5 (1996): "Information technology - Coding of Multimedia and Hypermedia information - Part 5: Support for Base-Level Interactive Applications".

[2] ISO/IEC 8824-1 (1990)/ITU-T Recommendation X.680 (1995): "Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation - Amendment 1: Rules of extensibility ".

[3] ISO/IEC 8825-1 (1990)/ITU-T Recommendation X.690 (1995): "Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of this ETS, the following definitions in ISO/IEC DIS 13522-5 [1], ISO/IEC 8824-1 [2] and ISO/IEC 8825-1 [3] apply:

3.2 Abbreviations

For the purposes of this ETS, the following abbreviations apply.

ASN.1	Abstract Syntax Notation One
BER	Basic Encoding Rules
DER	Distinguished Encoding Rules
DIS	Draft International Standard
EBNF	Extended Backus Naur Form
TE	Terminal Equipment

4 MHEG-5 ASN.1 notation design principles

The MHEG-5 ASN.1 notation was designed to achieve compactness of encoded objects.

In order to fulfil this criteria SEQUENCE and CHOICE types have been carefully tagged.

Within a SEQUENCE type, the type of an element is either implicitly encoded if the element is tagged or explicitly encoded if the element is not tagged. The size of the code is only the same if the tag value is less than 31. To reduce tag values within SEQUENCE types, the scope of tagging is not the whole sequence. Every mandatory element of a SEQUENCE closes a context and therefore allows tag values to be reused and tags to be avoided where possible.

EXAMPLE:

In SEQUENCE S two tagging contexts can be identified: {a,b,c} and {d,e,f}.

An ASN.1 compiler using the option "AUTOMATIC TAGS" would tag it as follows:

```
S ::= SEQUENCE
{
a                                   [0] INTEGER OPTIONAL,
b                                   [1] BOOLEAN OPTIONAL,
c                                   [2] INTEGER,
d                                   [3] INTEGER OPTIONAL,
e                                   [4] BOOLEAN OPTIONAL,
f                                   [5] INTEGER
}
```

In the MHEG-5 ASN.1 notation this type would be tagged as follows:

```
S ::= SEQUENCE
{
a                                   INTEGER OPTIONAL,
b                                   BOOLEAN OPTIONAL,
c                                   [0] INTEGER,
d                                   INTEGER OPTIONAL,
e                                   BOOLEAN OPTIONAL,
f                                   [0] INTEGER
}
```

A SEQUENCE type may be included in several other SEQUENCE types through the COMPONENTS OF statement. Then the tagging of the included type has to be compatible with the tagging of all embedding types.

EXAMPLE:

If SEQUENCE S is not included by other types the tagging is as follows:

```
S ::= SEQUENCE
{
a                                   INTEGER OPTIONAL,
b                                   BOOLEAN OPTIONAL,
c                                   [0] INTEGER
}
```

If included by SEQUENCE types S1 and S2 this type would be tagged as follows:

```
S ::= SEQUENCE
{
a                                   [0] INTEGER OPTIONAL,
b                                   [1] BOOLEAN OPTIONAL,
c                                   [2] INTEGER
}

S1 ::= SEQUENCE
{
a1                                  INTEGER OPTIONAL,
COMPONENTS OF                       S
}

S2 ::= SEQUENCE
{
a2                                  BOOLEAN OPTIONAL,
COMPONENTS OF                       S
}
```

For CHOICE types, the tagging method aims at letting these types remain untagged when they are used in a SEQUENCE or within another CHOICE type in order to reduce the size of the code.

EXAMPLE:

An ASN.1 compiler using the option "AUTOMATIC TAGS" would tag the CHOICE types A, B and C as follows:

```
A ::= CHOICE
{
a1                                  [0] INTEGER,
a2                                  [1] B,
a3                                  [2] BOOLEAN
}

B ::= CHOICE
{
b1                                  [0] INTEGER,
b2                                  [1] C,
b3                                  [2] BOOLEAN
}

C ::= CHOICE
{
c1                                  [0] INTEGER,
c2                                  [1] BOOLEAN
}
```

In the MHEG-5 ASN.1 notation this type would be tagged as follows:

```
A ::= CHOICE
{
a1                                  INTEGER,
a2                                  B,
a3                                  [3] BOOLEAN
}

B ::= CHOICE
{
b1                                  [0] INTEGER,
b2                                  C,
b3                                  [2] BOOLEAN
}

C ::= CHOICE
{
c1                                  [1] INTEGER,
c2                                  BOOLEAN
}
```

5 MHEG-5 ASN.1 notation

```
ISOMHEG-mh {joint-iso-itu-t(2) mheg(19) version(1) mheg-5(17)}
DEFINITIONS IMPLICIT TAGS ::= BEGIN

RootClass ::= SEQUENCE {
object-identifier                   ObjectReference,
no-caching                          BOOLEAN DEFAULT FALSE
}

GroupClass ::= SEQUENCE {
COMPONENTS OF RootClass,
standard-version                    INTEGER DEFAULT 0,
object-information                  OCTET STRING OPTIONAL,
on-start-up                         ActionClass OPTIONAL,
on-close-down                       [0] ActionClass OPTIONAL,
cache-priority                      [1] INTEGER DEFAULT 127,
items                               [2] SEQUENCE OF Item
}

Item ::= CHOICE {
procedure-class                     ProcedureClass,
palette-class                       [0] PaletteClass,
font-class                          [1] FontClass,
cursor-shape-class                  [2] CursorShapeClass,
variable-class                      [3] VariableClass,
link-class                          [4] LinkClass,
stream-class                        [5] StreamClass,
audio-class                         [6] AudioClass,
video-class                         [7] VideoClass,
rtgraphics-class                    [8] RTGraphicsClass,
bitmap-class                        [9] BitmapClass,
lineart-class                       [10] LineArtClass,
rectangle-class                     [11] RectangleClass,
hotspot-class                       [12] HotspotClass,
switch-button-class                 [13] SwitchButtonClass,
push-button-class                   [14] PushButtonClass,
text-class                          [15] TextClass,
```

```
entryfield-class                [16] EntryFieldClass,
hypertext-class                 [17] HypertextClass,
slider-class                    [18] SliderClass,
token-group-class               [19] TokenGroupClass,
template-group-class            [20] TemplateGroupClass,
list-class                      [21] ListClass
}

ApplicationClass ::= SEQUENCE {
OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) mheg-5(17) application (1)},
COMPONENTS OF GroupClass,
on-spawn-close-down             ActionClass OPTIONAL,
on-restart                      [0] ActionClass OPTIONAL
}

SceneClass ::= SEQUENCE {
OBJECT IDENTIFIER DEFAULT {joint-iso-itu-t(2) mheg(19) version(1) mheg-5(17) scene(2)},
COMPONENTS OF GroupClass,
input-event-register            INTEGER,
scene-coordinate-system         SceneCoordinateSystem,
aspect-ratio                    AspectRatio OPTIONAL,
moving-cursor                   BOOLEAN DEFAULT FALSE,
next-scenes                     [0] SEQUENCE OF NextScene OPTIONAL
}

SceneCoordinateSystem ::= SEQUENCE {
x-scene                         INTEGER,
y-scene                         INTEGER
}

AspectRatio ::= SEQUENCE {
width                           INTEGER DEFAULT 4,
height                          [0] INTEGER DEFAULT 3
}

NextScene ::= SEQUENCE {
scene-ref                       OCTET STRING,
scene-weight                    INTEGER
}

IngredientClass ::= SEQUENCE {
COMPONENTS OF RootClass,
initially-active                [0] BOOLEAN DEFAULT TRUE,
content-hook                    ContentHook OPTIONAL,
content-data                    ContentData OPTIONAL,
content-size                    [2] INTEGER OPTIONAL,
shared                          [3] BOOLEAN DEFAULT FALSE
}

ContentHook ::= CHOICE {
mheg-content-register           INTEGER,
proprietary-content-register    [1] INTEGER
}

ContentData ::= CHOICE {
included-content                OCTET STRING,
referenced-content              ContentReference
}
```

```
LinkClass ::= SEQUENCE {
COMPONENTS OF IngredientClass,
link-condition                  [4] LinkCondition,
link-effect                     ActionClass
}

LinkCondition ::= SEQUENCE {
event-source                    ObjectReference,
event-type                      EventType,
event-data                      EventData OPTIONAL
}

EventType ::= ENUMERATED {
is-available (1), is-deleted (2), is-running (3), is-stopped (4), user-input (5), timer-fired (6), asynch-stopped
(7), is-modified (8), token-moved-from (9), token-moved-to (10), stream-event (11), stream-playing (12),
stream-stopped (13), counter-trigger (14), highlight-on (15), highlight-off (16), cursor-enter (17), cursor-
leave (18), new-char (19), anchor-fired (20), is-selected (21), is-deselected (22)
}

EventData ::= CHOICE {
octetstring-value               OCTET STRING,
boolean-value                   BOOLEAN,
integer-value                   INTEGER
}

ProcedureClass ::= SEQUENCE {
COMPONENTS OF IngredientClass,
procedure-type                  ProcedureType,
procedure-name                  OCTET STRING OPTIONAL,
procedure-connection-tag        INTEGER OPTIONAL
}

ProcedureType ::= ENUMERATED {custom (1), remote (2), script (3)}

PaletteClass ::= IngredientClass

FontClass ::= IngredientClass

CursorShapeClass ::= IngredientClass

VariableClass ::= SEQUENCE {
COMPONENTS OF IngredientClass,
original-value                  OriginalValue
}

OriginalValue ::= CHOICE {
boolean-value                   [4] BOOLEAN,
integer-value                   [5] INTEGER,
octetstring-value               [6] OCTET STRING,
object-reference                [7] ObjectReference,
content-reference               [8] ContentReference
}

PresentableClass ::= IngredientClass

TokenManagerClass ::= SEQUENCE {
movement-table                  [4] SEQUENCE OF Movement OPTIONAL
}

Movement ::= SEQUENCE OF INTEGER
```

```
TokenGroupClass ::= SEQUENCE {
COMPONENTS OF PresentableClass,
COMPONENTS OF TokenManagerClass,
token-group-items                   [5] SEQUENCE OF TokenGroupItem
}

TokenGroupItem ::= SEQUENCE {
a-visible                           Avisible,
action-slots                        SEQUENCE OF ActionSlot OPTIONAL
}

Avisible ::= CHOICE {
text-class                          TextClass,
video-class                         [0] VideoClass,
rtgraphics-class                    [1] RTGraphicsClass,
bitmap-class                        [2] BitmapClass,
lineart-class                       [3] LineArtClass,
rectangle-class                     [4] RectangleClass,
hotspot-class                       [5] HotspotClass,
switch-button-class                 [6] SwitchButtonClass,
push-button-class                   [7] PushButtonClass,
slider-class                        [8] SliderClass,
hypertext-class                     [9] HypertextClass,
entryfield-class                    [10] EntryFieldClass
}

ActionSlot ::= CHOICE {
action                              ActionClass,
null-action                         NULL
}

TemplateGroupClass ::= SEQUENCE {
COMPONENTS OF PresentableClass,
COMPONENTS OF TokenManagerClass,
multiple-selection                  [5] BOOLEAN DEFAULT FALSE,
item-template                       [6] ItemTemplate,
items-data                          SEQUENCE OF ItemData,
on-get-token                        ActionClass OPTIONAL,
on-lose-token                       [0] ActionClass OPTIONAL,
on-toggle                           [1] ActionClass OPTIONAL,
on-select                           [2] ActionClass OPTIONAL,
on-deselect                         [3] ActionClass OPTIONAL,
template-actions                    [4] SEQUENCE OF TemplateAction OPTIONAL,
cell-positions                      [5] SEQUENCE OF CellPosition
}

ItemTemplate ::= SEQUENCE {
template-visibles                   SEQUENCE OF TemplateVisible OPTIONAL,
template-variables                  [0] SEQUENCE OF VariableClass OPTIONAL
}

TemplateVisible ::= SEQUENCE {
a-visible                           Avisible,
relative-to-scene                   BOOLEAN
}

ItemData ::= SEQUENCE {
visibles-content                    SEQUENCE OF VisibleContent OPTIONAL,
variables-content                   [0] SEQUENCE OF VariableContent OPTIONAL
```

```
}

VisibleContent ::= CHOICE {
content-reference                ContentReference,
null-visible                     NULL
}

VariableContent ::= CHOICE {
octetstring-value                OCTET STRING,
integer-value                    INTEGER,
boolean-value                    BOOLEAN,
object-reference                 ObjectReference,
content-reference                [0] ContentReference,
null-value                       NULL
}

TemplateAction ::= CHOICE {
action                           ActionClass,
null-action                      NULL
}

CellPosition ::= SEQUENCE {
x-cell                           INTEGER,
y-cell                           INTEGER
}

ListClass ::= SEQUENCE {
COMPONENTS OF TemplateGroupClass,
on-visible                       ActionClass OPTIONAL,
on-invisible                     [0] ActionClass OPTIONAL,
on-at-first-position             [1] ActionClass OPTIONAL,
on-leave-first-position          [2] ActionClass OPTIONAL,
on-at-last-position              [3] ActionClass OPTIONAL,
on-leave-last-position           [4] ActionClass OPTIONAL,
fixed-token                      BOOLEAN DEFAULT FALSE,
wrap-around                      [5] BOOLEAN DEFAULT FALSE
}

VisibleClass ::= SEQUENCE {
COMPONENTS OF PresentableClass,
original-box-size                [4] OriginalBoxSize,
original-position                OriginalPosition OPTIONAL
}

OriginalBoxSize ::= SEQUENCE {
x-length                         INTEGER,
y-length                         INTEGER
}

OriginalPosition ::= SEQUENCE {
x-position                       INTEGER DEFAULT 0,
y-position                       [0] INTEGER DEFAULT 0
}

BitmapClass ::= SEQUENCE {
COMPONENTS OF VisibleClass,
bitmap-palette-ref               [0] ObjectReference OPTIONAL,
drop-out-colour                  DropOutColour OPTIONAL,
transparency                     [1] INTEGER DEFAULT 0
}
```

```
DropOutColour ::= CHOICE {
integer-value                   INTEGER,
octetstring-value               OCTET STRING
}

LineArtClass ::= SEQUENCE {
COMPONENTS OF VisibleClass,
line-width                      INTEGER DEFAULT 1,
line-style                      LineStyle DEFAULT solid,
lineart-palette-ref             [0] ObjectReference OPTIONAL,
ref-line-colour                 Colour,
ref-fill-colour                 Colour OPTIONAL
}

LineStyle ::= ENUMERATED {solid (1), dashed (2), dotted (3)}

RectangleClass ::= LineArtClass

TextClass ::= SEQUENCE {
COMPONENTS OF VisibleClass,
font                            Font OPTIONAL,
font-attributes                 [1] OCTET STRING OPTIONAL,
text-colour                     [2] Colour OPTIONAL,
background-colour               [3] Colour OPTIONAL,
text-palette-ref                [4] ObjectReference OPTIONAL,
character-set                   INTEGER DEFAULT 2,
horizontal-justification        HorizontalJustification DEFAULT start,
vertical-justification          [5] VerticalJustification DEFAULT start,
line-orientation                [6] LineOrientation DEFAULT horizontal,
start-corner                    [7] StartCorner DEFAULT upper-left,
text-wrapping                   BOOLEAN DEFAULT FALSE
}

Font ::= CHOICE {
font-name                       OCTET STRING,
font-reference                  [0] ObjectReference
}

HorizontalJustification ::= ENUMERATED {start (1), end (2), centre (3), justified (4)}

VerticalJustification ::= ENUMERATED {start (1), end (2), centre (3), justified (4)}

LineOrientation ::= ENUMERATED {vertical (1), horizontal (2)}

StartCorner ::= ENUMERATED {upper-left (1), upper-right (2), lower-left (3), lower-right (4)}

StreamClass ::= SEQUENCE {
COMPONENTS OF PresentableClass,
multiplex                       [4] SEQUENCE OF StreamComponent,
storage                         Storage DEFAULT stream,
looping                         INTEGER DEFAULT 1
}

StreamComponent ::= SEQUENCE {
stream-tag                      INTEGER,
stream-content                  StreamContent
}
```

```
StreamContent ::= CHOICE {
audio-class                     AudioClass,
video-class                     [0] VideoClass,
rtgraphics-class                [1] RTGraphicsClass
}

Storage ::= ENUMERATED {memory (1), stream (2)}

AudioClass ::= SEQUENCE {
COMPONENTS OF PresentableClass,
original-volume                 [4] INTEGER DEFAULT 0
}

VideoClass ::= SEQUENCE {
COMPONENTS OF VisibleClass,
termination                     VideoTermination DEFAULT disappear
}

VideoTermination ::= ENUMERATED {freeze (1), disappear (2)}

RTGraphicsClass ::= VisibleClass

InteractibleClass ::= SEQUENCE {
engine-resp                     [8] BOOLEAN DEFAULT TRUE,
highlight-ref-colour            [9] Colour OPTIONAL,
interactible-palette-ref        [10] ObjectReference OPTIONAL
}

SliderClass ::= SEQUENCE {
COMPONENTS OF VisibleClass,
COMPONENTS OF InteractibleClass,
orientation                     Orientation,
initial-value                   INTEGER OPTIONAL,
initial-portion                 [0] INTEGER OPTIONAL,
min-value                       [1] INTEGER DEFAULT 1,
max-value                       [2] INTEGER,
step-size                       INTEGER DEFAULT 1,
slider-style                    SliderStyle DEFAULT normal,
slider-colour1                  Colour,
slider-colour2                  Colour
}

Orientation ::= ENUMERATED {left (1), right (2), up (3), down (4)}

SliderStyle ::= ENUMERATED {normal (1), thermometer (2), proportional (3)}

EntryFieldClass ::= SEQUENCE {
COMPONENTS OF TextClass,
COMPONENTS OF InteractibleClass,
input-type                      [11] InputType DEFAULT any,
char-list                       [12] OCTET STRING OPTIONAL,
obscured-input                  [13] BOOLEAN DEFAULT FALSE,
max-length                      [14] INTEGER DEFAULT 0
}

InputType ::= ENUMERATED {alpha (1), numeric (2), any (3), listed (4)}
```

```
HypertextClass ::= SEQUENCE {
COMPONENTS OF TextClass,
COMPONENTS OF InteractibleClass
}

ButtonClass ::= SEQUENCE {
COMPONENTS OF VisibleClass,
COMPONENTS OF InteractibleClass,
ref-colour1                     Colour OPTIONAL,
ref-colour2                     [0] Colour OPTIONAL
}

HotspotClass ::= ButtonClass

PushButtonClass ::= SEQUENCE {
COMPONENTS OF ButtonClass,
label                           [2] OCTET STRING OPTIONAL
}

SwitchButtonClass ::= SEQUENCE {
COMPONENTS OF ButtonClass,
label                           [2] OCTET STRING OPTIONAL
}

ActionClass ::= SEQUENCE OF ElementaryAction

ElementaryAction ::= CHOICE {
activate                        GenericObjectReference,
bring-to-front                  [1] GenericObjectReference,
call-action-slot                [2] CallActionSlot,
call-template-action            [3] CallTemplateAction,
close-connection                [4] CloseConnection,
deactivate                      [5] GenericObjectReference,
deselect                        [6] GenericObjectReference,
deselect-item                   [7] GenericObjectReference,
get-availability-status         [8] GetAvailabilityStatus,
get-cursor-position             [9] GetCursorPosition,
get-engine-support              [10] GetEngineSupport,
get-entry-point                 [11] GetEntryPoint,
get-item-content                [12] GetItemContent,
get-running-status              [13] GetRunningStatus,
get-selection-status            [14] GetSelectionStatus,
get-slider-value                [15] GetSliderValue,
get-state                       [16] GetState,
get-text-data                   [17] GetTextData,
launch                          [18] GenericObjectReference,
lock-screen                     [19] GenericObjectReference,
move                            [20] Move,
move-to                         [21] MoveTo,
open-connection                 [22] OpenConnection,
plug                            [23] Plug,
preload                         [24] GenericObjectReference,
put-before                      [25] PutBefore,
put-behind                      [26] PutBehind,
quit                            [27] GenericObjectReference,
read-persistent                 [28] ReadPersistent,
run                             [29] GenericObjectReference,
run-asynchronous                [30] RunAsynchronous,
run-synchronous                 [31] RunSynchronous,
scale-bitmap                    [32] ScaleBitmap,
```

```
scale-video                     [33] ScaleVideo,
scroll                          [34] Scroll,
select                          [35] GenericObjectReference,
select-item                     [36] GenericObjectReference,
send-event                      [37] SendEvent,
send-to-back                    [38] GenericObjectReference,
set-bitmap-palette-ref          [39] SetBitmapPaletteRef,
set-box-size                    [40] SetBoxSize,
set-cache-priority              [41] SetCachePriority,
set-counter-position            [42] SetCounterPosition,
set-counter-trigger             [43] SetCounterTrigger,
set-cursor-position             [44] SetCursorPosition,
set-cursor-shape                [45] SetCursorShape,
set-data                        [46] SetData,
set-entry-point                 [47] SetEntryPoint,
set-fill-colour                 [48] SetFillColour,
set-font-ref                    [49] SetFontRef,
set-highlight-status            [50] SetHighlightStatus,
set-interactible-palette        [51] SetInteractiblePalette,
set-interaction-status          [52] SetInteractionStatus,
set-item-content                [53] SetItemContent,
set-label                       [54] SetLabel,
set-lineart-palette             [55] SetLineartPalette,
set-line-colour                 [56] SetLineColour,
set-line-width                  [57] SetLineWidth,
set-overwrite-mode              [58] SetOverwriteMode,
set-portion                     [59] SetPortion,
set-position                    [60] SetPosition,
set-slider-value                [61] SetSliderValue,
set-speed                       [62] SetSpeed,
set-state                       [63] SetState,
set-text-palette-ref            [64] SetTextPaletteRef,
set-timer                       [65] SetTimer,
set-transparency                [66] SetTransparency,
set-variable                    [67] SetVariable,
set-volume                      [68] SetVolume,
spawn                           [69] GenericObjectReference,
stop                            [70] GenericObjectReference,
step                            [71] Step,
store-persistent                [72] StorePersistent,
test-variable                   [73] TestVariable,
toggle                          [74] GenericObjectReference,
toggle-item                     [75] GenericObjectReference,
transition-to                   [76] TransitionTo,
unload                          [77] GenericObjectReference,
unlock-screen                   [78] GenericObjectReference,
unplug                          [79] Unplug
}

CallActionSlot ::= SEQUENCE {
target                          GenericObjectReference,
index                           GenericInteger
}

CallTemplateAction ::= SEQUENCE {
target                          GenericObjectReference,
index                           GenericInteger
}
```

```
CloseConnection ::= SEQUENCE {
target                          GenericObjectReference,
connection-tag                  GenericInteger
}

GetAvailabilityStatus ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference
}

GetCursorPosition ::= SEQUENCE {
target                          GenericObjectReference,
x-out                           ObjectReference,
y-out                           ObjectReference
}

GetEngineSupport ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference
}

GetEntryPoint ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference
}

GetItemContent ::= SEQUENCE {
target                          GenericObjectReference,
content-variable                ObjectReference
}

GetRunningStatus ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference
}

GetSelectionStatus ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference
}

GetSliderValue ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference
}

GetState ::= SEQUENCE {
target                          GenericObjectReference,
state-variable                  ObjectReference
}

GetTextData ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference
}

Move ::= SEQUENCE {
target                          GenericObjectReference,
movement-identifier             GenericInteger
}
```

```
MoveTo ::= SEQUENCE {
target                          GenericObjectReference,
index                           GenericInteger
}

OpenConnection ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference,
protocol                        GenericOctetString,
address                         GenericOctetString,
connection-tag                  GenericInteger
}

Plug ::= SEQUENCE {
target                          GenericObjectReference,
item-data                       NewItemData,
index                           GenericInteger OPTIONAL
}

NewItemData ::= SEQUENCE {
new-visibles-content            SEQUENCE OF NewVisibleContent OPTIONAL,
new-variables-content           [0] SEQUENCE OF NewVariableContent OPTIONAL
}

NewVisibleContent ::= CHOICE {
new-content-reference           GenericContentReference,
null-visible                    NULL
}

NewVariableContent ::= CHOICE {
octetstring-value               GenericOctetString,
integer-value                   GenericInteger,
boolean-value                   [0] GenericBoolean,
object-reference                [2] GenericObjectReference,
content-reference               [3] GenericContentReference,
null-value                      NULL
}

PutBefore ::= SEQUENCE {
target                          GenericObjectReference,
reference-visible               GenericObjectReference
}

PutBehind ::= SEQUENCE {
target                          GenericObjectReference,
reference-visible               GenericObjectReference
}

ReadPersistent ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference,
out-variables                   SEQUENCE OF ObjectReference,
in-file-name                    GenericOctetString
}
```

```
RunAsynchronous ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference,
in-parameters                   InParameters,
out-parameters                  OutParameters
}

InParameters ::= SEQUENCE {
in-booleans                     SEQUENCE OF GenericBoolean OPTIONAL,
in-integer                      [0] SEQUENCE OF GenericInteger OPTIONAL,
in-octetstring                  [1] SEQUENCE OF GenericOctetString OPTIONAL,
in-object-reference             [2] SEQUENCE OF GenericObjectReference OPTIONAL,
in-content-reference            [3] SEQUENCE OF GenericContentReference OPTIONAL
}

OutParameters ::= SEQUENCE {
out-booleans                    SEQUENCE OF ObjectReference OPTIONAL,
out-integer                     [0] SEQUENCE OF ObjectReference OPTIONAL,
out-octetstring                 [1] SEQUENCE OF ObjectReference OPTIONAL,
out-object-reference            [2] SEQUENCE OF ObjectReference OPTIONAL,
out-content-reference           [3] SEQUENCE OF ObjectReference OPTIONAL
}

RunSynchronous ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference,
in-parameters                   InParameters,
out-parameters                  OutParameters
}

ScaleBitmap ::= SEQUENCE {
target                          GenericObjectReference,
x-scale                         GenericInteger,
y-scale                         GenericInteger
}

ScaleVideo ::= SEQUENCE {
target                          GenericObjectReference,
x-scale                         GenericInteger,
y-scale                         GenericInteger
}

Scroll ::= SEQUENCE {
target                          GenericObjectReference,
scroll-type                     ScrollType,
scroll-number                   GenericInteger
}

ScrollType ::= ENUMERATED {up (1), down (2), absolute (3)}

SendEvent ::= SEQUENCE {
target                          GenericObjectReference,
event-source                    ObjectReference,
event-type                      EventType,
emulated-event-data             EmulatedEventData OPTIONAL
}
```

```
EmulatedEventData ::= CHOICE {
generic-boolean                 GenericBoolean,
generic-integer                 [0] GenericInteger,
generic-octet-string            GenericOctetString
}

SetBitmapPaletteRef ::= SEQUENCE {
target                          GenericObjectReference,
new-bitmap-palette              GenericObjectReference
}

SetBoxSize ::= SEQUENCE {
target                          GenericObjectReference,
x-box-size                      GenericInteger,
y-box-size                      GenericInteger
}

SetCachePriority ::= SEQUENCE {
target                          GenericObjectReference,
new-cache-priority              GenericInteger
}

SetCounterPosition ::= SEQUENCE {
target                          GenericObjectReference,
new-counter-position            GenericInteger
}

SetCounterTrigger ::= SEQUENCE {
target                          GenericObjectReference,
trigger-identifier              GenericInteger,
new-counter-value               GenericInteger OPTIONAL
}

SetCursorPosition ::= SEQUENCE {
target                          GenericObjectReference,
x-cursor                        GenericInteger,
y-cursor                        GenericInteger
}

SetCursorShape ::= SEQUENCE {
target                          GenericObjectReference,
new-cursor-shape                GenericObjectReference OPTIONAL
}

SetData ::= SEQUENCE {
target                          GenericObjectReference,
new-content                     NewContent
}

NewContent ::= CHOICE {
new-included-content            GenericOctetString,
new-referenced-content          GenericContentReference
}

SetEntryPoint ::= SEQUENCE {
target                          GenericObjectReference,
new-entry-point                 GenericInteger
}
```

```
SetFillColour ::= SEQUENCE {
target                          GenericObjectReference,
new-fill-colour                 NewColour
}

NewColour ::= CHOICE {
colour-index                    GenericInteger,
absolute-colour                 GenericOctetString
}

SetFontRef ::= SEQUENCE {
target                          GenericObjectReference,
new-font-ref                    NewFontRef
}

NewFontRef ::= CHOICE {
font-name                       GenericOctetString,
font-reference                  ObjectReference
}

SetHighlightStatus ::= SEQUENCE {
target                          GenericObjectReference,
new-highlight-status            GenericBoolean
}

SetInteractiblePalette ::= SEQUENCE {
target                          GenericObjectReference,
new-interactible-palette        GenericObjectReference
}

SetInteractionStatus ::= SEQUENCE {
target                          GenericObjectReference,
new-interaction-status          GenericBoolean
}

SetItemContent ::= SEQUENCE {
target                          GenericObjectReference,
new-items-content               GenericOctetString
}

SetLabel ::= SEQUENCE {
target                          GenericObjectReference,
new-label                       GenericOctetString
}

SetLineartPalette ::= SEQUENCE {
target                          GenericObjectReference,
new-lineart-palette             GenericObjectReference
}

SetLineColour ::= SEQUENCE {
target                          GenericObjectReference,
new-line-colour                 NewColour
}

SetLineWidth ::= SEQUENCE {
target                          GenericObjectReference,
line-width                      INTEGER
}
```

```
SetOverwriteMode ::= SEQUENCE {
target                          GenericObjectReference,
new-overwrite-mode              GenericBoolean
}

SetPortion ::= SEQUENCE {
target                          GenericObjectReference,
new-portion                     GenericInteger
}

SetPosition ::= SEQUENCE {
target                          GenericObjectReference,
new-x-position                  GenericInteger,
new-y-position                  GenericInteger
}

SetSliderValue ::= SEQUENCE {
target                          GenericObjectReference,
new-slider-value                GenericInteger
}

SetSpeed ::= SEQUENCE {
target                          GenericObjectReference,
new-speed                       Rational
}

Rational ::= SEQUENCE {
nominator                       GenericInteger,
denominator                     GenericInteger OPTIONAL
}

SetState ::= SEQUENCE {
target                          GenericObjectReference,
new-state                       GenericOctetString
}

SetTextPaletteRef ::= SEQUENCE {
target                          GenericObjectReference,
new-text-palette                GenericObjectReference
}

SetTimer ::= SEQUENCE {
target                          GenericObjectReference,
timer-id                        GenericInteger,
timer-value                     GenericInteger OPTIONAL
}

SetTransparency ::= SEQUENCE {
target                          GenericObjectReference,
new-transparency                GenericInteger
}

SetVariable ::= SEQUENCE {
target                          GenericObjectReference,
new-variable-value              NewVariableValue
}

NewVariableValue ::= CHOICE {
generic-integer                 GenericInteger,
generic-boolean                 [0] GenericBoolean,
```

```
generic-octet-string            GenericOctetString,
generic-object-reference        [2] GenericObjectReference,
generic-content-reference       [3] GenericContentReference
}

SetVolume ::= SEQUENCE {
target                          GenericObjectReference,
new-volume                      GenericInteger
}

Step ::= SEQUENCE {
target                          GenericObjectReference,
nb-of-steps                     GenericInteger
}

StorePersistent ::= SEQUENCE {
target                          GenericObjectReference,
result                          ObjectReference,
in-variables                    SEQUENCE OF ObjectReference,
out-file-name                   GenericOctetString
}

TestVariable ::= SEQUENCE {
target                          GenericObjectReference,
comparison-value                ComparisonValue,
action-true                     ActionClass,
action-false                    ActionClass
}

ComparisonValue ::= CHOICE {
generic-integer                 GenericInteger,
generic-boolean                 [0] GenericBoolean,
generic-octet-string            GenericOctetString,
generic-object-reference        [2] GenericObjectReference,
generic-content-reference       [3] GenericContentReference
}

TransitionTo ::= SEQUENCE {
target                          GenericObjectReference,
connection-tag                  GenericInteger OPTIONAL,
transition-effect               [0] GenericInteger
}

Unplug ::= SEQUENCE {
target                          GenericObjectReference,
index                           GenericInteger OPTIONAL
}

ObjectReference ::= SEQUENCE {
group-identifier                OCTET STRING OPTIONAL,
object-number                   INTEGER
}

ContentReference ::= SEQUENCE {
public-identifier               OCTET STRING OPTIONAL,
system-identifier               [0] OCTET STRING
}
```

```
GenericObjectReference ::= CHOICE {
direct-reference                    ObjectReference,
indirect-reference                  [0] ObjectReference
}

GenericContentReference ::= CHOICE {
content-reference                   ContentReference,
indirect-reference                  [0] ObjectReference
}

GenericInteger ::= CHOICE {
integer-value                       INTEGER,
indirect-reference                  ObjectReference
}

GenericBoolean ::= CHOICE {
boolean-value                       BOOLEAN,
indirect-reference                  ObjectReference
}

GenericOctetString ::= CHOICE {
octetstring-value                   OCTET STRING,
indirect-reference                  [1] ObjectReference
}

Colour ::= CHOICE {
colour-index                        [1] INTEGER,
absolute-colour                     OCTET STRING
}

END
```

History

Document history			
July 1996	Public Enquiry	PE 110:	1996-07-22 to 1996-11-15